Multi-Objective Optimization with Estimation of Distribution Algorithm in a Noisy Environment

Vui Ann Shim g0800438@nus.edu.sg
Department of Electrical and Computer Engineering,
National University of Singapore, 117576, Singapore

Kay Chen Tan[*] eletankc@nus.edu.sg
Department of Electrical and Computer Engineering,
National University of Singapore, 117576, Singapore

Jun Yong Chia g0900313@nus.edu.sg
Department of Electrical and Computer Engineering,
National University of Singapore, 117576, Singapore

Abdullah Al Mamun eleaam@nus.edu.sg
Department of Electrical and Computer Engineering,
National University of Singapore, 117576, Singapore

Abstract

Many real-world optimization problems are subjected to uncertainties that may be characterized by the presence of noise in the objective functions. The estimation of distribution algorithm (EDA), which models the global distribution of the population for searching tasks, is one of the evolutionary computation techniques that deals with noisy information. This paper studies the potential of EDAs; particularly an EDA based on restricted Boltzmann machines that handles multi-objective optimization problems in a noisy environment. Noise is introduced to the objective functions in the form of a Gaussian distribution. In order to reduce the detrimental effect of noise, a likelihood correction feature is proposed to tune the marginal probability distribution of each decision variable. The EDA is subsequently hybridized with a particle swarm optimization algorithm in a discrete domain to improve its search ability. The effectiveness of the proposed algorithm is examined via eight benchmark instances with different characteristics and shapes of the Pareto optimal front. The scalability, hybridization, and computational time are rigorously studied. Comparative studies show that the proposed approach outperforms other state of the art algorithms.

Keywords

Estimation of distribution algorithm, evolutionary algorithm, multi-objective optimization, noisy fitness function, particle swarm optimization, restricted Boltzmann machine.

[*]Corresponding Author.

1 Introduction

Evolutionary algorithms (EAs; Tomassini, 1996; Bäck et al., 1997; Zhang et al., 2011), a class of stochastic search techniques, have been successfully applied to many real-world optimization problems (Cebrian et al., 2009; Damas et al., 2011; Yogev et al., 2010). These algorithms draw inspiration from evolution and employ biologically inspired operators such as mutation and crossover to sample, learn, and adapt from a population of candidate solutions. However, it was reported in Larrañaga and Lozano (2001) that the variation operators in standard EAs may disrupt the building block of good schemas and make the movement toward optimality extremely difficult to predict. Motivated by the idea of exploiting the linkage information among the decision variables, the estimation of distribution algorithm (EDA) has been regarded as a new computing paradigm in the field of evolutionary computation (Mühlenbein and Paass, 1996; Larrañaga and Lozano, 2001; Lozano et al., 2006; Chen and Chen, 2010). Neither crossover nor mutation is implemented in EDAs. Instead, the reproduction is carried out by building a representative probabilistic model where new solutions are subsequently generated through the sampling of a corresponding probabilistic model. Recently, EDAs have been adapted into a multi-objective optimization framework, commonly known as multi-objective estimation of distribution algorithms (MOEDAs; Pelikan et al., 2006; Shim et al., 2010).

Though multi-objective evolutionary algorithms (MOEAs; Tan et al., 2005; Fonseca and Fleming, 1995; Deb, 2001; Coello Coello et al., 2002; Tan et al., 2009) have gained satisfactory performance for certain static optimization problems, the implementation of MOEAs in a noisy environment still requires major study. In noisy environments, the presence of noise in the cost functions may affect the ability of the algorithms to drive the search process toward optimality. To understand the detrimental effect of noise in evolutionary optimization processes, Beyer (2000) carried out an investigation and found that the presence of noise may reduce the convergence rate, resulting in suboptimal solutions. In another study by Goh and Tan (2007), it was reported that the low level of noise helps an MOEA to produce better solutions for some problems, but a higher noise level may degenerate the optimization process into a random search. Darwen and Pollack (1999) concluded that resampling can reduce the effect of noise for a small population, but may not be as helpful for a larger population. In order to reduce the effect of noise in evolutionary processes, a number of noise handling approaches such as probability dominance (Hughes, 2001; Teich, 2001), fitness inheritance (Bui et al., 2005), and Dempster-Shafer framework of dominance (Limbourg, 2005) have been proposed.

While the implementation of standard MOEAs in a noisy environment has gained some attention from researchers, the adoption of MOEDAs in this particular case remains unexplored. As such, this paper attempts to study the performance of MOEDAs under the influence of noise. The advantage MOEDAs have over standard MOEAs in handling noisy information comes from the construction of noise handling features in the probabilistic model (Hong, Ren, and Zeng, 2005). Hong, Ren, Zeng, et al. (2005) showed that when a smoothing filter is incorporated into an EDA, it is more suited to tackle noisy optimization problems than a genetic algorithm (GA) in single objective optimization. More specifically, the authors argued that the univariate marginal distribution algorithm (UMDA), one of the simplest EDAs, is able to converge to global optimality in the One-max problem under the influence of noise.

In this study, we propose the use of EDAs to tackle multi-objective optimization problems in a noisy environment based on the restricted Boltzmann machine (REDA). However, REDA uses global information only in guiding the search, which may not explore the search space thoroughly. In addition, REDA may be trapped at local optima (Tang et al., 2010). To overcome this limitation, REDA is hybridized with a particle swarm optimization (PSO; Robinson and Rahmat-Samii, 2004; Goh et al., 2010) algorithm. The PSO is another stochastic computing algorithm inspired by swarm intelligence of insects in forming groups and movements such as birds and fishes. PSO is chosen because of its ease of implementation and reduced sensitivity to parameter settings. In addition, PSO has already been successfully applied to many real-world optimization problems (Iqbal et al., 2008; Azevedo et al., 2010) due to its satisfactory search performance and fast convergence rate. The hybridization is expected to improve the performance since the particles may move out of the region modeled by the probabilistic model, providing extra solutions which are unexplored by the model. On the other hand, the presence of noise may affect the selection operator, resulting in a wrong decision. This may directly affect the plausibility of the probabilistic model and cause premature convergence or trap the algorithm at local optima. Therefore, a likelihood correction feature is proposed to tune the marginal probability in each decision variable. The likelihood correction takes advantage of the probability error. The probability error is calculated from the model suggested by Hughes (2001) to determine the probability of making a wrong decision during the selection process due to the presence of noise.

The rest of the paper is as follows. Section 2 briefly describes the background concept of dominance in multi-objective optimization, MOEDAs, and noise handling features. Section 3 presents the framework of the proposed algorithm. Implementation, test instances, and performance indicators are outlined in Section 4. Section 5 examines the performance of the algorithms under the influences of noise, scalability, hybridization and computational time. The conclusion is presented in Section 6.

2 Background Information

This section briefly reviews the concept of dominance in multi-objective optimization, MOEDAs, and noise handling features for MOEAs.

2.1 Concept of Dominance in Multi-Objective Optimization

Without loss of generality, the test problems are considered to have m objective functions and n decision variables. Considering the minimization case, the problem is formulated as follows.

Minimize:

$$f(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)$, $x \in \mathbb{R}^n$, $\mathbb{R}^n$ is the decision space, x is the decision vector, $f(x) \in \mathbb{R}^m$, $\mathbb{R}^m$ is the objective space, and $f(x)$ is the objective vector. Let a and b be two decision vectors, then a is said to dominate b ($a \prec b$) if and only if

$$f_i(a) \leq f_i(b)\ \forall i \in \{1, 2, \ldots, m\} \tag{2}$$

and a and b are incomparable ($a \sim b$) if and only if

$$\exists i \in \{1, 2, \ldots, m\} : f_i(a) > f_i(b)\ \ and\ \ \exists j \in \{1, 2, \ldots, m\} : f_j(a) < f_j(b) \tag{3}$$

A decision vector $x^* \in \mathbb{R}^n$ is said to be nondominated if and only if $\nexists b \in \mathbb{R}^n : b \prec x^*$ and x^* is a Pareto optimal solution. The set of all Pareto optimal points is called the Pareto-optimal set (PS) and the corresponding objective vectors form the Pareto-optimal front (PF; Deb, 2001).

2.2 Multi-Objective Estimation of Distribution Algorithms (MOEDAs)

EDAs were introduced as a new computing paradigm in the field of evolutionary computation (Mühlenbein and Paass, 1996). In EDAs, the offspring are produced by sampling the estimated probability distribution of the parent population. EDAs can be divided into different categories according to the level of interaction among the decision variables that their probabilistic model performs (Larrañaga and Lozano, 2001). Detailed information about the interaction among the decision variables can be found in Ting et al. (2010) and Zhu et al. (2010). Recently, several EDAs have been proposed in the context of multi-objective optimization.

Bosman and Thierens (2002) presented a mixture distribution as a probabilistic model where each component in the mixture is an univariate factorization. This algorithm is known as multi-objective mixture-based iterated density estimation evolutionary algorithm (MIDEA). MIDEA has an advantage in preserving diversity due to its widespread exploration capability toward the Pareto optimal front. This algorithm can serve as a baseline algorithm for MOEDAs for its simplicity, speed, and effectiveness. In another study, Laumanns and Ocenasek (2002) incorporated a Bayesian optimization algorithm based on the binary decision tree as a model building technique—the Bayesian multi-objective optimization algorithm (BMOA)—to approximate the set of Pareto optimal solutions. In Costa and Minisci (2003), the authors proposed the multi-objective Parzen-based EDA (MOPED) to approximate the probability distribution of the solutions lying on the Pareto optimal front. Furthermore, a spreading technique which uses a Parzen estimator in the phenotypic space to identify the crowding level of the solutions was also proposed. The algorithm showed promising results in several well-known test problems.

Li et al. (2004) proposed a hybrid EDA (MOHEDA) by integrating a local search that is based on the weighted sum method while its constraint handling is based on a random repair method for solving multi-objective 0/1 knapsack problems. A stochastic clustering method was also introduced to preserve the diversity of the solutions. The results showed that MOHEDA outperforms several state of the art algorithms. Zhang et al. (2008) proposed a regularity model-based multi-objective estimation of the distribution algorithm (RM-MEDA) by considering regularity in building the probabilistic model. A local principle component analysis was used to extract the regularity patterns of the candidate solutions from previous searches. This algorithm showed good performance in test instances with a nonlinear variable linkage for a scalable number of variables. Okabe et al. (2004) introduced Voronoi diagrams to construct the probability model in an algorithm called Voronoi-based EDA (VEDA). VEDA is able to adjust the reproduction process based on the problem structure and at the same time, it takes advantage of the problem structure by estimating the most appropriate probability distribution.

Pelikan et al. (2005) described a scalable algorithm by combining non-dominated sorting GA II (NSGAII) (Deb, Pratap, et al., 2002), hierarchical Bayesian optimization algorithm (hBOA) and clustering in the phenotypic space to solve decomposable problems. In the proposed algorithm, each cluster is allocated an almost equal number of solutions. The results showed that the algorithm outperforms many standard MOEAs when the numbers of decision variables are scaled up. Another study was carried out by Zhong and Li (2007) where a decision-tree-based multi-objective estimation of distribution algorithm (DT-MEDA) for continuous-valued optimization problems was developed. The conditional dependencies among the decision variables were extracted by a decision-tree-based probabilistic model.

Recently, several attempts were carried out to model neural networks as EDAs in the framework of multi-objective optimization. Marti et al. (2009) implemented a growing neural gas network for modeling purposes. The algorithm showed promising results in high-dimensional problems. In another study, Tang et al. (2010) adapted an unsupervised learning neural network of restricted Boltzmann machine (RBM) to capture interdependent information among the decision variables. The network learns the probability distribution of the input stimuli in terms of energy equilibrium. The final information is returned to each decision variable by clamping the synaptic weights and biases into a marginal probability distribution. The algorithm performed well in high-dimensional problems.

2.3 Noise Handling Features

Over the past few years, several noise handling techniques have been proposed in the area of utilizing MOEAs to reduce the detrimental effect of noise. Hughes (2001) proposed the multi-objective probabilistic selection evolutionary algorithm (MOPSEA) where the ranking process is reformulated by incorporating the probability dominance among the solutions. The modified ranking process showed promising results in reducing the disturbance of noise. In another independent study by Teich (2001), the concept of probabilistic dominance was introduced to deal with uncertain objective values constrained within certain intervals. This concept was implemented in the strength Pareto evolutionary algorithm (SPEA) and called estimate SPEA (ESPEA).

Büche et al. (2002) proposed a noise-tolerant SPEA (NTSPEA). The study focused on improving the robustness of the solutions when the objective functions are subjected to noise and outliers. Three features were incorporated, including domination dependent lifetime, re-evaluation of solutions, and update of the archive population. A lifetime, which is dependent on the fraction of archived solutions it dominates, is assigned to each of the nondominated individuals. The solutions in the archive will be removed once the lifetime has expired.

Goh and Tan (2007) carried out an extensive investigation to understand the effect of noise in evolutionary multi-objective optimization. They found that the decision error ratio for selection, ranking, and archiving is lower in the early stages of an evolution, the number of nondominated solutions found in an archive is reduced when the noise level increases and the average of the population distribution remains invariant in the decision space. Based on these observations, three noise handling features were proposed. Experiential learning directed perturbation (ELDP) performs the mutation paradigm by deciding which gene will undergo mutation. This decision is based on after the fact knowledge of the favorable movements in the search space. Gene adaptation selection strategy (GASS) adopts the feature to prevent search processes from premature convergence or degeneration into random searches. Finally, a possibilistic archiving methodology updates the archive by considering the probability that the domination by an individual is wrong.

Recently, Bui et al. (2009) adapted local models in dealing with noisy multi-objective optimization problems. In the proposed algorithm, the search space is divided into several nonoverlapping hyper-spheres in order to limit the search within the spheres. Noise filtering was carried out in each hyper-sphere and the future movement of the sphere was adjusted according to the direction of improvement. This implementation reduces the effect of noise based on the movement of the spheres. Other approaches that have been proposed to reduce the effect of noise are fitness inheritance (Bui et al., 2005), indicator model (Basseur et al., 2006), stochastic dominance and significant

Algorithm 1. Framework of PLREDA

```
Begin
    Initialization: At generation $g=0$, randomly generate $N$ individuals as the initial
    population $Pop(0)$; generate empty archives $A1$ and $A2$.
    Evaluation: Evaluate all the solutions in $Pop(g)$.
    Do While ("maximum generation is not reached")
        1.  Fitness Assignment: Perform Pareto ranking and crowding distance over the
            solutions in $Pop\ (g)$.
        2.  Selection: Select $N$ solutions based on binary tournament selection to form a new
            population $Pop\ (g)$.
            Probability Dominance: For every solution that wins the tournament, calculate its
            probability error according to Equation (10).
        3.  Clustering: Cluster the solutions in $Pop\ (g)$ according to Equation (18).
        4.  RBM Training: Train the RBM by using the CD training method.
        5.  Modeling using likelihood correction: Construct the probabilistic model
            according to Equation (19).
        6.  Sampling: Sample $N$ new solutions based on the probabilistic model built in Step
            5 according to Equation (7). Store the offspring in $A1$.
        7.  Evaluation: Calculate the objective values of all offspring in $A1$.
        8.  Elitism: Combine population $Pop\ (g)$ with offspring in $A1$ to form a pool of $2N$
            solutions. Select $N$ solutions to form a new $Pop\ (g)$.
        9.  PSO: Perform PSO update to all solutions in $Pop\ (g)$. Store the offspring in $A2$.
        10. Evaluation: Calculate the objective values of all offspring in $A2$ .
        11. Elitism: Combine population $Pop\ (g)$ with $A2$ to form a pool of $2N$ individuals.
            Select $N$ individuals to form $Pop\ (g+1)$. $g=g+1$.
    End Do
        12. Final population: Return $Pop\ (g)$ as the final population.
End
```

dominance (Eskandari and Geiger, 2009), the Dempster-Shafer framework of dominance (Limbourg, 2005), and confidence-based operators (Boonma and Suzuki, 2009; Syberfeldt et al., 2010), among others.

2.4 Noisy Test Instances

In this paper, noise is introduced into the fitness functions in the objective space as Gaussian noise, $N(0, \sigma^2)$, with zero mean and different noise levels represented by variance (σ^2). Mathematically, the noisy fitness function is modeled as:

$$F(x) = f(x) + N(0, \sigma^2) \tag{4}$$

3 Proposed Algorithm

REDA surpasses the standard MOEAs in handling noisy information by constructing a noise handling feature in the built probabilistic model. In order to show this advantage, likelihood correction is proposed in order to tune the error in the constructed probabilistic model. Furthermore, REDA is hybridized with PSO in order to improve its search ability. In this section, the framework of the proposed algorithm and other relevant features will be discussed.

3.1 Algorithm Framework

The proposed algorithm incorporates PSO and likelihood correction in REDA and is called PLREDA. The pseudocode of the PLREDA is presented in Algorithm 1 and the algorithm works as follows. First, at generation $g = 0$, N initial individuals are randomly generated to form an initial population, $Pop(g = 0)$. All the solutions in $Pop(g)$

are evaluated to calculate their objective values. Based on the objective values, Pareto ranking, and crowding distance (Deb, Pratap, et al., 2002) are performed over the solutions. Next, binary tournament selection (Miller and Goldberg, 1995; M. Chakraborty and U. Chakraborty, 1997) is carried out. In each selection process, the probability error in selecting a particular individual is computed. Based on this information, the population is clustered into several groups. The population is then rendered into RBM. The RBM is subsequently trained by using the contrastive divergence (CD; Hinton, 2002) training method to obtain the corresponding weights and biases. The probabilistic model using likelihood correction is then built. From the constructed probabilistic model, N new offspring are sampled, and the solutions are stored in archive $A1$. The solutions in $Pop(g)$ and $A1$ are combined to form a pool of $2N$ solutions. Subsequently, N solutions with the lowest Pareto rank and highest crowding distance are selected from the pool to form the new $Pop(g)$. Next, PSO is performed to update the solutions in $Pop(g)$ and the generated N new solutions are stored in archive $A2$. The solutions in $Pop(g)$ and $A2$ are then combined to form a pool of $2N$ solutions. N solutions with the lowest Pareto rank and highest crowding distance are selected from the pool to form the next population $Pop(g+1)$. The same process is continued until the terminating criterion is reached.

3.2 Particle Swarm Optimization (PSO)

REDA has its limitations in exploring the search space and may be trapped at any local optimum (Tang et al., 2010). In order to overcome these limitations, hybridization is carried out (Neri and Mininno, 2010; Ong et al., 2010). In the implementation stage, REDA is hybridized with a PSO algorithm. This hybridization is expected to improve the performance since the particles may now move out of the regions modeled by REDA, and thus provide extra solutions that REDA alone was not able to tap into.

PSO is another stochastic computing algorithm that is inspired by the swarm intelligence of several animals presenting collective behavior, such as birds and fishes. PSO has gained much attention from research communities in solving real-world optimization problems due to its ease in implementation and great search performance. PSO is particularly efficient for problems presented in real values. The conversion of PSO from real number representation to binary number representation is direct (Engelbrecht, 2006). The first discrete PSO was developed by Kennedy and Eberhart (1997). In binary PSO, each particle represents a position in binary space. Cedeno and Agrafiotis (2002) implemented a binary PSO in feature selection by treating the position vectors as continuous-valued vectors. The continuous-valued vectors are then transformed to discrete-valued vectors by setting a threshold to decide between the bit values 0 and 1.

In this implementation, the velocity ($v_{k,d}(g)$) of each dth dimension of particle k in generation g is updated according to the continuous-valued case as:

$$v_{k,d}(g+1) = \alpha v_{k,d}(g) + c_1 r_1 \left(pbest_{k,d}(g) - x_{k,d}(g)\right) + c_2 r_2 \left(gbest_d(g) - x_{k,d}(g)\right) \quad (5)$$

where $v_{k,d} \in [vmax_d, vmin_d]$, $vmax_d$ and $vmin_d$ are the maximum and minimum velocity in the dth dimension, respectively. $gbest_d(g)$ is the location in dth dimension parameter space of the best fitness returned for the entire swarm in generation g. In this implementation, tournament selection is applied to find the best individual at the current iteration. The best individual is the one that is nondominated and less crowded by other solutions. $pbest_{k,d}(g)$ is the location in the dth dimension parameter space of the best fitness returned for a particle k in generation g. The factors r_1 and r_2 are uniformly

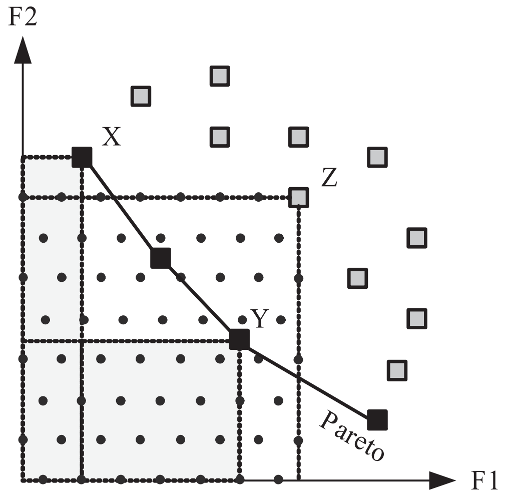


Figure 1: Concept of dominance.

distributed random variables in the range of [0 1], while c_1 and c_2 are the weights that regulate the influences between global and local information. Another factor α, known as initial weight, aims to balance the global and local search abilities. After updating the velocity, the position of each particle is updated according to the equations as follows.

$$y_{k,d}(g+1) = y_{k,d}(g) + v_{k,d}(g+1) \tag{6}$$

$$x_{k,d}(g+1) = \begin{cases} 1 & \text{if } y_{k,d}(g+1) \geq \mathit{Threshold} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $y_{k,d}$ is the updated position in the dth dimension of particle k in a continuous domain while $x_{k,d}$ is the corresponding position value in a discrete domain after transformation by a reference threshold. The threshold is evolutionary and is treated as a variable to be optimized.

3.3 Probability Dominance

The concept of probability dominance was proposed in Hughes (2001). In PLREDA, this concept is implemented to determine the probability error of each selected individual. This information is used to group the population into several clusters before a probabilistic model is built.

Assume that $f_i(A)$ and $f_i(B)$ are two solutions in the objective space with m objective functions ($i = 1, 2, \ldots, m$). In a noise-free situation, $f_i(A)$ is said to strictly dominate $f_i(B)$ if $f_i(A)$ is smaller than $f_i(B)$ in all the objective values in the minimization case. On the other hand, $f_i(A)$ and $f_i(B)$ are mutually nondominated only if not all the objective values in one solution are lower than that of the other. Figure 1 further illustrates the concept of dominance. In the figure, point Z is strictly dominated by point Y, while points X and Y are mutually nondominated.

In a noisy domain, the above statements may not correctly represent the true domination behavior. Even through $f_i(A)$ appears to strictly dominate $f_i(B)$, the noise may distort the actual fitness values where $f_i(B)$ is supposed to dominate $f_i(A)$. In the selection process, the selection error occurs when the less fit individual is chosen. Therefore,

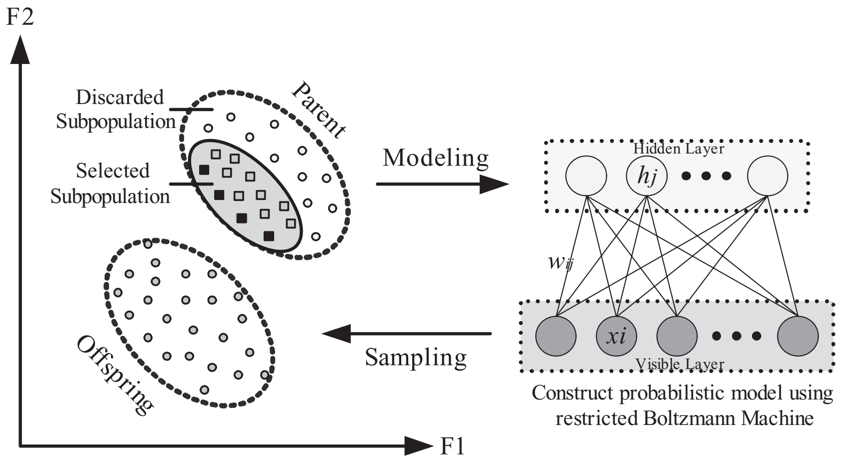


Figure 2: Evolution process using RBM.

the probability to make an error in the selection process could be utilized to improve the decision making process.

Hughes (2001) suggested a probabilistic selection model which employs the probabilistic dominance to account for the noise effect in the objective space. The noise is assumed to be normally distributed. In this model, $P(f_i(A) < f_i(B))$ is the probability error that $f_i(A)$ dominates $f_i(B)$ and is calculated as:

$$P\left(f_i\left(A\right) < f_i\left(B\right)\right) = 0.5 + 0.5\, erf\left(\frac{f_i(A) - f_i(B)}{2\sigma_n}\right) \tag{8}$$

where σ_n is the standard deviation of the distribution of the noise, and $erf(x)$ is the error function. However, the calculation of the error function is time-consuming. Therefore, the probability error (P_e) is approximated as follows.

$$P_{e,i} \equiv P\left(f_i\left(A\right) < f_i\left(B\right)\right) \approx 0.5 + 0.5 \tanh\left(\frac{f_i(A) - f_i(B)}{1.6\sigma_n}\right) \tag{9}$$

For m objectives, the final probability (P_e) that solution A dominates solution B is calculated as follows.

$$P_e = P\left(F\left(A\right) < F\left(B\right)\right) = \prod_{i=1}^{m} P\left(f_i(A) < f_i(B)\right) \tag{10}$$

3.4 Restricted Boltzmann Machine in Noisy Environments

3.4.1 Restricted Boltzmann Machine as Estimation of Distribution Algorithm (REDA)

In this paper, REDA was applied to build the probabilistic model. This algorithm was proposed by Tang et al. (2010). REDA estimates the joint probability distribution of the selected individuals by using an RBM. An RBM is a kind of neural network that learns the probability distribution in terms of energy equilibrium. The network consists of an input layer (visible layer) and a hidden layer. The network is simple and efficient in modeling, able to learn interdependencies among the decision variables, and easy to implement. The evolution process of REDA is illustrated in Figure 2. First of all, the

Algorithm 2. CD training. The $\varphi(y) = \frac{1}{1+e^{-y}}$ is the logistic function, b_i is the bias for the visible unit, b_j is the bias for the hidden unit, $P(x_i|h)$ is the conditional probability returned for x_i given h, and $P(h_j|x)$ is the conditional probability of h_j given x. $<>_0$ denotes the states of the neurons before reconstruction and $<>_1$ denotes the states of the neurons after a single step of reconstruction.

```
Begin
    Input: The decision vectors of the selected solutions (candidate solutions) are rendered to the visible units $< x_i >_0$ of
           the RBM.
    Do While ("maximum training epoch is not reached")
            %Positive phase
            1.  Compute the conditional probability of the hidden units given the visible units as
                    $P(h_j|x) = \varphi\left(\sum_i w_{ij} x_i - b_j\right)$
            2.  Sample the hidden states from $P(h_j|x)$.
            %Negative phase
            3.  Recompute the visible states $< x_i >_1$ by sampling according to
                    $P(x_i|h) = \varphi\left(\sum_j w_{ij} h_j - b_i\right)$
            4.  Recompute the hidden states $< h_i >_1$ by sampling according to Step 1.
            5.  Update the weights and biases
                    $w'_{ij} = w_{ij} + \epsilon\,(< x_i h_j >_0 - < x_i h_j >_1)$
                    $b'_i = b_i + \epsilon\,(< x_i >_0 - < x_i >_1)$
                    $b'_j = b_j + \epsilon\,(< h_j >_0 - < h_j >_1)$
    End Do
    Output: Trained weights and biases that are used to compute the energy function of the
            network.
End
```

selected candidate solutions are rendered into the RBM. The decision vectors of the candidate solutions are treated as the input vectors to the visible units in the RBM. The CD training (see Algorithm 2) is performed to obtain the weights and biases of the network.

In the CD training, two phases are carried out before the weights and biases of the network are updated. During the positive phase, the conditional probability of the hidden units given the visible states is computed according to Algorithm 2 (Step 1). Subsequently, the states of the hidden units are sampled. During the negative phase, the hidden units are stochastically fired to reconstruct the visible units by sampling according to Algorithm 2 (Step 3). Next, the hidden units are recomputed from the visible layer according to Algorithm 2 (Step 1). This process is called one step reconstruction of the visible and hidden units. Next, the weights and biases of the network are updated according to Algorithm 2 (Step 5). One training epoch is stopped here. The same procedures (positive and negative phases) are continued until the maximum number of training epoch is reached. The output of the network is the marginal probability distribution of the decision variables. The distribution is factorized as a product of independent univariate marginal distribution of each decision variable. Finally, sampling is performed to generate N offspring. This completes a generation.

In binary representation, the joint probability distribution at generation g is formulated as:

$$p_g(x) = p(x|Pop) = \prod_{i=1}^{n} p_g(x_i) \tag{11}$$

where

$$p_g(x_i = 1) = \frac{P(x_i)^+}{P(x_i)^+ + P(x_i)^-} \tag{12}$$

$$P(x_i)^+ = \sum_{h=1}^{H} e^{-E(x_i=1,h)} \tag{13}$$

$$P(x_i)^- = \sum_{h=1}^{H} e^{-E(x_i=0,h)} \tag{14}$$

$$E(x, h) = -\sum_{i=1}^{n}\sum_{j=1}^{H} x_i h_j w_{ij} - \sum_{i=1}^{n} x_i b_i - \sum_{j=1}^{H} h_j b_j \tag{15}$$

and where x_i is the ith decision variable, n is the number of decision variables, H is the number of hidden units, $P(x_i)^+$ is the marginal cost of x_i when the cardinality of $x_i = 1$, $P(x_i)^-$ is the marginal cost of x_i when the cardinality of $x_i = 0$, and E is the energy function of the RBM. This model has a limitation during the training process when the marginal probability distribution of any decision variable reaches a maximum value of 1.0 or a minimum value of 0.0. In order to alleviate this limitation, the lower and upper bounds are added to the probability distribution. The final probability distribution is constructed as follows.

$$p_g(x_i = 1) = \frac{P(x_i)^+ + \text{avg}\,(P(x_i))}{P(x_i)^+ + P(x_i)^- + r_i \text{avg}\,(P(x_i))} \tag{16}$$

where avg$(P(x))$ is the average cost of cardinality and r_i is the number of different values that x_i may take. In the binary case, r_i is 2. Based on this probability distribution, the offspring are generated by sampling the probability distribution according to Equation (17).

$$x_i = \begin{cases} 1 & \text{if random}\,(0, 1) \le p_g\,(x_i) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

3.4.2 Likelihood Correction

In a noisy condition, the selected candidate solutions may not represent the best solutions (best in terms of lower Pareto rank or larger crowding distance). Therefore, the probabilistic model built by the REDA may not correctly represent the real distribution of the best solutions. In order to improve the appropriate probabilistic model, likelihood correction is proposed. This correction is based on the heuristic that if the distribution can be approximated as close to the real distribution of the best solutions, the detrimental effect of the noise can then be reduced. To approximate the real distribution, the probability of making an error in the selection process is adapted in the probabilistic modeling. In binary tournament selection, if two solutions in the tournament have a huge distinction in their objective values, for example $f_i(A)$ dominates $f_i(B)$ by far, then the selection error for selecting $f_i(A)$ is small. On the other hand, if two individuals in the tournament are near to each other in the objective space, then the selection error is larger. Therefore, if the probabilistic model built by the REDA is only based on individuals with small selection error, then the model may avoid distortions caused by those solutions with a large selection error.

However, the probability distribution may not come close to the real distribution if the number of individuals with smaller selection error is too little. Thus, a method to combine the distribution between solutions with small selection error and those with large selection error is designed. This combination is based on the penalty approach where individuals with smaller selection error will be penalized less while solutions with larger selection error will be more heavily penalized. This is because the real distribution is more likely to follow the distribution of the population with smaller selection error than those with larger selection error.

Thus, the first step is to determine the number of groups (Gp) and then sort the individuals into Gp clusters according to the predefined threshold (Th_y, $y = \{1, 2, \ldots, \mathrm{Gp} - 1\}$) of the probability error (P_e) according to Equation (18).

$$x_i = \begin{cases} 1 & \text{if } 0 \leq P_e < \mathrm{Th}_1 \quad \text{then } P_e = 0.0 \\ 2 & \text{if } \mathrm{Th}_1 \leq P_e < \mathrm{Th}_2 \quad \text{then } P_e = \mathrm{Th}_1 \\ \vdots & \qquad \vdots \qquad\qquad \vdots \\ \mathrm{Gp} & \text{if } \mathrm{Th}_{\text{Gp-1}} \leq P_e < 1 \quad \text{then } P_e = \mathrm{Th}_{\text{Gp-1}} \end{cases} \tag{18}$$

P_e is calculated by using Equation (10). Th_y, the threshold, is determined through experimental investigation. In Equation (18), the function of Th_y is to define the criterion for grouping. For example, solutions with P_e less than Th_1 will be clustered into the first group. Solutions with P_e in between Th_1 and Th_2 will be clustered into the second group, and so forth. Afterward, the P_e is redetermined to be the value of the threshold (penalty value). Then the distribution in each cluster is combined according to Equation (19).

$$p_g(x_i = 1) = \frac{P(x_i)^+(1 - P_e) + \mathrm{avg}(P(x_i))}{P(x_i)^+(1 - P_e) + P(x_i)^-(1 - P_e) + r_i \mathrm{avg}(P(x_i))} \tag{19}$$

In Equation (19), the penalty assigned to each cluster is determined by the threshold of the probability error of the members in the cluster. The probability error P_e in cluster 1 is always treated as 0.0 because cluster 1 has the smallest selection error compared to other clusters; thus, the probability distribution contributed by the solutions in this cluster will not be penalized. The solutions in other clusters will be penalized (according to the threshold value) since these solutions may not be the fittest ones in the population.

4 Implementation

This section presents the test functions, performance metrics, and other algorithms used in the comparison. The experimental settings of the implementation are also outlined. In the experimental studies, the assumption is made that there is advance knowledge of the presence of noise.

4.1 Test Instances

Eight benchmark test instances, including ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, DTLZ1, ZDTL2, and DTLZ3, are used to test the performance of the PLREDA in terms of converging to the true Pareto optimal front and maintaining a set of diverse solutions. All of the test functions have different characteristics and shapes of the Pareto optimal front. ZDT problems (Zitzler et al., 2000) consist of two objectives functions while DTLZ problems (Deb, Thiele, et al., 2002) possess three objective functions. Detailed information on these test problems can be found in their original papers.

4.2 Performance Metrics

Three performance indicators are used to show the numerical comparison between the PLREDA and other state of the art algorithms. Let PF* be the evolved solutions and PF be the Pareto optimal solutions.

4.2.1 Generational Distance (GD)

GD (Veldhuizen and Lamont, 1998) is defined as

$$\mathrm{GD} = \sqrt{\frac{\sum_{i=1}^{N} d(p, p^*)_i^2}{N}} \tag{20}$$

where N is the number of solutions in PF*, $p \in \mathrm{PF}$, $p^* \in \mathrm{PF}^*$, and $d(p, p^*)_i$ is the minimum Euclidean distance in the objective space between p^* and p for each member i. GD illustrates the convergence ability of the algorithm by measuring the closeness between the Pareto optimal front and the evolved Pareto front. Thus, a lower value of GD shows that the evolved Pareto front is closer to the Pareto optimal front.

4.2.2 Maximum Spread (MS)

This metric measures how well the PF is covered by PF*. The mathematical formulation of the metric is shown below.

$$MS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[\frac{\min(f_i^{\max} - F_i^{\max}) - \max(f_i^{\min} - F_i^{\min})}{F_i^{\max} - F_i^{\min}}\right]^2} \tag{21}$$

where $F_i^{\max}$ and $F_i^{\min}$ are the maximum and minimum of the ith objective in PF, respectively; and $f_i^{\max}$ and $f_i^{\min}$ are the maximum and minimum of the ith objective in PF*, respectively (Zitzler et al., 2000). A higher value of MS indicates that the evolved Pareto front has a better spread.

4.2.3 Inverted Generational Distance (IGD)

IGD performs the near similar calculation as done by GD. The difference is that GD calculates the distance of each solution in PF* to PF while IGD calculates the distance of each solution in PF to PF*. In this indicator, both convergence and diversity are taken into consideration. A lower value of IGD implies that the algorithm has better performance.

4.3 Other Algorithms Used in Comparison

Seven algorithms were chosen for performance comparison with the PLREDA. Since PLREDA was developed based on REDA, REDA should be included in the comparison. Likelihood correction is introduced into REDA and forms LREDA. This algorithm could show the advantages of incorporating the likelihood correction. Other than that, five state of the art algorithms are also chosen for performance comparison. First, NSGAII (Deb, Pratap, et al., 2002) is a popular algorithm in evolutionary multi-objective optimization for its ability to generate promising solutions for most of the test problems. The resampling mechanism is incorporated into NSGAII and the algorithm is called RNSGAII. Next, NTSPEA (Büche et al., 2002) and MOPSEA (Hughes, 2001) are algorithms which were specifically proposed to handle noisy objective functions. The fundamental noise handling principle in MOPSEA is based on the probability dominance. Finally, MOEARF (Goh and Tan, 2007) is a recently published noise handling MOEA.

Table 1: Parameter settings.

Representation	Binary representation with 15 bits per decision variable
Population size	100 in ZDT problems and 200 in DTLZ problems
Archive size	100 in ZDT problems and 200 in DTLZ problems
Selection	Binary tournament selection
Fitness evaluations	40,000 in ZDT problems and 80,000 in DTLZ problems
Noise levels	0%, 5%, 10%, 20%
Independent runs	30
Crossover rate in NSGAII, RNSGAII, NTSPEA, MOPSEA, and MOEARF	0.8 (Goh and Tan, 2007)
Mutation rate in NSGAII, RNSGAII, NTSPEA, MOPSEA, and MOEARF	1/(number of variables x number of bits per variable) (Goh and Tan, 2007)
Re-sampling in RNSGAII	4
Hidden units in REDA, LREDA, and PLREADA	10 in ZDT problems and 5 in DTLZ1 problems as suggested in Tang et al. (2010)
Training epochs in REDA, LREDA, and PLREADA	20 in all test problems as suggested in Tang et al. (2010)
Number of groups in LREDA and PLREDA	3
Thresholds (Th_y) in LREDA and PLREDA	$Th_1 = 0.25$, $Th_2 = 0.5$
α in PLREDA	0.99
c_1 and c_2 in PLREDA	2 and 1.5
v_{max} and v_{min} in PLREDA	1.0 and −1.0

4.4 Experimental Settings

All algorithms were implemented in C++ and ran on an Intel Core 2 Duo, 3.0 GHz personal computer. The experimental settings are presented in Table 1. The parameter settings were determined through preliminary experimental studies or taken from previous works.

5 Computational Results

In this section, the studies are divided into four categories. The comparison results of the algorithms are presented in the first category. This is followed by an analysis of the scalability behavior of the algorithms. The possibility of hybridization with other evolutionary paradigms is discussed next. Finally, the computational time for the algorithms to perform a single simulation run is examined.

5.1 Comparison Results

Table 2 shows the results in terms of the GD measurement obtained from the different algorithms. The first column indicates the test problems, while the number in the parentheses is the number of the decision variables. The mean and standard deviation of the results over 30 independent runs are presented. The best result in each test instance is highlighted in bold. It is obvious from the tabulation that PLREDA is able to obtain the best GD results in most of the test instances, followed by MOEARF and MOPSEA. In a noiseless condition, MOPSEA obtained the best results in four of the test instances, with MOEARF and PLREDA each obtaining two best results. When the noise levels are increased, PLREDA outperformed other algorithms except for DTLZ2. In addition, we can see the improvement from REDA to LREDA in most of the test instances, with the noisy condition. The performance of LREDA is further enhanced when it is hybridized with PSO (PLREDA). It was also observed that the hybridization

Table 2: Comparison of PLREDA to the different MOEAs with respect to the GD performance indicator

Problems	Noise levels	Algorithms							
(*n*)	(%)	PLREDA	LREDA	REDA	NSGAII	NTSPEA	MOPSEA	RNSGAII	MOEARF
ZDT1 (30)	0	0.0045±0.0016	0.0038±0.0003	0.0038±0.0003	0.0038±0.0003	0.0063±0.00058	**0.0035±0.00026**	0.0038±0.0003	0.0039±0.00042
	5	0.0598±0.0351	0.0689±0.0117	0.0711±0.0125	0.1156±0.0180	0.1116±0.0174	0.0968±0.0138	0.1003±0.0158	**0.0468±0.0052**
	10	**0.0760±0.0339**	0.1319±0.0188	0.1341±0.0245	0.1966±0.0325	0.1949±0.0324	0.1493±0.0255	0.1586±0.0278	0.0948±0.0136
	20	**0.1520±0.0697**	0.2400±0.0511	0.2650±0.0487	0.3323±0.0469	0.3351±0.0498	0.2464±0.0340	0.2769±0.0533	0.1769±0.0213
ZDT2 (30)	0	0.0039±0.0005	0.0038±0.0002	0.0038±0.0002	0.0038±0.0002	0.0063±0.0035	**0.0036±0.0002**	0.0038±0.0002	0.0042±0.0005
	5	**0.0429±0.0423**	0.0996±0.0174	0.1081±0.0234	0.1513±0.0292	0.1296±0.0245	0.1084±0.0172	0.1757±0.0359	0.0494±0.0063
	10	**0.0614±0.0380**	0.2036±0.0371	0.1942±0.0411	0.2833±0.0542	0.2563±0.0422	0.1916±0.0261	0.2694±0.0609	0.0971±0.0137
	20	**0.1386±0.1230**	0.4360±0.0726	0.4412±0.0833	0.5225±0.0937	0.5042±0.1078	0.3875±0.0553	0.4493±0.0892	0.2218±0.0387
ZDT3 (30)	0	0.0093±0.0017	0.0086±0.0008	0.0086±0.0008	0.0086±0.0008	0.0096±0.0009	**0.0064±0.0006**	0.0086±0.0023	0.0087±0.0011
	5	**0.0331±0.0104**	0.0413±0.0050	0.0393±0.0060	0.0553±0.0112	0.0595±0.0177	0.0499±0.0097	0.0782±0.0151	0.0463±0.0077
	10	**0.0558±0.0383**	0.0704±0.0204	0.0708±0.0209	0.1129±0.0404	0.1277±0.0471	0.0767±0.0245	0.0993±0.0250	0.0744±0.0211
	20	**0.1228±0.0688**	0.2375±0.0299	0.2299±0.0353	0.2739±0.0545	0.2709±0.0488	0.1289±0.0703	0.2348±0.0435	0.1764±0.0553
ZDT4 (10)	0	0.5130±1.6893	0.5332±0.2244	0.5332±0.2244	0.5169±0.1978	0.5777±0.1847	0.5493±0.2190	0.5169±0.1679	**0.0148±0.0034**
	5	**0.0047±0.0010**	0.4658±0.2169	0.5324±0.1717	0.5432±0.2086	0.5533±0.2268	0.5131±0.1883	0.5686±0.1650	0.0105±0.0028
	10	**0.0042±0.0014**	0.5942±0.2827	0.5405±0.1999	0.5857±0.1945	0.5538±0.2046	0.5826±0.2264	0.5927±0.1889	**0.0262±0.0106**
	20	**0.0066±0.0047**	0.6040±0.2204	0.6232±0.2372	0.6090±0.2147	0.6007±0.1768	0.5939±0.2307	0.6701±0.2649	0.4861±1.0196
ZDT6 (10)	0	0.0130±0.0010	0.0128±0.0010	0.0128±0.0010	0.0125±0.0011	0.0426±0.1194	0.3706±0.7834	0.0125±0.0011	**0.0101±0.0043**
	5	**0.0101±0.0027**	0.0623±0.1702	0.0717±0.1682	0.0252±0.0016	0.0250±0.0026	0.7933±0.1971	0.9842±0.1275	0.0635±0.1391
	10	**0.0081±0.0011**	0.2378±0.3061	0.2924±0.3158	0.3219±0.3582	0.6900±0.2574	1.0991±0.1726	1.1682±0.1895	0.6011±0.3988
	20	**0.0070±0.0021**	0.8045±0.4053	0.8379±0.5210	1.3716±0.2406	1.4351±0.2195	1.7154±0.2660	1.5529±0.1691	1.0745±0.3266
DTLZ1 (30)	0	**57.7013±95.42**	204.73±50.023	204.73±50.023	122.80±35.134	128.60±38.403	1066.2±347.89	122.80±35.134	486.70±138.86
	5	**238.91±77.411**	345.50±71.277	377.5384±70.179	698.10±215.22	800.40±176.85	1051.3±173.33	1484.4±95.267	1237.0±269.63
	10	**245.663±62.950**	399.83±97.6119	457.16±83.127	849.50±198.22	777.20±151.10	1004.8±203.57	1510.6±83.930	1298.0±179.37
	20	**275.41±472.80**	466.79±109.74	472.80±79.401	913.30±217.38	840.40±180.99	1034.9±178.07	1551.4±66.218	1249.4±253.14
DTLZ2 (30)	0	0.0664±0.0069	0.1044±0.0123	0.1044±0.0123	0.0629±0.0033	0.0757±0.0044	**0.0530±0.0068**	0.0629±0.0033	0.0777±0.0127
	5	0.1425±0.0237	0.2045±0.0345	0.2124±0.0416	0.1622±0.0393	0.1645±0.0641	**0.1029±0.0199**	0.4997±0.1680	0.2906±0.0586
	10	0.2224±0.0591	0.2968±0.0654	0.3112±0.0591	0.2663±0.0729	0.2397±0.0667	**0.1471±0.0230**	0.5134±0.1353	0.3949±0.1275
	20	0.4682±0.0775	0.4873±0.1003	0.4911±0.0804	0.5195±0.0969	0.6225±0.1999	**0.2709±0.0326**	0.6589±0.1325	0.6032±0.1373
DTLZ3 (30)	0	**94.1752±137.75**	224.99±54.689	224.99±54.689	128.50±35.946	266.00±252.70	1493.2±295.54	128.50±35.946	571.60±336.39
	5	**427.72±109.86**	541.58±95.543	554.49±145.25	651.00±251.07	1075.5±310.33	1111.0±171.45	1684.2±105.79	935.00±193.19
	10	**537.46±160.78**	566.62±130.31	653.08±112.36	608.90±177.56	942.40±236.77	1109.2±163.71	1700.4±120.75	898.20±179.99
	20	**515.01±156.10**	570.19±130.86	589.37±146.69	653.60±218.42	1071.4±307.64	1047.4±193.59	1644.9±106.64	809.2±339.23

between LREDA and PSO may affect the search ability in some noiseless circumstances (ZDT1, ZDT2, and ZDT3). However, the performance of the PLREDA was outstanding in noisy environments. Since GD measures the distance between the evolved Pareto front and the Pareto optimal front, the closer they are, the smaller the GD value. It was observed that all of the algorithms are able to obtain a set of solutions that is near to the Pareto optimal front in ZDT1, ZDT2, ZDT3, ZDT6, and DTLZ2. For the rest of the test problems (ZDT4, DTLZ1, and DTLZ3), most of the algorithms are trapped in local optima. This is attributed to the fact that three of these test problems consisted of many local optimal fronts. MOEARF was able to generate good solutions in ZDT4; however, its performance is poor in DTLZ1 and DTLZ3. PLREDA performed better than MOEARF in all DTLZ problems, even though the algorithm also failed to reach the Pareto optimal front.

The good performance of PLREDA was due to the combination of REDA, PSO, and the likelihood correction feature. First of all, REDA is more robust than NSGAII because the performance of REDA is better than NSGAII in noisy conditions. REDA, which performs the search by modeling the global probability distribution of the population, is more responsive since the reproduction is based on the global information and not individual solutions. Furthermore, likelihood correction is able to tune the probability distribution so that the distribution of the solutions is more likely to follow the one with a smaller selection error. The hybridization has further enhanced the ability of REDA in exploring the search space, especially in noisy conditions. This hybridization is utterly important when the REDA fails to model the promising regions in the search space. In that case, the hybridization could provide opportunities to explore those regions, thus improving the search ability. Resampling is probably one of the simplest noise handling mechanisms which has been implemented in RNSGAII. The performance of this algorithm is better than NSGAII in ZDT problems. However, its performance is poor in DTLZ problems. Even though resampling is able to reduce the effect of noise, it incurred extra fitness evaluations. Since the fitness evaluation in DTLZ problems is limited to 80,000, RNSGAII failed to converge at the end of the evolution. NTSPEA and MOPSEA, which are noise handling algorithms, have shown better results in noisy environments compared to NSGAII. This is unquestionable since NTSPEA and MOPSEA tackled the noisy information by incorporating the domination dependent lifetime and probability dominance respectively, while NSGAII did not take any extra measurement into account when handling the noisy data. For MOEARF, the noise handling mechanism is based on after the fact knowledge of the favorable movements; thus, it was able to obtain good results in ZDT problems. However, the performance of MOEARF in DTLZ problems is poorer than those of all other algorithms except RNSGAII.

Table 3 shows the results in terms of MS measurement obtained from the different algorithms. Generally, PLREDA was able to evolve a set of most diverse solutions in most of the test instances, followed by MOEARF and MOPSEA. The incorporation of likelihood correction was also able to improve the diversity of REDA in some of the test instances. There was no clear difference in performance between REDA and NSGAII. Furthermore, resampling was unable to maintain or improve the diversification of NSGAII. It was also observed that the noise interfered with the diversity performance of all algorithms in all ZDT problems. For DTLZ problems, the negative influence of noise toward the diversity preservation in all algorithms was far smaller than for ZDT problems.

Among the algorithms, PLREDA, LREDA, REDA, NSGAII, and RNSGAII maintained the diversity of the solutions through crowding measurement while NTSPEA, MOPSEA, and MOEARF kept the diversity of the solutions by using a niche sharing

Table 3: Comparison of PLREDA to the different MOEAs with respect to the MS performance indicator.

Problems	Noise levels	Algorithms							
(*n*)	(%)	PLREDA	LREDA	REDA	NSGAII	NTSPEA	MOPSEA	RNSGAII	MOEARF
ZDT1 (30)	0	**0.9982±0.0014**	0.9928±0.0016	0.9928±0.0016	0.9979±0.0007	0.9970±0.0007	0.9881±0.0045	0.9979±0.0009	0.9768±0.0144
	5	0.9406±0.0281	0.9263±0.0230	0.9251±0.0217	0.9133±0.0267	0.9116±0.0229	0.9162±0.0297	0.9264±0.0292	**0.9680±0.0161**
	10	0.9130±0.0399	0.8856±0.0358	0.8998±0.0303	0.8714±0.0445	0.8561±0.0362	0.8535±0.0546	0.8904±0.0308	**0.9380±0.0161**
	20	0.8731±0.0447	0.8319±0.0582	0.8297±0.0437	0.8128±0.0432	0.7693±0.0464	0.7894±0.0586	0.8399±0.0322	**0.8949±0.0432**
ZDT2 (30)	0	**0.9973±0.0023**	0.9877±0.0011	0.9877±0.0011	0.9934±0.0027	0.9901±0.0046	0.9817±0.0072	0.9934±0.0027	0.9812±0.0118
	5	0.9073±0.0451	0.8646±0.0301	0.8513±0.0277	0.8300±0.0408	0.8381±0.0395	0.8022±0.0370	0.7865±0.0449	**0.9565±0.0082**
	10	0.8629±0.0643	0.7517±0.0410	0.7478±0.0548	0.7318±0.0665	0.7301±0.0458	0.5340±0.1602	0.6632±0.1516	**0.9169±0.0187**
	20	0.7056±0.1237	0.4699±0.1898	0.4778±0.1370	0.5086±0.2445	0.3979±0.2498	0.1102±0.1428	0.4784±0.2083	**0.7412±0.2345**
ZDT3 (30)	0	0.9987±0.0022	0.9985±0.0007	0.9985±0.0007	**0.9998±0.0001**	0.9862±0.0240	0.9753±0.0415	**0.9998±0.0001**	0.9712±0.0537
	5	**0.9707±0.0158**	0.9602±0.0106	0.9625±0.0106	0.8973±0.0370	0.8871±0.0391	0.8901±0.0557	0.8916±0.0384	0.9418±0.0061
	10	**0.9566±0.0215**	0.9409±0.0136	0.9418±0.0131	0.8760±0.0196	0.8226±0.0668	0.7706±0.1480	0.8615±0.0824	0.9265±0.0138
	20	**0.9190±0.0550**	0.8654±0.0677	0.8821±0.0575	0.8117±0.0561	0.7517±0.0836	0.6446±0.1646	0.8284±0.0643	0.8775±0.0587
ZDT4 (10)	0	0.9302±0.1159	0.7774±0.0714	0.7774±0.0714	0.7545±0.0681	0.7352±0.0500	0.7176±0.0716	0. 7545±0.0681	**0.9544±0.0182**
	5	0.9372±0.0217	0.7569±0.0820	0.7339±0.0596	0.7469±0.0642	0.7394±0.0710	0.7328±0.0618	0.7358±0.0468	**0.9769±0.0175**
	10	0.9168±0.0221	0.7117±0.0665	0.7171±0.0707	0.7337±0.0626	0.7179±0.0571	0.6921±0.0720	0.7156±0.0494	**0.9648±0.0303**
	20	0.8575±0.0653	0.6865±0.0700	0.6762±0.0554	0.7084±0.0765	0.6599±0.0680	0.6344±0.0903	0.6921±0.0663	**0.8692±0.0944**
ZDT6 (10)	0	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	0.9894±0.0520	0.9223±0.1292	0.9232±0.1217	0.9963±0.0128
	5	**0.9993±0.0015**	0.9879±0.0535	0.9904±0.0485	0.9946±0.0013	0.9925±0.0039	0.6929±0.0371	0.6999±0.0006	0.9948±0.0005
	10	**0.9990±0.0017**	0.9336±0.1209	0.8976±0.1360	0.8769±0.1457	0.7289±0.0897	0.6925±0.0182	0.6967±0.0102	0.9947±0.0014
	20	**0.9982±0.0027**	0.7751±0.1258	0.7838±0.1319	0.6983±0.0048	0.6921±0.0204	0.6457±0.1701	0.6988±0.0027	0.9939±0.0030
DTLZ1 (30)	0	**0.9980±0.0037**	0.9944±0.0024	0.9944±0.0024	0.9959±0.0025	0.9956±0.0048	0.9816±0.0067	0.9959±0.0025	0.9716±0.0094
	5	**0.9927±0.0040**	0.9850±0.0087	0.9828±0.02	0.9880±0.0040	0.9850±0.0085	0.9688±0.0140	0.9701±0.0037	0.9579±0.0091
	10	**0.9905±0.0069**	0.9790±0.0168	0.9755±0.0167	0.9862±0.0041	0.9836±0.0097	0.9630±0.0173	0.9684±0.0042	0.9543±0.0114
	20	**0.9876±0.0080**	0.9688±0.0291	0.9743±0.0160	0.9830±0.0045	0.9794±0.0084	0.9554±0.0208	0.9668±0.0049	0.9529±0.0164
DTLZ2 (30)	0	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	0.9993±0.0008	0.9998±0.0009	**1.0000±0.0000**	**1.0000±0.0000**
	5	0.9989±0.0001	0.9986±0.0001	0.9985±0.0001	0.9995±0.0004	0.9990±0.0007	0.9985±0.0014	0.9997±0.0002	**0.9999±0.0002**
	10	0.9987±0.0008	0.9978±0.0014	0.9982±0.0010	0.9990±0.0008	0.9989±0.0008	0.9982±0.0018	0.9994±0.0005	**0.9998±0.0003**
	20	0.9978±0.0023	0.9977±0.0016	0.9979±0.0011	0.9983±0.0017	0.9983±0.0015	0.9980±0.0013	0.9993±0.0005	**0.9996±0.0003**
DTLZ3 (30)	0	**1.0000±0.0000**	0.9994±0.0013	0.9994±0.0013	**1.0000±0.0000**	0.9996±0.0018	0.9998±0.0003	**1.0000±0.0000**	0.9998±0.0011
	5	**1.0000±0.0000**	0.9790±0.0309	0.9835±0.0284	0.9994±0.0011	0.9977±0.0029	0.9981±0.0024	0.9989±0.0022	0.9979±0.0027
	10	**1.0000±0.0000**	0.9827±0.0215	0.9814±0.0195	0.9993±0.0008	0.9986±0.0015	0.9963±0.0036	0.9980±0.0048	0.9961±0.0049
	20	**1.0000±0.0000**	0.9743±0.0205	0.9656±0.0466	0.9971±0.0043	0.9979±0.0022	0.9933±0.0061	0.9973±0.0055	0.9961±0.0038

mechanism. There was not much difference between the diversity preservation mechanisms in both noisy and noiseless environments. In fact, both of the methods were able to maintain a set of diverse solutions even though the convergence was not good, which is indicated by the MS values being near to 1.0. In DTLZ problems, no matter how far the evolved solutions are from optimality, all algorithms were successful in maintaining a set of diverse solutions (MS > 0.9). This may be due to the fact that the number of nondominated solutions in DTLZ problems is much more than those in ZDT problems since DTLZ problems consist of three objective functions, while ZDT problems only have two objective functions. In addition, it may also be caused by the setting of the population size.

Table 4 shows the results in terms of IGD measurement obtained from the different algorithms. PLREDA showed the best performance in most of the test instances, followed by MOEARF. The performance of REDA is improved in most of the test instances when likelihood correction is incorporated (LREDA). Comparing REDA and NSGAII, algorithms without any noise handling feature, REDA is more robust than NSGAII. This is shown by a lower value of IGD in REDA. However, the robustness of the REDA is not due to the type of Gaussian noise implemented in this paper (i.e., Gaussian noise with zero mean). This is because REDA does not take any average information from the population. Instead, it measures the probability of existence of any cardinality in the decision variables. As with the observation in Table 2, resampling in RNSGAII has improved the IGD value of NSGAII in ZDT problems but the performance is poor in DTLZ problems. For MOEARF, the convergence and diversity maintenance were good in ZDT problems; however, the performance was poor in DTLZ problems.

Figure 3 shows the Pareto front of ZDT3 generated from the different algorithms. The solutions plotted are the nondominated solutions obtained from all 30 simulation runs. It is clear that all of the algorithms were able to maintain a set of good diverse solutions. In terms of convergence, PLREDA and MOEARF have an evolved Pareto front which is closed to the Pareto optimal front. Figure 4 shows the Pareto front of DTLZ1 generated from the different algorithms. It could be seen that all of the algorithms failed to converge to global optimality. As a comparison, the results generated from PLREDA were nearer to the global Pareto optimal front. Furthermore, it was observed that REDA, LREDA, and PLREDA have many more nondominated solutions in their final front than the other five algorithms. Among them, PLREDA had more nondominated solutions, followed by LREDA and REDA.

5.2 Scalability Analysis

In this section, the scalability behavior of the different algorithms was tested. This is done by spanning the number of decision variables from 30 to 100. Figure 5 shows the behavior of the algorithms in test problem ZDT1, with different number of decision variables measured with IGD indicator under (a) 0% and (b) 20% noise levels. From Figure 5(a), it is observed that the performance of all of the algorithms is decreased when the number of decision variables is increased. This was expected since the difficulty and complexity of the problem was raised with the increase in the number of decision variables. MOPSEA and PLREDA clearly outperformed the other algorithms. MOEARF is the most robust toward the increase in the number of decision variables as shown by the similar IGD values, while the performance of NTSPEA was the poorest. At the 20% noise level, PLREDA was able to maintain good performance, thus outperforming the other algorithms. The second best performance was obtained by MOEARF. Other than that, the performance of the other algorithms was very much affected by the increase in the number of decision variables.

Table 4: Comparison of PLREDA to the different MOEAs with respect to the IGD performance indicator.

Problems	Noise levels	Algorithms							
(*n*)	(%)	PLREDA	LREDA	REDA	NSGAII	NTSPEA	MOPSEA	RNSGAII	MOEARF
ZDT1 (30)	0%	0.0054±0.0015	**0.0042±0.0002**	**0.0042±0.0002**	0.0047±0.0002	0.0073±0.0006	0.0044±0.0004	0.0047±0.0002	0.0098±0.0016
	5%	0.0547±0.0288	0.0633±0.0099	0.0659±0.0109	0.1101±0.0158	0.1092±0.0165	0.0966±0.0125	0.0946±0.0155	**0.0460±0.0053**
	10%	**0.0684±0.0278**	0.1139±0.0138	0.1128±0.0212	0.1823±0.0324	0.1912±0.0275	0.1544±0.0264	0.1475±0.0261	0.0841±0.0110
	20%	**0.1178±0.0512**	0.1960±0.0421	0.2087±0.0392	0.2894±0.0389	0.3239±0.0461	0.2486±0.0314	0.2448±0.0473	0.1492±0.0212
ZDT2 (30)	0%	0.0048±0.0004	**0.0043±0.0001**	**0.0043±0.0001**	0.0047±0.0003	0.0089±0.0034	0.0048±0.0003	0.0047±0.0003	0.0098±0.0012
	5%	0.0502±0.0349	0.0934±0.0168	0.1009±0.0214	0.1529±0.0322	0.1367±0.0256	0.1295±0.0195	0.1800±0.0372	**0.0500±0.0051**
	10%	**0.0724±0.0341**	0.1882±0.0338	0.1834±0.0419	0.2740±0.0633	0.2688±0.0421	0.3040±0.1019	0.3011±0.1052	0.0877±0.0117
	20%	**0.1680±0.1202**	0.4973±0.1511	0.4808±0.1330	0.6573±0.2060	0.6436±0.2214	0.7329±0.1145	0.5454±0.2075	0.2237±0.1476
ZDT3 (30)	0%	0.0062±0.0030	0.0058±0.0007	0.0058±0.0007	**0.0053±0.0004**	0.0104±0.0055	0.0128±0.0072	**0.0053±0.0004**	0.0154±0.0078
	5%	**0.0423±0.0160**	0.0769±0.0096	0.0744±0.0150	0.1097±0.0162	0.1225±0.0219	0.1028±0.0149	0.1089±0.0154	0.0571±0.0085
	10%	**0.0943±0.0408**	0.1225±0.0197	0.1245±0.0164	0.1739±0.0216	0.2048±0.0260	0.1884±0.0433	0.1580±0.0333	0.1027±0.0169
	20%	**0.1575±0.0533**	0.2299±0.0332	0.2222±0.0324	0.3012±0.0552	0.3510±0.0534	0.3287±0.0585	0.2333±0.0389	0.1838±0.0260
ZDT4 (10)	0%	0.3898±0.9901	0.5403±0.2386	0.5403±0.2386	0.5060±0.2019	0.5663±0.1926	0.5405±0.2256	0.5853±0.1753	**0.0108±0.0018**
	5%	**0.0157±0.0042**	0.4714±0.2358	0.5346±0.1796	0.5509±0.2257	0.5757±0.2513	0.5241±0.2027	0.5748±0.1791	0.0192±0.0029
	10%	**0.0291±0.0088**	0.5541±0.2277	0.5382±0.2145	0.5870±0.2142	0.5680±0.2157	0.6020±0.2399	0.5994±0.2057	0.0332±0.0070
	20%	**0.0581±0.0205**	0.5746±0.2268	0.5845±0.2193	0.6021±0.2351	0.6097±0.1778	0.6358±0.2493	0.6584±0.2521	0.1330±0.0884
ZDT6 (10)	0%	**0.0027±0.0003**	**0.0027±0.0003**	**0.0027±0.0003**	0.0029±0.0003	0.0270±0.1246	0.1775±0.3085	0.0029±0.3138	0.0134±0.0098
	5%	0.0161±0.0045	0.0451±0.1344	0.0391±0.1046	0.0104±0.0026	0.0200±0.0070	1.0768±0.2711	1.3054±0.1382	**0.0117±0.0025**
	10%	**0.0229±0.0096**	0.1984±0.3230	0.2901±0.3714	0.3722±0.4578	0.8744±0.3446	1.4113±0.2081	1.4786±0.2097	0.0242±0.0087
	20%	**0.0365±0.0154**	0.8468±0.5366	0.8846±0.5840	1.6392±0.2960	1.7359±0.2337	2.0257±0.2888	1.8646±0.1986	0.0499±0.0236
DTLZ1 (30)	0%	**35.396±58.886**	143.91±35.028	143.91±35.028	88.702±24.354	81.192±19.633	299.93±116.66	88.702±24.354	286.56±71.423
	5%	**140.92±47.737**	206.25±41.229	222.20±43.111	285.35±87.155	266.32±67.300	413.02±99.206	810.45±74.198	535.86±111.55
	10%	**135.14±34.452**	233.78±48.610	267.82±45.780	331.05±100.50	256.62±59.640	424.27±115.08	830.09±82.451	566.59±91.068
	20%	**147.18±35.460**	259.58±51.036	263.98±50.793	363.79±104.99	277.40±65.426	468.61±114.92	841.14±81.941	555.32±109.69
DTLZ2 (30)	0%	0.0590±0.0075	0.0967±0.0129	0.0967±0.0129	**0.0549±0.0031**	0.0679±0.0049	0.0741±0.0056	0.2656±0.0815	0.0829±0.0138
	5%	0.1302±0.0204	0.1809±0.0273	0.1859±0.0325	0.1329±0.0238	**0.1024±0.0142**	0.1400±0.0153	0.3343±0.0933	0.2369±0.0505
	10%	0.2008±0.0291	0.2469±0.0436	0.2572±0.0435	0.2178±0.0467	**0.1389±0.0155**	0.1898±0.0234	0.3531±0.0869	0.3027±0.0783
	20%	0.3556±0.0446	0.3748±0.0672	0.3802±0.0517	0.4059±0.0807	0.3271±0.0811	**0.2824±0.0338**	0.4495±0.0951	0.4318±0.0797
DTLZ3 (30)	0%	**79.2636±113.35**	192.88±43.167	192.88±43.167	123.32±34.176	172.20±88.578	655.55±227.521	822.56±166.19	370.93±77.204
	5%	**222.60±59.115**	399.92±56.994	376.40±66.955	326.11±114.48	393.38±116.36	635.81±137.61	986.85±107.06	498.70±82.094
	10%	**235.55±62.717**	388.10±84.239	396.61±57.952	279.47±61.408	365.94±89.696	678.54±98.491	953.77±134.41	511.29±90.213
	20%	**218.48±72.925**	395.49±79.214	401.35±80.442	295.52±72.263	398.62±118.45	650.39±132.55	988.71±132.23	438.92±162.49

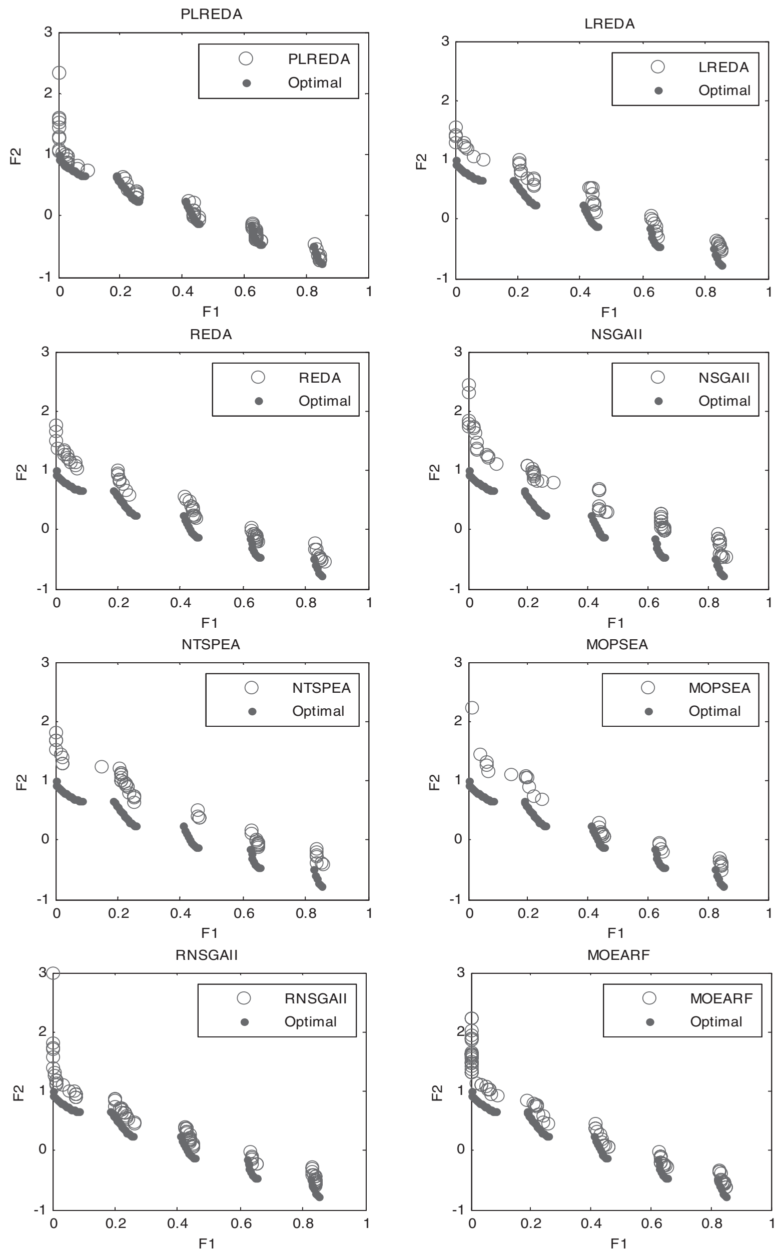


Figure 3: Pareto front of ZDT3 generated from the different algorithms.

Figure 6 shows the scalability behavior of the algorithms in test problem DTLZ1 with different numbers of decision variables measured with the IGD indicator under (a) 0% and (b) 20% noise levels. The results observed in this problem were quite different from the previous one. At the 0% noise level, the performance of NTSPEA was the best compared to other algorithms, while MOEARF showed the worst performance. PLREDA performed better than other algorithms except for NTSPEA. Under

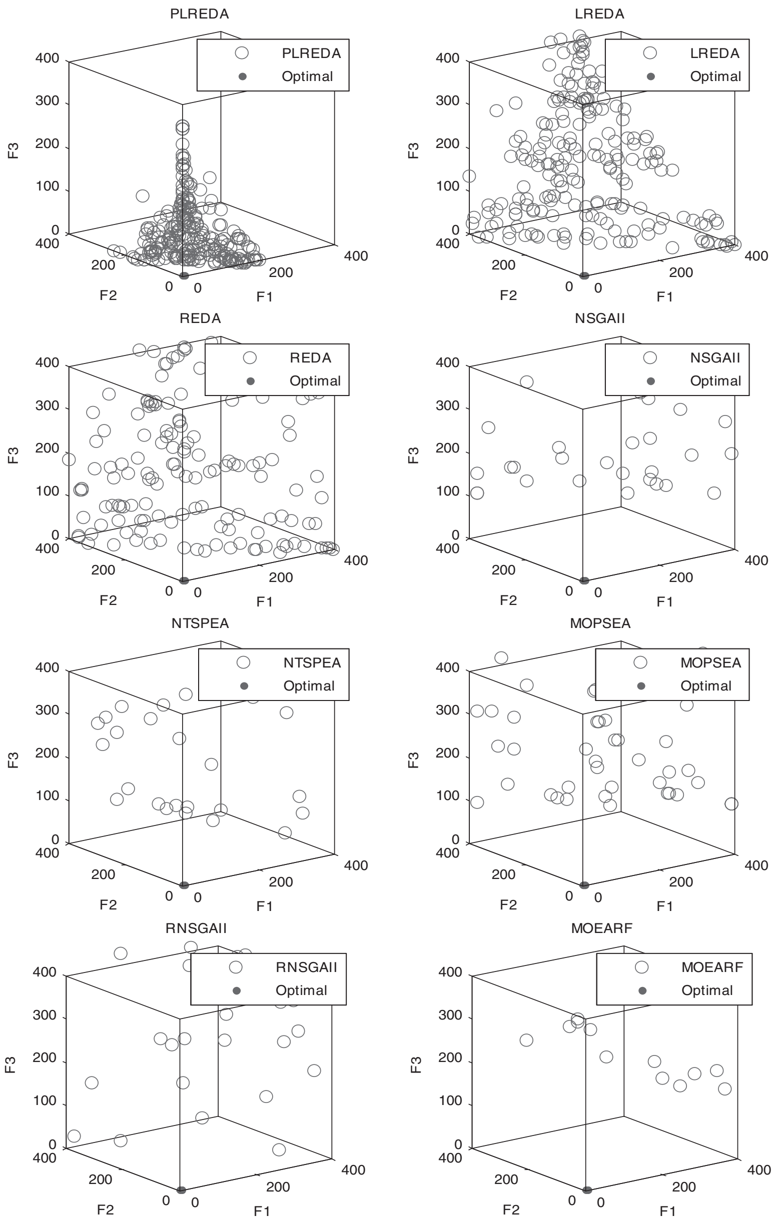
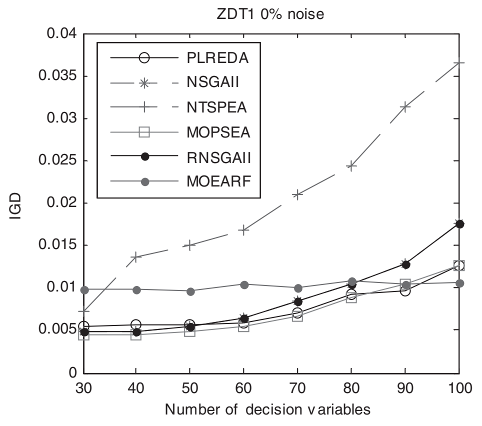


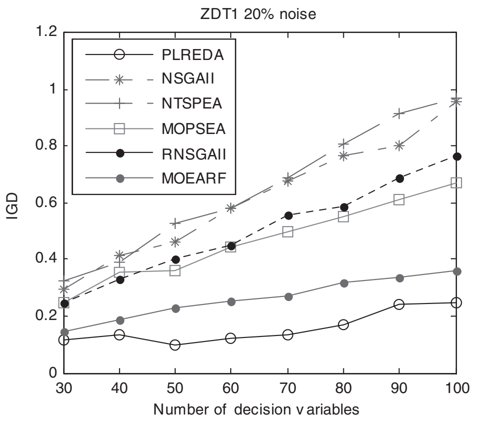
Figure 4: Pareto front of DTLZ1 generated from the different algorithms.

the influence of the 20% noise level, PLREDA outperformed NTSPEA and dominated the overall performance. Even though MOEARF achieved good IGD in ZDT1, its performance was poor in DTLZ1. The worst result was produced by RNSGAII.

Overall, PLREDA obtained the most promising results in both of the test problems with different numbers of decision variables. This may be due to the learning capabilities of the RBM. In PLREDA, the probabilistic model was built from the RBM network. The network learned the multivariate dependencies among the decision variables and

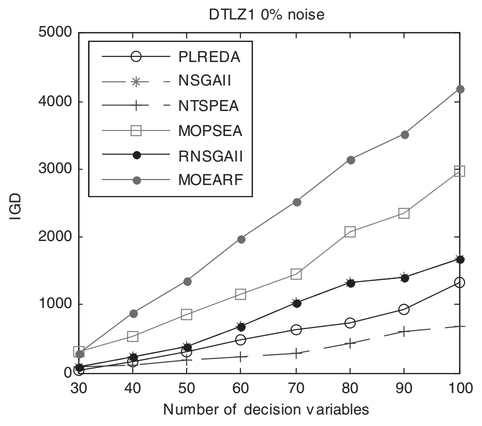


(a)

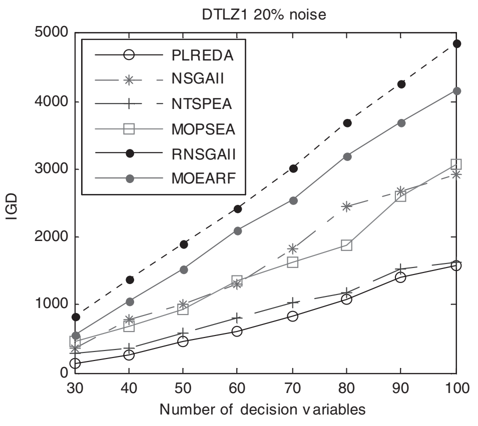


(b)

Figure 5: Performance metric of IGD versus the number of decision variables in test problem ZDT1 under (a) 0% and (b) 20% noise levels.



(a)



(b)

Figure 6: Performance metric of IGD versus the number of decision variables in test problem DTLZ1 under (a) 0% and (b) 20% noise levels.

returns the information in terms of energy stability. This characteristic has the potential to estimate the global probabilistic distribution to guide the search during the evolution process. In addition, the hybridization with PSO has enhanced the search ability of REDA. This is important as PSO provides a directional search which may explore promising regions where the probabilistic model may fail to explore. In conclusion, PLREDA scaled well with the number of decision variables compared to the other algorithms.

5.3 Possibility of Other Hybridizations

In this section, the possibility of other hybridizations with LREDA is investigated. The proposed algorithm, PLREDA, is the hybridization between LREDA and PSO. In order

Table 5: Performance metric of IGD obtained from the different hybridizations.

Problem	Noise level	PSO	GA	DE	LREDA
ZDT1 (30)	0%	0.0054±0.0015	0.0047±0.0003	0.0046±0.0003	0.0042±0.0002
	20%	**0.1178±0.0512**	0.2178±0.0342	**0.1552±0.0206**	0.1960±0.0421
ZDT2 (30)	0%	0.0048±0.0004	0.0046±0.0002	0.0046±0.0001	0.0043±0.0001
	20%	**0.1680±0.1202**	**0.3572±0.0751**	**0.3392±0.2217**	0.4973±0.1511
ZDT3 (30)	0%	0.0062±0.0030	**0.0055±0.0004**	**0.0054±0.0003**	0.0058±0.0007
	20%	**0.1575±0.0533**	0.2244±0.0178	**0.1790±0.0532**	0.2299±0.0332
ZDT4 (10)	0%	**0.3898±0.9901**	**0.4872±0.1955**	**0.4366±0.1898**	0.5403±0.2386
	20%	**0.0581±0.0205**	**0.5409±0.2948**	**0.4369±0.1608**	0.5746±0.2268
ZDT6 (10)	0%	**0.0027±0.0003**	**0.0026±0.0004**	0.0031±0.0005	0.0027±0.0003
	20%	**0.0365±0.0154**	**0.6202±0.5297**	1.3922±0.3021	0.8468±0.5366
DTLZ1 (30)	0%	**35.396±58.886**	**89.579±23.4504**	**76.113±21.804**	143.91±35.028
	20%	**147.18±35.460**	**202.77±34.3597**	**190.10±35.433**	259.58±51.036
DTLZ2 (30)	0%	**0.0590±0.0075**	**0.0739±0.0080**	**0.0724±0.0081**	0.0967±0.0129
	20%	**0.3556±0.0446**	0.4113±0.0440	**0.3321±0.0441**	0.3748±0.0672
DTLZ3 (30)	0%	**79.264±113.35**	**129.86±23.880**	**123.01±22.637**	192.88±43.167
	20%	**218.48±72.925**	**287.64±79.8410**	**244.08±76.056**	395.49±79.214

to study the potential of other hybridizations, LREDA is hybridized with a GA and a differential evolution (DE). The simplest and most common GA is applied, where single point crossover and bit-flip mutation are implemented. For DE, the standard recombination proposed in Storn and Price (1997) is applied. Table 5 shows the IGD values obtained from the different hybridizations. LREDA is the algorithm without any hybridization. The results are highlighted in bold if the hybridization shows improvement compared to the original algorithm (LREDA). From the charts, it is clear that all hybridizations are able to improve the performance of LREDA, in most of the test problems, under both noiseless and 20% noise level. Among them, hybridization with PSO gave the best results followed by DE and then GA. The function of hybridization is to provide extra search ability for LREDA as LREDA performs the search by using only global statistical information. This hybridization therefore enhances the ability for LREDA in exploring the search space, especially in the early stage of evolutions where the search space is huge. The search using position information (GA, PSO, DE) is also essential and useful especially to explore and exploit certain promising regions. Thus, hybridization is an important mechanism to improve the search performance of EDAs.

5.4 Computational Time Analysis

The incorporation of any noise handling feature or enhancement operator will incur additional computational cost. In this section, the CPU times of the different algorithms are studied. Table 6 shows the CPU time (s) used by the different algorithms to complete a single simulation run under 0% noise level. It was observed that REDA and LREDA were the most time-consuming algorithms, followed by PLREDA and MOEARF. The rest of the algorithms took less computational time. The CPU time used by the different algorithms to complete a single simulation run under the 20% noise level is tabulated in Table 7. The CPU times used by REDA, LREDA, PLREDA, and NSGAII were not very different from that in the noiseless circumstance, while there was much less computation needed for the other algorithms. REDA was the most time-consuming algorithm. This is due to the training and modeling part in RBM. In REDA, training is conducted at each generation and stops when the number of training epochs is reached. This training process is more complicated than the genetic operators in the other MOEAs,

Table 6: CPU time(s) used by the different algorithms to complete a single simulation run in the different test problems under the 0% noise level.

Problems	Algorithm							
(n)	PLREDA	LREDA	REDA	NSGAII	NTSPEA	MOPSEA	RNSGAII	MOEARF
ZDT1 (30)	122.51±1.0389	231.89±0.7859	234.14±4.3558	2.6207±0.1078	11.011±0.5287	12.266±0.8743	2.6207±0.1078	126.71±7.5289
ZDT4 (10)	43.499±0.3696	75.761±0.3535	78.111±1.2185	1.4713±0.1424	9.9386±0.5988	12.564±0.6009	1.4713±0.1424	74.708±6.8302
DTLZ1 (30)	92.536±0.1147	176.25±2.1791	173.56±1.6536	6.0777±0.0685	9.7985±1.0396	46.804±1.1997	6.0777±0.0685	77.333±6.3065

Table 7: CPU time(s) used by the different algorithms to complete a single simulation run in the different test problems under the 20% noise level.

Problems	Algorithm							
(n)	PLREDA	LREDA	REDA	NSGAII	NTSPEA	MOPSEA	RNSGAII	MOEARF
ZDT1 (30)	120.70±0.8073	222.81±0.4885	228.62±8.8671	2.1863±0.0748	1.7295±0.1511	4.0765±0.1056	0.7892±0.1476	23.493±0.2349
ZDT4 (10)	43.114±0.2843	76.617±0.1805	76.326±0.2052	0.9991±0.1401	0.8452±0.1021	3.8687±0.1763	0.4004±0.0485	4.0263±0.1019
DTLZ1 (30)	91.133±0.4819	175.95±0.5336	172.70±1.0873	5.7052±0.1372	12.662±0.5601	65.694±1.3063	2.03±0.0192	93.594±6.8869

thus incurring extra computational time. The incorporation of likelihood correction into REDA increased the CPU time of LREDA slightly; however, the extra time incurred was less than the training time in REDA. As for PLREDA, the computation of PSO was fast. Therefore, PLREDA took less CPU time than LREDA and REDA. The noise handling mechanism in NTSPEA, MOPSEA, and MOEARF measured certain information (e.g., domination behavior among all solutions) in order to cope with noisy information. In noisy conditions, the number of nondominated solutions was fewer than in noiseless situations. Thus, NTSPEA, MOPSEA, and MOEARF took less computational time in noisy environments than in noiseless circumstances. Even though PLREDA took more CPU time than other MOEAs with genetic operators, the time taken to perform a single simulation run was acceptable. This is because most of the real-world optimization cost functions are very time-consuming; for example, a few minutes is required for a single fitness evaluation.

6 Conclusions

This research studied the potential of an MOEDA in dealing with noisy environments. REDA, one of the recently developed MOEDAs, has been used to tackle the noisy information. The presence of noise in the objective functions may cause weaker individuals to be selected to the next generation. Therefore, a likelihood correction scheme, which utilizes the information of probability error in selection, was proposed. This feature adjusts the independent marginal distribution in each decision variable according to the heuristic that the true probability distribution of the solutions is more likely to follow the distribution of the individuals with a smaller probability selection error. The REDA, which uses only global information in guiding the search, has limitations in exploring certain promising search spaces. In order to improve the search ability, a particle swarm optimization algorithm is hybridized with REDA. The empirical results showed that the proposed algorithm outperformed most of the algorithms in most of the test instances and has better scalability. Finally, the hybridization between EDAs and other evolutionary paradigms, particularly GA and DE, has been shown to improve the search performance.

References

Azevedo, F., Vale, Z. A., Oliveira, P., and Khodr, H. (2010). A long-term risk management tool for electricity markets using swarm intelligence. *Electric Power Systems Research*, 80(4):380–389.

Bäck, T., Hammel, U., and Schwefel, H. P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17.

Basseur, M., Zitzler, E., and Talbi, E. (2006). A preliminary study on handling uncertainty in indicator-based multiobjective optimization. In *EvoWorkshops*, pp. 727–739.

Beyer, H. G. (2000). Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. *Computer Methods in Applied Mechanics and Engineering*, 186(2):239–267.

Boonma, P., and Suzuki, J. (2009). A confidence-based dominance operator in evolutionary algorithms for noisy multiobjective optimization problems. In *IEEE International Conference on Tools with Artificial Intelligence*, pp. 387–394.

Bosman, P. A. N., and Thierens, D. (2002). Multi-objective optimization with diversity preserving mixture-based iterated density estimation evolutionary algorithms. *International Journal of Approximate Reasoning*, 31(3):259–289.

Büche, D., Stoll, P., Dornberger, R., and Koumoutsakos, P. (2002). Multiobjective evolutionary algorithm for the optimization of noisy combustion processes. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 32(4):460–473.

Bui, L. T., Abbass, H. A., and Essam, D. (2005). Inheritance for noisy evolutionary multi-objective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 779–785.

Bui, L. T., Abbass, H. A., and Essam, D. (2009). Localization for solving noisy multi-objective optimization problems. *Evolutionary Computation*, 17(3):379–409.

Cebrian, M., Alfonseca, M., and Ortega, A. (2009). Towards the validation of plagiarism detection tools by means of grammar evolution. *IEEE Transactions on Evolutionary Computation*, 13(3):477–485.

Cedeno, W., and Agrafiotis, D. K. (2002). Application of niching particle swarms to QSAR and QSPR. In *Proceedings of the Fourteenth European Symposium on QSAR*, pp. 8–13.

Chakraborty, M., and Chakraborty, U. (1997). An analysis of linear ranking and binary tournament selection in genetic algorithms. In *Proceedings of the 1st International Conference on Information Communications and Signal Processing*, pp. 407–411.

Chen, Y., and Chen, C. (2010). Enabling the compact genetic algorithm for real-parameter optimization by using adaptive discretization. *Evolutionary Computation*, 18(2):199–228.

Coello Coello, C. A., Van Veldhuizen, D. A., and Lamont, G. B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Norwell, MA: Kluwer Academic Publishers.

Costa, M., and Minisci, E. (2003). Moped: A multi-objective Parzen-based estimation of distribution algorithm for continuous problems. In *Evolutionary Multi-Criterion Optimization. Second International Conference*, pp. 282–294.

Damas, S., Cordón, O., and Santamaría, J. (2011). Medical image registration using evolutionary computation: An experimental survey. *IEEE Computational Intelligence Magazine*, 6(4):26–42.

Darwen, P., and Pollack, J. (1999). Coevolutionary learning on noisy tasks. In *IEEE Congress on Evolutionary Computation*, pp. 1724–1731.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. New York: John Wiley.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002). Scalable multi-objective optimization test problems. In *Proceedings of the Congress on Evolutionary Computation*, pp. 825–830.

Engelbrecht, A. P. (2006). *Fundamentals of computational swarm intelligence*. New York: John Wiley.

Eskandari, H., and Geiger, C. D. (2009). Evolutionary multiobjective optimization in noisy problem environments. *Journal of Heuristics*, 15(6):559–595.

Fonseca, C. M., and Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16.

Goh, C. K., and Tan, K. C. (2007). An investigation on noisy environments in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 11(3):354–381.

Goh, C. K., Tan, K. C., Liu, D. S., and Chiam, S. C. (2010). A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design. *European Journal of Operational Research*, 202(1):42–54.

Hinton, G. E. (2002). Training products of experts by minimizing constrative divergence. *Neural Computation*, 14(8):1771–1800.

Hong, Y., Ren, Q., and Zeng, J. (2005). Optimization of noisy fitness functions with univariate marginal distribution algorithm. In *IEEE Congress on Evolutionary Computation*, pp. 1410–1417.

Hong, Y., Ren, Q., Zeng, J., and Chang, Y. (2005). Convergence of estimation of distribution algorithms in optimization of additively noisy fitness functions. In *IEEE International Conference on Tools with Artificial Intelligence*, pp. 219–223.

Hughes, E. J. (2001). Evolutionary multi-objective ranking with uncertainty and noise. In *Evolutionary Multi-Criterion Optimization, First International Conference*, pp. 329–343.

Iqbal, M., Freitas, A. A., and Johnson, C. G. (2008). Protein interaction inference using particle swarm optimization algorithm. In *6th European Conference on Evolutionary Computation, Machine Learning, and Data Mining in Bioinformatics*, pp. 61–70.

Kennedy, J., and Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation*, pp. 4101–4108.

Larrañaga, P., and Lozano, J. A. (2001). *Estimation of distribution algorithms. A new tool for evolutionary computation*. Norwell, MA: Kluwer Academic Publishers.

Laumanns, M., and Ocenasek, J. (2002). Bayesian optimization algorithms for multi-objective optimization. In *Parallel Problem Solving from Nature, PPSN VII. 7th International Conference*, pp. 298–307.

Li, H., Zhang, Q., Tsang, E., and Ford, J. A. (2004). Hybrid estimation of distribution algorithm for multiobjective knapsack problem. In *Evolutionary Computation in Combinatorial Optimization, 4th European Conference*, pp. 145–154.

Limbourg, P. (2005). Multi-objective optimization of problems with epistemic uncertainty. In *Evolutionary Multi-Criterion Optimization, Third International Conference, EMO*, pp. 413–427.

Lozano, J. A., Larrañaga, P., and Bengoetxea, E. (2006). *Towards a new evolutionary computation: Advances on estimation of distribution algorithms, Studies in fuzziness and soft computing*. Berlin: Springer-Verlag.

Marti, L., Garcia, J., Berlanga, A., and Molina, J. M. (2009). Solving complex high-dimensional problems with the multi-objective neural estimation of distribution algorithm. In *11th Annual Genetic and Evolutionary Computation Conference*, pp. 619–626.

Miller, B. L., and Goldberg, D. E. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9(3):193–212.

Mühlenbein, H., and Paass, G. (1996). From recombination of genes to the estimation of distributions, I. Binary parameters. In *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, pp. 178–187.

Neri, F., and Mininno, E. (2010). Memetic compact differential evolution for Cartesian robot control. *IEEE Computational Intelligence Magazine*, 5(2):54–65.

Okabe, T., Jin, Y., Sendhoff, B., and Olhofer, M. (2004). Voronoi-based estimation of distribution algorithm for multi-objective optimization. In *Proceedings of the Congress on Evolutionary Computation*, pp. 1594–1601.

Ong, Y. S., Lim, M., and Chen, X. S. (2010). Memetic computation; Past, present and future. *IEEE Computational Intelligence Magazine*, 5(2):24–31.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2005). Multiobjective HBOA, clustering, and scalability. In *Genetic and Evolutionary Computation Conference*, pp. 663–670.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2006). Multiobjective estimation of distribution algorithm. *Scalable Optimization via Probabilistic Modeling*, 33:223–248.

Robinson, J., and Rahmat-Samii, Y. (2004). Particle swarm optimization in electromagnetics. *IEEE Transactions on Antennas and Propagation*, 52(2):397–407.

Shim, V. A., Tan, K. C., and Chia, J. Y. (2010). An investigation on sampling technique for multi-objective restricted Boltzmann machine. In *IEEE Congress on Evolutionary Computation*, pp. 1081–1088.

Storn, R., and Price, K. (1997). Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

Syberfeldt, A., Ng, A., John, R., and Moore, P. (2010). Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling. *European Journal of Operational Research*, 204(3):533–544.

Tan, K. C., Chiam, S. C., Mamun, A. A., and Goh, C. K. (2009). Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *European Journal of Operational Research*, 197(2):701–713.

Tan, K. C., Khor, E. F., and Lee, T. H. (2005). *Multiobjective evolutionary algorithms and applications*. Berlin: Springer-Verlag.

Tang, H. J., Shim, V. A., Tan, K. C., and Chia, J. Y. (2010). Restricted Boltzmann machine based algorithm for multi-objective optimization. In *IEEE Congress on Evolutionary Computation*, pp. 3958–3965.

Teich, J. (2001). Pareto-front exploration with uncertain objective. *Evolutionary Multi-Criterion Optimization, First International Conference*, pp. 314–328.

Ting, C. K., Zeng, W. M., and Lin, T. C. (2010). Linkage discovery through data mining. *IEEE Computational Intelligence Magazine*, 5(1):10–13.

Tomassini, M. (1996). Evolutionary algorithms. *Proceedings of the International Workshop Towards Evolvable Hardware*, pp. 19–47.

Veldhuizen, D. A., and Lamont, G. B. (1998). Evolutionary computation and convergence to a Pareto front. *Late Breaking Papers at the Genetic Programming Conference*, pp. 221–228.

Yogev, O., Shapiro, A. A., and Antonsson, E. K. (2010). Computational evolutionary embryogeny. *IEEE Transactions on Evolutionary Computation*, 14(2):301–325.

Zhang, J., Zhan, Z., Lin, Y., Chen, N., Gong, Y., Zhong, J., Chung, H., Li, Y., and Shi, Y. (2011). Evolutionary computation meets machine learning: A survey. *IEEE Computational Intelligence Magazine*, 6(4):68–75.

Zhang, Q., Zhou, A., and Jin, Y. (2008). RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1):41–63.

Zhong, X., and Li, W. (2007). A decision-tree-based multi-objective estimation of distribution algorithm. In *International Conference on Computational Intelligence and Security*, pp. 114–118.

Zhu, Z. X., Jia, S., and Ji., Z. (2010). Towards a memetic feature selection paradigm. *IEEE Computational Intelligence Magazine*, 5(2):41–53.

Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195.