Information Sciences 634 (2023) 650–664

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# Dynamic multi-objective evolutionary algorithms in noisy environments

Shaaban Sahmoud [a], Haluk Rahmi Topcuoglu [b,*]

[a] Fatih Sultan Mehmet Vakif University, Computer Engineering Department, Istanbul, Turkey
[b] Marmara University, Computer Engineering Department, Faculty of Engineering, Istanbul, Turkey

ARTICLE INFO

*Keywords:*
Change detection
Dynamic multi-objective optimization problems
Noise detection
Noisy optimization problems
Uncertainty

ABSTRACT

Real-world multi-objective optimization problems encounter different types of uncertainty that may affect the quality of solutions. One common type is the stochastic noise that contaminates the objective functions. Another type of uncertainty is the different forms of dynamism including changes in the objective functions. Although related work in the literature targets only a single type, in this paper, we study Dynamic Multi-objective Optimization problems (DMOPs) contaminated with stochastic noises by dealing with the two types of uncertainty simultaneously. In such problems, handling uncertainty becomes a critical issue since the evolutionary process should be able to distinguish between changes that come from noise and real environmental changes that resulted from different forms of dynamism. To study both noisy and dynamic environments, we propose a flexible mechanism to incorporate noise into the DMOPs. Two novel techniques called Multi-Sensor Detection Mechanism (MSD) and Welford-Based Detection Mechanism (WBD) are proposed to differentiate between real change points and noise points. The proposed techniques are incorporated into a set of Dynamic Multi-objective Evolutionary Algorithms (DMOEAs) to analyze their impact. Our empirical study reveals the effectiveness of the proposed techniques for isolating noise from real dynamic changes and diminishing the noise effect on performance.

## 1. Introduction

Evolutionary Algorithms (EAs) demonstrate promising achievements on optimization problems in various fields including signal processing, pattern recognition, cloud computing, social networks, games, and healthcare [1–3]. A large number of state-of-art Multi-objective Evolutionary Algorithms (MOEAs) are proposed for optimization problems that have multiple conflicting objectives, called Multi-objective Optimization Problems (MOPs) [4].

On the other hand, some of the optimization problems in the literature demonstrate the existence of dynamism in various forms [5] and suffer from the existence of one or more types of uncertainty [6]. According to Jin and Branke [7], uncertainty or dynamism of optimization problems can be classified into four types depending on the source of uncertainty:

- The uncertainty arises in decision space in design parameters or environmental variables.
- The uncertainty arises in objective space as noise.

* Corresponding author.
*E-mail addresses:* ssahmoud@fsm.edu.tr (S. Sahmoud), haluk@marmara.edu.tr (H.R. Topcuoglu).

https://doi.org/10.1016/j.ins.2023.03.094
Received 30 October 2022; Received in revised form 12 March 2023; Accepted 17 March 2023

Available online 21 March 2023
0020-0255/© 2023 Elsevier Inc. All rights reserved.

- The uncertainty arises in objective space due to the existence of one or more error-prone objective functions.
- The uncertainty arises due to one or more objective function(s) changes over time.

Although there are many studies that consider these types of uncertainty in the literature, the majority of the related work concentrates only on a single type by assuming that only one type of uncertainty can exist in an optimization problem. Therefore, most of the MOEAs proposed in the literature are designed to solve optimization problems with one type of uncertainty: either the problems that have various forms of dynamism or the ones that are contaminated with noise [7,8]. However, many real-world optimization problems have different sources of uncertainties [7], where two different types of uncertainties are overlapped at the same time. One example is the adaption process of the controller for a dynamic process, where the sensors that are used to obtain readings may return noisy values from the process plant [5].

To fill this research gap, in this study we focus on dynamic multi-objective optimization problems where the objective surfaces are contaminated with noise. Since Multi-objective Evolutionary Algorithms (MOEAs) are highly sensitive to noise and uncertainty [8], handling dynamism and noise correctly is critical for finding a good and robust set of solutions. Furthermore, the noise has an impact on the convergence and the diversity of evolutionary multi-objective algorithms. To the best of our knowledge, this paper is the first attempt to study this scenario. First, a flexible noise model integration for Dynamic Multi-objective Optimization Problems (DMOPs) is proposed to contaminate noise to the DMOPs and control its characteristics.

We propose two new robust methods to differentiate between the various forms of dynamism and noise points in noisy DMOPs. The performance of proposed change detection mechanisms and the other popular change detection mechanisms in literature are tested using the proposed noisy dynamic multi-objective optimization problems under different sets of parameters. Our experimental results demonstrate and validate the efficiency of proposed change detection techniques in both isolating noises from real environmental changes and in reducing noise effects on the performance of DMOEAs.

The originality of this paper is summarized in the following points:

- A new mechanism is proposed to incorporate noise into dynamic multi-objective optimization problems with the ability to control the characteristics of the noise.
- Two effective and robust change detection mechanisms are proposed to detect the real environmental changes of dynamic multi-objective optimization problems in the existence of noise.
- We combine the proposed change detection mechanisms with dynamic multi-objective evolutionary algorithms to measure the performance of these algorithms when dealing with noisy optimization problems.

The rest of this paper is organized as follows: Section 2 gives a brief research summary on both dynamic multi-objective optimization and noisy evolutionary optimization. The proposed model for noise integration into Dynamic Multi-objective Optimization Problems (DMOPs) is given in detail in Section 3. In Section 4, we present our proposed change detection schemes for noisy DMOPs. The benchmarks and performance metrics of the experimental study are described in Section 5. The experimental results and discussions are given in Section 6. In Section 7, we end up with conclusions and future work.

## 2. Related work

In a dynamic multi-objective optimization problem (DMOP), there are at least two conflicting objectives where the objective(s), constraint(s), or problem parameter(s) changes over time. In a set of conflicting objectives, an improvement in one objective leads to the deterioration of one or more objectives. The formal definition of a DMOP is given in the following equation:

$$\begin{aligned} &minimize f(x,t) = \{f_1(x,t), f_2(x,t), ....f_M(x,t)\} \\ &\quad subject \quad to \\ &\qquad g_i(x,t) \leq 0, i = 1,2,...N_1 \\ &\qquad h_j(x,t) = 0, j = 1,2,...N_2 \\ &\qquad x = (x_1, x_2, ....x_n) \quad and \quad x \in [x_{min}, x_{max}] \end{aligned} \tag{1}$$

where x represents the decision variables, M is the number of objectives, and $f(x,t)$ is the set of objectives in the optimization problem that change over time t. The terms $g(x,t)$ and $h(x,t)$ represent the inequality and equality constraints of the optimization problem. The main goal of a Dynamic Multi-objective Evolutionary Algorithm (DMOEA) is to find and track the set of changing optimal solutions over time.

According to Farina et al. [9], depending on where the change occurs, DMOPs can be classified into four types:

- *Type 1*. The Pareto Optimal Set (POS) changes but the Pareto Optimal Front (POF) remains static.
- *Type 2*. Both the POF and the POS change over time.
- *Type 3*. The POF changes over time but the POS remains static.
- *Type 4*. Both the POF and the POS remain static; i.e. there is no change.

In this section, we present a short review of related work in two parts, where the first subsection summarizes the literature survey on dynamic multi-objective optimization and the second one discusses noisy evolutionary optimization.

### 2.1. Dynamic multi-objective optimization

In the literature, a large number of Dynamic Multi-objective Evolutionary Algorithms (DMOEAs) have been successfully applied to solve Dynamic Multi-objective Optimization Problems (DMOPs) [10]. The key issue for most of the existing DMOEAs is the adaptation to new environments after changes as quickly as possible. Therefore, the vast majority of current DMOEAs utilize both change detection and change reaction components to detect and respond to the changes. According to these two components, we can classify the proposed DMOEAs into four categories: *diversity introduction*, *memory-based*, *prediction-based*, and *machine learning-based* approaches.

In the diversity introduction approach, a set of random or semi-random solutions is inserted into the population after each change detection. The number of inserted solutions and the randomness degree of solutions are the main two factors that researchers utilize to enhance the convergence and diversity speed. The number of replaced solutions can be fixed as in [11] where 20% of solutions are replaced with completely new random solutions; or it is variable as proposed in [12,13] where the number of random solutions injected is dynamically adjusted according to the severity of change. Based on conducted results, the dynamic diversity introduction algorithms show better robustness and outperform other static diversity introduction methods. Updating the mutation rate based on the diversity degree of the solutions is another mechanism that can be performed to introduce the diversity [14].

The algorithms in the second category, called memory-based techniques, use a fixed or changed-size memory/archive to periodically store or accumulate the best set of solutions for future use [15]. Usually, after a change occurs, some of these stored solutions are reused and inserted into the population [16–18]. The memory-based DMOEAs achieve very good results when environmental changes occur periodically in the environment. The memory-based mechanisms are often merged with other mechanisms from other categories such as diversity introduction [16] and prediction-based methods [17].

The main idea for the prediction-based category is keeping a brief history of the past good solutions, called Pareto Optimal Set (POS) in multi-objective optimization problems, and exploiting this information to predict/estimate the next POS after the next change [19]. Prediction-based algorithms have been analyzed in many papers and become popular especially when the considered optimization problem has regular and predictable environmental changes [20–23].

The last category, machine learning-based approaches is the most recent one in that researchers utilize one of the machine learning techniques to accurately estimate the next POS after an environmental change. Actually, the machine learning-based approach is similar to the prediction-based approach since both of them use the history of good solutions to estimate the next POS after a change occurs. The main difference is that in machine learning-based algorithms, the selected learning model needs to be trained before use, whereas in prediction-based algorithms, the model can be used directly without spending time in training. Support Vector Machine (SVM) [24,25]and transfer learning [26] are two example algorithms from the literature.

### 2.2. Noisy evolutionary optimization

Many real-world optimization problems have noise in the form of several models and distributions such as Gaussian and Uniform distributions [27]. In this subsection, we summarize the most important approaches and algorithms that deal with noise for both single-objective and multi-objective optimization problems. To the best of our knowledge, there is no research work that considers noise for DMOPs. Therefore, all discussions in this subsection are based on static optimization problems.

The averaging or sampling approach is one of the early and popular strategies to mitigate the effect of noise on the evolutionary algorithm [28]. In this strategy, the noisy objective functions are evaluated a number of times and then a mean or average-based aggregated function is used to estimate the fitness value. Although averaging strategy helps in diminishing the impact of noise by periodically evaluating the objective functions, it may fail in some cases when a large sample size is used with a small population size [29]. Therefore, many sampling schemes were proposed to handle the issues of simple sampling such as Dynamic Sampling (DS). The DS schemes use a low fixed re-sample rate in the cases of low noise levels since it is approved to be sufficient to obtain good results, whereas in the cases of larger noise levels gradually increasing the number of minimum samples for elite members improves the performance [30,31].

Since noise may not always be in the form of uniform distributions such as normal or Gaussian [32], alternative noise handling mechanisms with the ability to adapt to nonuniform distributions are needed. An approach called effective fitness estimation was proposed to work in such situations [33]. In this approach, the authors estimate the quality of some of the solutions rather than evaluating every solution using the previously observed objective function values of neighboring individuals. Interpolation and regression algorithms were tested and the results demonstrated that it is possible to reach an acceptable fitness level much faster than if all individuals are evaluated.

Implicit averaging is another approach that utilizes a dynamic population sizing mechanism to increase the size of the evolutionary algorithm population to reduce the noise impact on generated solutions [34].

Additionally, a set of mechanisms are proposed to improve evolutionary search and to guide evolutionary algorithms toward optimal regions in the case of noisy environments. Mutation adaptation [35], Bayesian frequent model [27], and Memetic-based mechanism [36] are examples of such mechanisms to be used as noise-handling strategies.

As mentioned in Section 1, existing DMOEAs assume that the optimization problem is clear of noise; and there is only one type of uncertainty which is the dynamism in objective functions, constraints, or problem parameters. On the other hand, in noisy

evolutionary optimization, most of the current research work target only the noise and they ignore the possibility of another type of uncertainty. To fill this research gap and merge these two scenarios, we propose a DMOEA that can deal with these two uncertainty types simultaneously, as part of this study.

## 3. Noise model integration for dynamic multi-objective optimization problems

In order to evaluate the impact of noise on DMOPs, it is important to identify the appropriate benchmark functions with diverse characteristics. Although there are a large number of benchmark problems for dynamic multi-objective optimization, they do not consider contamination with noise [9,37]. In this study, we integrate a noise model into a set of selected DMOPs. In order to control the several characteristics of a noisy environment, our proposed noise generator mechanism utilizes three factors:

- Noise level (L): This factor controls the severity level of the noise. A low noise level provides that the objective function values after adding noise will be close to the real values. On the other hand, a high noise level leads to different values that are far from the real ones for objective functions.
- Noise percentage (P): This factor controls the amount of noise required to be incorporated into the optimization problem. A small value indicates that noise rarely occurs whereas a large value is used in more frequent noise cases.
- Number of noisy objectives (N): This factor determines the number of objectives in a DMOP that is contaminated with noise. Similar to real-world scenarios, in some cases, the noise exists in one or in a subset of the objective functions. Therefore, some objective functions give accurate measurements whereas others are not. In the worst scenario, all objective functions of the problem have noise.

In this paper, noise is incorporated as an additive factor to the real objective functions. As stated in the related work [7,5], the noise is normally distributed with zero mean and variance $\sigma^2 = 1$.

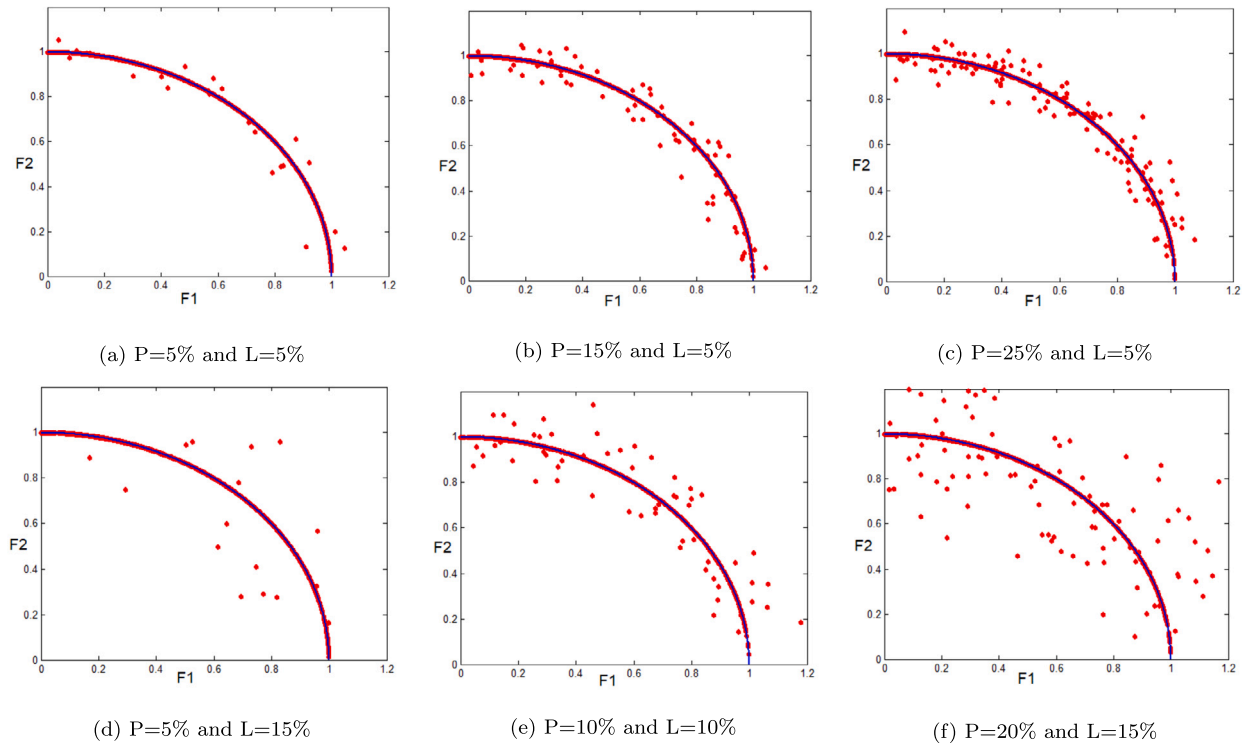
Our proposed noise generation mechanism can be used with any dynamic multi-objective test problem. Eq. (2) and Eq. (3) give the definition of FDA1 and FDA4 benchmarks after incorporating the noise generation mechanism.

$$FDA1: \begin{cases} f_1(x_I) = x_1 + (\mu * noise) \\ f_1 = h.g \\ g(x_{II}) = 1 + \sum_{x_i \in x_{II}} (x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} + (\beta * noise) \\ G(t) = \sin(0.5\pi t), \\ where \\ \quad x_I = (x_1) \in [0,1], \\ \quad x_{II} = (x_2, \ldots, x_n) \in [-1,1] \\ \quad noise = L * Norm(0, \sigma^2) \end{cases} \tag{2}$$

$$FDA4: \begin{cases} f_1(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \prod_{i=1}^{M-1} \cos(\frac{x_i \pi}{2}) \\ \qquad +(\mu * noise) \\ f_2(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \sin(\frac{x_1 \pi}{2}) \\ \qquad +(\beta * noise) \\ where \\ \quad g(\mathbf{x}_{II}) = \sum_{x_i} \in \mathbf{x}_{II} (x_i - G(t))^2 \\ \quad G(t) = |\sin(0.5\pi t)|, \\ \quad \mathbf{x}_{II} = (x_M, \ldots, x_n), \\ \quad x_i \in [0,1] \quad i = 1 : n \\ \quad noise = L * Norm(0, \sigma^2) \end{cases} \tag{3}$$

$$t = \frac{1}{n_t} * \frac{\tau}{\tau_t} \tag{4}$$

where $L$ represents the noise level (i.e., the severity of noise; $Norm$ denotes the normal distribution function; $t$ is the time variable that can be calculated using Eq. (4); $\mu$ and $\beta$ are binary variables to determine the functions that the noise will be incorporated. If both of them are assigned to one, it indicates that noise exists in both objective functions. Fig. 1 shows the effect of using different noise percents ($P$) and noise levels ($L$) on the Pareto Optimal Front (POF) of the FDA4 benchmark at time $t = 0$. As shown in the figure, when the value of noise percent (P) increases, the distortion in the real Pareto Optimal Front (POF) increases since more solutions move away from their original positions on the POF. Similarly, increasing the value of noise level (L) leads to generating different solutions that are far away from the POF.




(a) P=5% and L=5% (b) P=15% and L=5% (c) P=25% and L=5%

(d) P=5% and L=15% (e) P=10% and L=10% (f) P=20% and L=15%

**Fig. 1.** Adding Noise to the FDA4 test problem using different values of noise percent (P) and level of noise (L).

## 4. Detecting changes in a noisy environment

A large number of dynamic multi-objective evolutionary algorithms consider detecting environmental changes as a crucial phase when dealing with various forms of dynamism. The main reason is that these algorithms apply a *detect-and-respond mechanism* to handle the dynamism of the environment. More specifically, a proper responding mechanism is executed only in the case a change is detected. Therefore, the efficiency of the selected change detection method significantly affects the quality of the generated solutions.

There are two different types of strategies for detecting environmental changes, which are sensor-based detection and population-based detection [38,12]. For the former type, a number of selected sensors are distributed over the feasible landscape and re-evaluated in every generation [39,40]. On the other hand, the behavior of the solutions is analyzed periodically using a non-parametric statistical test to detect if a change occurred in the environment for the population-based strategies. Wilcoxon–Mann–Whitney and Jensen–Shannon are two common examples for the second category [41,42]. It is worth mentioning that all proposed change detection methods for DMOPs in the literature assume that the environment is stable and there is no noise. To the best of our knowledge, our work is the first attempt to detect the changes for noisy DMOPs and measure the performance of existing change detection methods in such noisy environments.

In this paper, we propose two new mechanisms for detecting changes in noisy DMOPs, which are designed by adapting sensor-based change detection strategies for noisy environments.

### 4.1. Multi-Sensor Detection Mechanism (MSD)

The first phase of this mechanism is to select a number of solutions ($N$) as sensors. There are various alternatives in the literature [40] for selecting sensors from the population that is evolving by the DMOEA. One of them is the $pRANK$ mechanism which selects one solution from every domination rank/level [12]. Another alternative is to select sensors randomly out of the population (called $np1$ mechanism in [12]), which determines a set of solutions randomly other than the solutions of the population. In this study, we consider the $np1$ scheme since it obtains the highest detection accuracy.

In the second phase, each selected sensor is reevaluated in every generation and its objective function values are compared with the previous ones. To mitigate the effect of noise on the detection performance, all sensors are evaluated together and a difference is considered to have occurred only if a change is detected in a number of sensors greater than a threshold $T$. Here, $T$ is a threshold that represents the minimum number of sensors that must detect a change to determine a change detection in the environment. If the number of sensors that detect a change is less than $T$, then it indicates that either there is no change or some noise leads to slight sensor alarms. In this case, the algorithm has no action to perform. The main difference between the proposed and other existing change detection methods for DMOPs is that detecting a change in one sensor is not enough to declare a change in the environment. In other words, a change is detected in the environment only if $T$ or more sensors detect a change. Determining the value of the

**Algorithm 1** Multi-Sensor Detection Mechanism (MSD).

```
begin
⋮
i ← 1
while i < Numberof Sensors do
    S_i ← select one sensor randomly
    F_i ← evaluate sensor S_i
    i ← i + 1
end while
while gen < MaxNumberof Generations do
    ChangedSensors ← 0
    while i < Numberof Sensors(N) do
        F_gen ← reevaluate the objective functions of S_i
        if F_gen−1 ≠ F_gen then
            ChangedSensors ← ChangedSensors + 1
        end if
        i ← i + 1
        F_gen−1 ← F_gen
    end while
    if ChangedSensors ≥ T then
        ChangeHandling() {A change is detected}
    else
        {No action required}
    end if
    ⋮
    gen ← gen + 1
end while
end
```

threshold $T$ is important and significantly affects the performance of the detector. In general, selecting a small value of $T$ makes the proposed detection method more sensitive to noise which leads to considering many noise points as environmental changes. On the other hand, selecting a large value of $T$ leads to excluding many environmental changes and determining them as noise instead. Experimentally, it is found that the best performance is obtained when the value of $T$ is 40% of used sensors. Algorithm 1 describes the steps of the proposed change detection mechanism.

### 4.2. Welford-Based Detection Mechanism (WBD)

Welford's method is a popular single-pass algorithm for computing the running variance or standard deviation. It is used in many applications especially in online data streaming when the analyzed data is being collected without enough storage to store all values [43]. Welford's algorithm computes the variance in a single pass by looking at the differences between the sums of squared differences for N and N-1 instances where N is the number of currently arrived instances. In order to utilize Welford's algorithm for change detection in noisy environments, it is incorporated with a sensor-based detection mechanism. Algorithm 2 describes the basic steps of the WBD change detection mechanism. As the first phase, a number of sensors are selected randomly out of the population as in $np1$ mechanism [12]. Then, the mean and the variance of each objective function are computed during the population evolving process using Welford's method. The values of sensors' evaluations do not need to be stored in every generation for a second pass. Instead, the following equations are used:

$$mean_n = mean_{n-1} + \frac{x_n^i - mean_{n-1}}{N} \tag{5}$$

$$\sigma_n^2 = \sigma_{n-1}^2 + \frac{(x_n^i - mean_{n-1})(x_n^i - mean_n) - \sigma_{n-1}^2}{N} \tag{6}$$

$$x_n^i = \frac{\sum_{i=1}^{S} (F(x^i, n) - F(x^i, n-1))}{S} \tag{7}$$

where $mean_{n-1}$ and $\sigma_{n-1}$ denote the mean and the variance before the arrival of instance $x_n^i$, respectively. Similarly, $mean_n$ and $\sigma_n$ denote the mean and the variance after $x_n^i$ arrives, where $S$ is the number of sensors.

To make the proposed mechanism robust and independent of the objective function values, we apply Welford's method on the average difference between the current and the previous function values that are computed using the selected sensors. Specifically, $x_n^i$ denotes the average difference of objective functions $F(x^i, n)$ and $F(x^i, n-1)$ before and after the current generation, respectively, for the considered objective function $F^i$ as shown in Eq. (7). A change is determined by checking if the average difference between the current and the previous function values (i.e., $x_n^i$ value) is within an acceptable range. As shown in Algorithm 2, this range is determined by the computed mean $mean_n$ and variance $\sigma_n^2$ of each objective function.

**Algorithm 2** Welford-Based Detection Mechanism (WBD).

```
begin
⋮
i ← 1
mean[NumberOfObjectives] ← 0
variance[NumberOfObjectives] ← 0
while i < Numberof Sensors do
    S_i ← select one sensor randomly
    F_i ← evaluate sensor S_i
    i ← i + 1
end while
while gen < MaxNumberofGenerations do
    while i < NumberOfObjectives do
        Compute x_n^i using Eq. (7)
    end while
    while i < NumberOfObjectives do
        if x_n^i < mean[i] − (3 ∗ variance[i]) or x_n^i > (mean[i] + 3 ∗ variance[i]) then
            ChangeHandling() {A change is detected}
        else
            {No change or noise}
            Update mean[i] using Eq. (5)
            Update variance[i] using Eq. (6)
        end if
        i ← i + 1
    end while
    ⋮
    gen ← gen + 1
end while
end
```

**Table 1**
Selected dynamic multi-objective test problems and their properties.

| Problem | Type | POF | Test Suite |
|---|---|---|---|
| FDA1 | Type 1 | convex and continuous | FDA |
| FDA4 | Type 1 | non-convex and continuous | FDA |
| JY1 | Type 1 | convex and non-convex | JY |
| FDA5 | Type 2 | non-convex and continuous | FDA |
| dMOP2 | Type 2 | convex and continuous | dMOP |
| JY2 | Type 2 | shape changing | JY |
| dMOP1 | Type 3 | convex and continuous | dMOP |
| HE2 | Type 3 | convex and discontinuous | HE |
| SJY5 | Type 4 | non-convex | SJY |

## 5. Experimental design

### 5.1. Benchmarks

In order to incorporate noise and evaluate the performance of our change detection mechanisms, nine dynamic multi-objective test problems are selected from five different test suites. The test problems are selected to consider all four types of DMOPs given in Section 2.1. Table 1 summarizes the properties of the selected test problems.

The FDA [9] test suite is one of the most widely used test suites for evaluating the performance of DMOEAs, where three test problems are selected (FDA1, FDA4 and FDA5) from this suite. The problems of this test suite are mainly designed based on DTLZ multi-objective test problems [44]. Every test problem has different characteristics where FDA1 has a linear POF, and both FDA4 and FDA5 have non-convex POF. Additionally, two test problems (dMOP1 and dMOP2) are selected from the dMOP test suite proposed by Goh and Tan [15]. Both of them have a convex POF but they have different dynamism types. In our experimental study, two test problems (JY1 and JY2) are selected from the JY [45] test suite, where the JY1 is a type I problem that is mainly designed to test the convergence speed of the DMOEA. The JY2 is a type 2 problem where the POF shape changes over time. Furthermore, the HE2 problem is selected from the HE test suite [37]. The HE2 is a Type 3 problem with a discontinuous POF (the POF has various disconnected sub-regions). The last test problem is the SJY5 from the SJY framework [46]. This problem is the only Type 4 problem and has a stationary and non-convex POF.

### 5.2. Performance metrics

In our experimental study, we consider a total of five performance metrics that can be classified into two sets. The first set targets to evaluate the performance of the proposed change detection schemes, whereas the second set is used to measure the performance

of DMOEAs after incorporating our change detection schemes into them. The first set includes the first and the second metrics given below, where the remaining ones belong to the second set.

- *True Positive Rate (TPR)*: It measures the percentage of the correctly detected changes in the environment. The best value of this metric is one when the considered detector correctly identifies all changes in the environment. TPR can be calculated using the following equation:

$$TPR = \frac{C_c}{C} \tag{8}$$

  where $C_c$ is the correctly detected changes and $C$ is the total number of changes that occurred in the environment.
- *True Negative Rate (TNR)*: This metric represents the generations with no change that are correctly identified as generations with no change. It is used in our experiments to measure the percentage of false-alarm or fake changes that are falsely identified as changes by detectors where there is no change in fact. Usually, these false detection cases are due to the existence of noise in the environment. The best-preferred value of this metric is one or 100% which indicates that there is no falsely detected change in the environment.

$$TNR = \frac{C_f}{G_{NoChange}} \tag{9}$$

  where $C_f$ is the falsely detected changes and $G_{NoChange}$ is the total number of executed generations without any change.
- *Mean Inverted Generational Distance (mIGD)*: This metric is derived from the $IGD$ metric and it is used to test the convergence and the diversity of the solutions that are obtained from the algorithms [12]. For dynamic environments, the mIGD metric is usually computed by averaging the values of the generational distance of all generations for the obtained solutions [47] as described in the following Equation:

$$\text{mIGD} = \frac{1}{g_n} * \sum_{t=1}^{g_n} IGD_t \tag{10}$$

  where $g_n$ is the number of generations and $IGD_t$ is the generational distance of the generation $t$.
- *Mean inverted generational distance just before the change (mIGDB)*: This metric is similar to mIGD metric. The only difference is that it computes the average performance of DMOEAs at the last generations just before changes [48] other than averaging the values of all generations as in mIGD. This metric is calculated using the following equation,

$$\text{mIGDB} = \frac{1}{C_n} * \sum_{t=1}^{C_n} IGD_t \tag{11}$$

  where $C_n$ is the number of changes and $IGD_t$ is calculated just before the next change.
- *Scott's Spacing (SS)*: Scotts spacing metric [49] is a popular metric that used to measure the diversity of the obtained solutions by estimating the distribution degree of the discovered Pareto front. This metric is defined using the following Equation:

$$\text{SS} = \sqrt{\frac{1}{n-1} \sum_{1}^{n} (D_i - D_m)^2} \tag{12}$$

  where $D_i$ is the Euclidean distance between the ith member in POF and its nearest member in POF; and $D_m$ is the average value of $D_i$.

## 6. Experiments and discussion

We present an empirical evaluation of the proposed change detection techniques (the MSD and the WBD techniques) in two parts. The first subsection evaluates the performance of the change detectors independently. On the other hand, the change detectors are incorporated with a set of selected DMOEAs and the performance of the integrations is presented in the second subsection.

### 6.1. Evaluating performance of detection schemes in noisy environments

In this section, we study the performance of the MSD and the WBD techniques in noisy environments. Four change detection techniques were selected for comparisons in our empirical study: They are the NP1, the PR, the PRank techniques selected from [12] and the Kolmogorov-Smirnov (KS) technique [38]. NP1 is a sensor-based scheme where sensors are selected randomly out-of-population and then reevaluated in each generation to detect any change in the environment. PR scheme simply selects a set of solutions randomly from the population of the evolutionary algorithm to serve as sensors. The PRank is similar to the PR mechanism except that the sensors are selected from different ranks/levels to ensure a good distribution of sensors. Unlike the previous three techniques, the last one utilizes the Kolmogorov-Smirnov statistical test to detect if there is any change in the environment [42]. Specifically, KS is used to measure the degradation of the population that may be due to environmental changes.

To provide fair performance comparisons, all change detection schemes are integrated into the DNSGA-II-A algorithm as a baseline evolutionary algorithm. For all test problems, the severity of change ($n_t$) is set to 10, the number of variables is set to 10, and the

**Table 2**
Average values of TPR and TNR metrics for each change detection scheme using different change frequencies.

| Problem | (nt, fr) | NP1 | | PR | | PRank | | KS | | MSD | | WBD | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR |
| FDA1 | (10,10) | **100.0** | 47.1 + | 92.7 + | 62.6 + | 93.3 + | 63.6 + | 79.0 + | 33.4 + | **100.0** | 98.0 | **100.0** | **98.1** |
| | (10,20) | **100.0** | 47.6 + | 93.0 + | 65.7 + | 92.6 + | 65.8 + | 75.8 + | 51.5 + | **100.0** | **98.3** | **100.0** | 97.9 |
| FDA4 | (10,10) | **100.0** | 47.7 + | 93.9 + | 58.9 + | 93.5 + | 59.9 + | 77.0 + | 36.6 + | **100.0** | **98.1** | **100.0** | 98.0 |
| | (10,20) | **100.0** | 47.3 + | 93.2 + | 63.9 + | 93.2 + | 63.9 + | 83.5 + | 42.0 + | **100.0** | **98.2** | **100.0** | 97.8 |
| JY1 | (10,10) | **100.0** | 47.3 + | **100.0** | 54.3 + | 99.9 | 54.5 + | 61.4 | 59.4 + | **100.0** | **98.2** | **100.0** | 97.8 |
| | (10,20) | **100.0** | 47.4 + | 99.8 + | 55.6 + | 99.8 | 55.9 + | 77.6 + | 53.4 + | **100.0** | **98.2** | **100.0** | 97.9 |
| FDA5 | (10,10) | **100.0** | 47.3 + | **100.0** | 58.2 + | **100.0** | 58.5 + | 85.5 + | 39.3 + | **100.0** | **98.2** | **100.0** | 97.9 |
| | (10,20) | **100.0** | 47.0 + | **100.0** | 62.6 + | **100.0** | 63.2 + | 89.9 + | 46.2 + | **100.0** | **98.1** | **100.0** | 98.0 |
| dMOP2 | (10,10) | **100.0** | 47.7 + | 99.0 | 62.7 + | 98.9 | 63.7 + | 67.6 + | 40.6 + | 99.9 | **98.3** | **100.0** | 97.9 |
| | (10,20) | **100.0** | 47.2 + | 98.9 | 66.2 + | 98.5 | 66.3 + | 69.6 + | 50.4 + | **100.0** | **98.2** | **100.0** | 97.7 |
| JY2 | (10,10) | **100.0** | 47.0 + | 96.9 + | 61.5 + | 97.0 + | 66.7 + | 75.6 + | 48.1 + | **100.0** | **98.4** | **100.0** | 97.8 |
| | (10,20) | **100.0** | 47.5 + | 96.8 + | 64.4 + | 97.1 + | 64.5 + | 83.9 + | 47.4 + | **100.0** | **98.2** | **100.0** | 97.9 |
| dMOP1 | (10,10) | 88.6 + | 47.4 + | 78.2 + | 68.0 + | 81.2 + | 67.4 + | 63.2 + | 38.7 + | 65.6 + | **98.2** | **100.0** | 98.0 |
| | (10,20) | 88.7 + | 47.3 + | 80.9 + | 68.1 + | 83.8 + | 68.3 + | 65.8 + | 37.3 + | 66.1 + | **98.2** | **100.0** | 98.1 |
| HE2 | (10,10) | 97.1 + | 47.3 + | 89.6 + | 68.0 + | 89.9 + | 67.4 + | 66.2 + | 48.5 + | 91.7 + | **98.2** | **100.0** | 98.1 |
| | (10,20) | 96.8 + | 47.6 + | 90.5 + | 68.1 + | 90.6 + | 68.3 + | 67.8 + | 48.8 + | 91.9 + | **98.2** | **100.0** | 97.8 |
| SJY5 | (10,10) | 86.5 + | 47.5 + | 34.1 + | 67.7 + | 33.4 + | 68.2 + | 65.2 + | 35.5 + | 64.0 + | **98.2** | **100.0** | 98.1 |
| | (10,20) | 84.9 + | 46.9 + | 33.6 + | 67.8 + | 32.2 + | 68.0 + | 62.1 + | 37.6 + | 64.9 + | **98.2** | **100.0** | **98.2** |

number of objective functions is set to 2. The number of changes is equal to 100 for every run, and 30 independent runs are executed for each test instance. Both of the noise parameters: noise level (L) and noise percent (P) were set to 10%, and the noise was presented in only one objective function unless otherwise stated. The best results are marked in bold font for each metric separately. For each test instance, to check if the best algorithm statistically outperforms the other algorithms or not, the Wilcoxon rank-sum test [50] was applied at a significance level of 0.05. Specifically, the Wilcoxon rank-sum test is carried out between the best algorithm (highlighted in bold) and all other algorithms, and a '+' sign is used to indicate that it significantly outperforms the considered algorithm for the given test instance.

The TPR and TNR values (given as percentages) for the nine noisy DMOPs with different frequencies of change values are shown in Table 2. It is observed that the proposed change detection schemes (the MSD and the WBD techniques) significantly outperform the other schemes in most of the considered cases. Furthermore, while other schemes suffer from oscillating performance, the MSD and the WBD schemes show high-performance stability against noise in the majority of test problems. On the other hand, the results confirm that the critical issue in noisy environments is the TNR value which indicates many false alarms of change occurring. Detecting false changes and activating corresponding the response mechanism in case there is no real change can significantly degrade the performance of the evolutionary algorithm and slows down the convergence process. As shown in Table 2, NP1, PR, PRank, and KS schemes achieve significantly lower TNR values than the proposed strategies, where the maximum obtained value is 68.3. In a noisy environment, in the best case, the traditional change detection schemes falsely detect environmental change in more than 30% of the generations. This impact will be more severe if we consider 1,000 generations and the change detection strategy detects changes in the 300 generations when there is a total of 10 real changes.

Table 3 shows the performance of each change detection scheme when we have noise in one or two objective functions. The other parameters are set to the values indicated in the previous experiment. The results demonstrate that the proposed schemes still outperform the other schemes and achieve better results even in the case of adding noise to both objective functions. The WBD scheme shows more promising results than the MSD, especially in the case of adding noise to both objectives where it obtains the best results in 8 out of 9 cases based on the TNR metric. In case of a highly noisy environment, the performance degradation of the MSD scheme is much more severe than that of the WBD scheme. This is because the sensors of the MSD scheme detect more changes as a result of noise and it becomes unable to distinguish between real and not-real change points. On the other hand, the WBD scheme is more robust against higher noise levels since its mechanism does not rely directly on sensors and can estimate whether there is a real change or not using Welford's method. Therefore, the WBD scheme is outperformed in only one case for the TNR metric and in three cases for the TPR, and it provides the best-achieved results among the other schemes for all remaining cases.

Another experiment is conducted in order to validate the impact of increasing the noise percentage (P) on the performance of the proposed detection schemes. In this experiment, the number of noisy objectives is set to one, the noise level is fixed to 10%, and the noise percentage (P) (i.e. the noise probability) is set to different values: 5%, 10%, 15%, 20%, and 25%. Fig. 2 and Fig. 3 show the results of TPR and TNR metrics, respectively, for the nine noisy dynamic test problems. According to the results of Fig. 2, all schemes (except the KS scheme) achieve good results (more than 90% TPR values), especially in Type 1 and Type 2 test problems. On the other hand, the performance of the schemes increases by increasing the noise probability, which is not the expected behavior. Increasing noise percentage helps the sensors of traditional detection schemes to detect more changes although they may not be the

**Table 3**
Average values of TPR and TNR metrics for each change detection scheme by adding noise to one and two objective functions.

| Problem | Noisy Funs | NP1 | | PR | | PRank | | KS | | MSD | | WBD | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR | TPR | TNR |
| FDA1 | 1 | **100.0** | 47.1 + | 92.7 + | 62.6 + | 93.3 + | 63.6 + | 79.0 + | 33.4 + | **100.0** | 98.0 | **100.0** | **98.1** |
| | 2 | **100.0** | 43.3 + | 93.0 + | 63.5 + | 92.8 + | 63.8 + | 82.0 + | 31.2 + | **100.0** | **84.3** | 99.8 | 50.4 + |
| FDA4 | 1 | **100.0** | 47.7 + | 93.9 + | 58.9 + | 93.5 + | 59.9 + | 77.0 + | 36.6 + | **100.0** | **98.1** | **100.0** | 98.0 |
| | 2 | **100.0** | 43.0 + | 94.0 + | 61.6 + | 93.2 + | 61.9 + | 77.4 + | 35.1 + | **100.0** | 84.6 + | 99.0 | **96.1** |
| JY1 | 1 | **100.0** | 47.3 + | **100.0** | 54.3 + | 99.9 | 54.5 + | 61.4 + | 59.4 + | **100.0** | **98.2** | **100.0** | 97.8 |
| | 2 | **100.0** | 42.9 + | 99.9 | 64.8 + | **100.0** | 65.1 + | 60.4 + | 59.8 + | **100.0** | 84.3 + | 99.9 | **96.3** |
| FDA5 | 1 | **100.0** | 47.3 + | **100.0** | 58.2 + | **100.0** | 58.5 + | 85.5 + | 39.3 + | **100.0** | **98.2** | **100.0** | 97.9 |
| | 2 | **100.0** | 43.7 + | **100.0** | 61.1 + | **100.0** | 61.3 + | 86.9 + | 38.5 + | **100.0** | 84.2 | **100.0** | **96.0** |
| dMOP2 | 1 | **100.0** | 47.7 + | 99.0 | 62.7 + | 98.9 | 63.7 + | 67.6 + | 40.6 + | 99.9 | **98.3** | **100.0** | 97.9 |
| | 2 | **100.0** | 43.3 + | 99.2 | 63.1 + | 99.1 | 63.7 + | 69.9 + | 39.7 + | **100.0** | 84.2 + | 98.7 | **95.9** |
| JY2 | 1 | **100.0** | 47.0 + | 96.9 + | 61.5 + | 97.0 + | 66.7 + | 75.6 + | 48.1 + | **100.0** | **98.4** | **100.0** | 97.8 |
| | 2 | **100.0** | 43.3 + | 97.0 + | 61.8 + | 96.9 + | 63.0 + | 76.2 + | 45.4 + | **100.0** | 84.0 | 99.4 | **96.4** |
| dMOP1 | 1 | 88.6 + | 47.4 + | 78.2 + | 68.0 + | 81.2 + | 67.4 + | 63.2 + | 38.7 + | 65.6 + | **98.2** | **100.0** | 98.0 |
| | 2 | **90.3** | 43.4 + | 82.0 + | 65.8 + | 83.8 + | 65.7 + | 66.8 + | 37.8 + | 73.3 + | 83.8 + | 55.6 + | **96.0** |
| HE2 | 1 | 97.1 + | 47.3 + | 89.6 + | 68.0 + | 89.9 + | 67.4 + | 66.2 + | 48.5 + | 91.7 + | **98.2** | **100.0** | 98.1 |
| | 2 | **97.5** | 43.2 + | 90.6 + | 65.5 + | 90.3 + | 65.3 + | 68.2 + | 46.3 + | 92.0 + | 84.1 + | 77.1 + | **96.0** |
| SJY5 | 1 | 86.5 + | 47.5 + | 34.1 + | 67.7 + | 33.4 + | 68.2 + | 65.2 + | 35.5 + | 64.0 + | **98.2** | **100.0** | 98.1 |
| | 2 | **87.2** | 43.1 + | 36.3 + | 65.4 + | 36.1 + | 66.1 + | 68.2 + | 32.2 + | 71.2 + | 84.0 + | 42.7 + | **96.3** |

actual changes in the environment. Those schemes (NP1, PR, and PRank) detect more points by considering some noises as dynamic change points where in fact the number of real changes is fixed.

This case is similar to the over-fitting issue of machine learning algorithms where the trained models catch the noise and use it during the classifier building process which causes high training performance but low performance in the testing phase.

Not only the TPR values but also the TNR values should be analyzed for evaluating the performance of change detection schemes. Examining the results of the TNR metric for all test problems given in Fig. 3 demonstrates that the detection performance of NP1, PR, and PRank schemes decreases significantly by increasing the noise probability. This implies that the traditional change detection schemes are very sensitive to noise and falsely detect more changes by adding more noise to the environment.

The WBD scheme obtains the best and the most robust performance among them since it is not affected by increasing the noise percentage for 7 out of 9 test problems. The MSD scheme also shows acceptable results but its performance slightly degrades for higher noise percentage values. Our proposed change detection schemes perform much better than the traditional change detection schemes when the considered optimization problem is contaminated with noise. This is because most conventional change detection systems are designed to detect all types of changes without considering the severity or potential for noise. Therefore, these algorithms are very sensitive to noise and slight environmental changes. On the contrary, our proposed change detection schemes do not rely directly on the change values detected from reevaluating sensors. Instead, our schemes target to investigate more by evaluating more sensors (as in the MSD scheme), or by calculating the severity of changes and statistically determining if it is real or not (as in the WBD scheme). As a result, the proposed schemes perform much better than the traditional schemes in a noisy environment since they detect the significant changes only and exclude the noise.

### *6.2. The impact of proposed schemes on the performance of DMOEAs*

In this section, we evaluate the impact of using our proposed change detection schemes on the performance of Dynamic Multi-objective Evolutionary Algorithms (DMOEAs) when there is noise in the environment. Four well-known DMOEAs are used in this experiment: NSGAII-A [11]e, CR-NSGAII [12], dCOEA [15], and SGEA [20]. For each algorithm, two different change detection schemes are evaluated. In the first version, the DMOEA is used with its original change detection scheme (which is the PR scheme that is widely used by most of the DMOEAs); and in the second version, the WBD scheme is incorporated instead of the original change detection scheme. All experiments performed in this study are replicated 30 times and the number of environmental changes is set to 10. The results of each DMOEA are compared separately to validate the impact of the incorporated change detection scheme and the best results are highlighted in bold font. The Wilcoxon ranksum test [50] is applied at a significance level of 0.05 for each test problem. Table 4, Table 5 and Table 6 show the results of the mIGD, the mIGDB, and the SS metrics respectively. For each test problem, the noise is added as explained in section 3 and two levels of noise probability (P) are examined: 10% and 20%. The other remaining parameters are set as in the previous section.

Table 4 presents the results of the mIGD metric which allows measuring both the convergence and diversity. The results show that the WBD change detection scheme has a significant impact on the performance of all tested DMOEAs. When a total of 18 test cases are considered, the WBD scheme outperforms the PR scheme in 15 cases for NSGAII-A and CR-NSGAII, 18 cases for dCOEA,

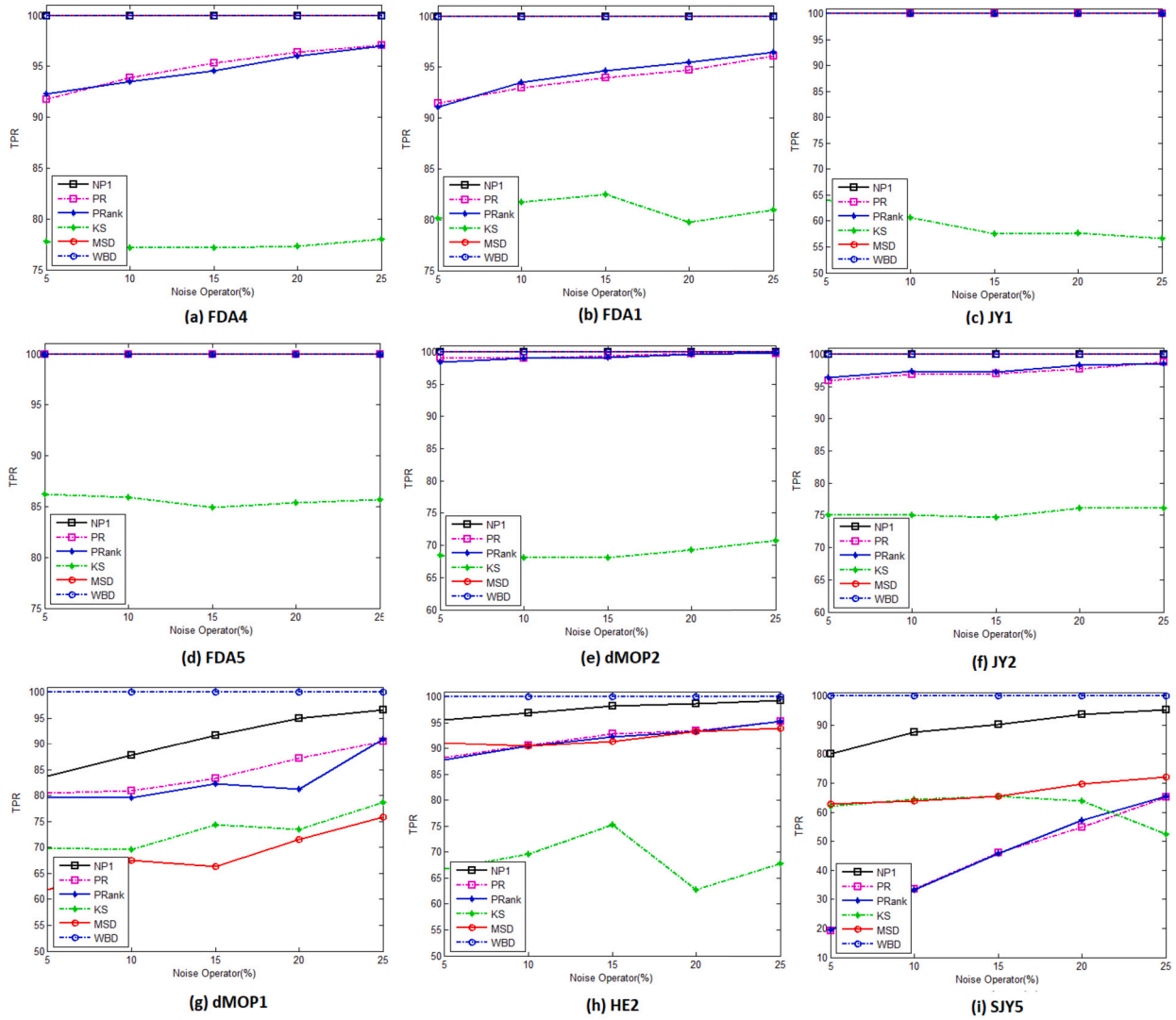



**Fig. 2.** Average values of TPR metric for each change detection scheme by using different values of noise percent (P).

and 12 cases for SGEA. As shown in previous results, the WBD scheme is able to detect most of the change points and filter the noise at the same time. As a result, the DMOEA activates the responding mechanisms only when there is a real change. On the other hand, the PR mechanism (the common detection mechanism in many DMOEAs) is very sensitive to noise which causes activation of the responding mechanism frequently in case of no change in the environment. This action slows down the convergence process of the DMOEA since a set of new solutions is added to the population in most of the responding mechanisms. Moreover, the WBD-integrated algorithms statistically outperform the PR-based versions for most of the test cases considered. Specifically, the DMOEAs with PR scheme obtain statistically better results only for 3 SGEA-based cases (marked by + sign) that are Type 3 and Type 4 test problems. The dCOEA algorithm achieved the highest improvement among other algorithms since the WBD-integrated algorithm significantly outperforms the PR-integrated algorithm in 16 out of 18 cases.

The results of the mIGDB metric are presented in Table 5. This metric measures the performance of the DMOEA just before the next change rather than in every generation as in the mIGD metric. Therefore, it focuses on the overall performance of the DMOEA and neglects the performance behavior during convergence. When the results of Table 5 are examined, we find that WBD-integrated algorithms obtained better results than the mIGD results given in Table 4. Moreover, the WBD-integrated algorithms significantly outperform the PR-integrated algorithms in the majority of the cases considered. The analysis indicates the level of significance of the proposed WBD change detection mechanism for handling noises in an environment.

The last experiment compares the diversity performance of different algorithms in our framework using the SS metric (see Table 6). It is clear that the WBD mechanism has a significant impact on the diversity of the solutions during the evolving of DMOEAs. Except for the Type 4 problem (i.e., the SJY5 test problem), the WBD-integrated algorithms achieve the best results in all test cases. The WBD-integrated algorithms statistically outperform the PR-integrated counterparts in the majority of tested cases,

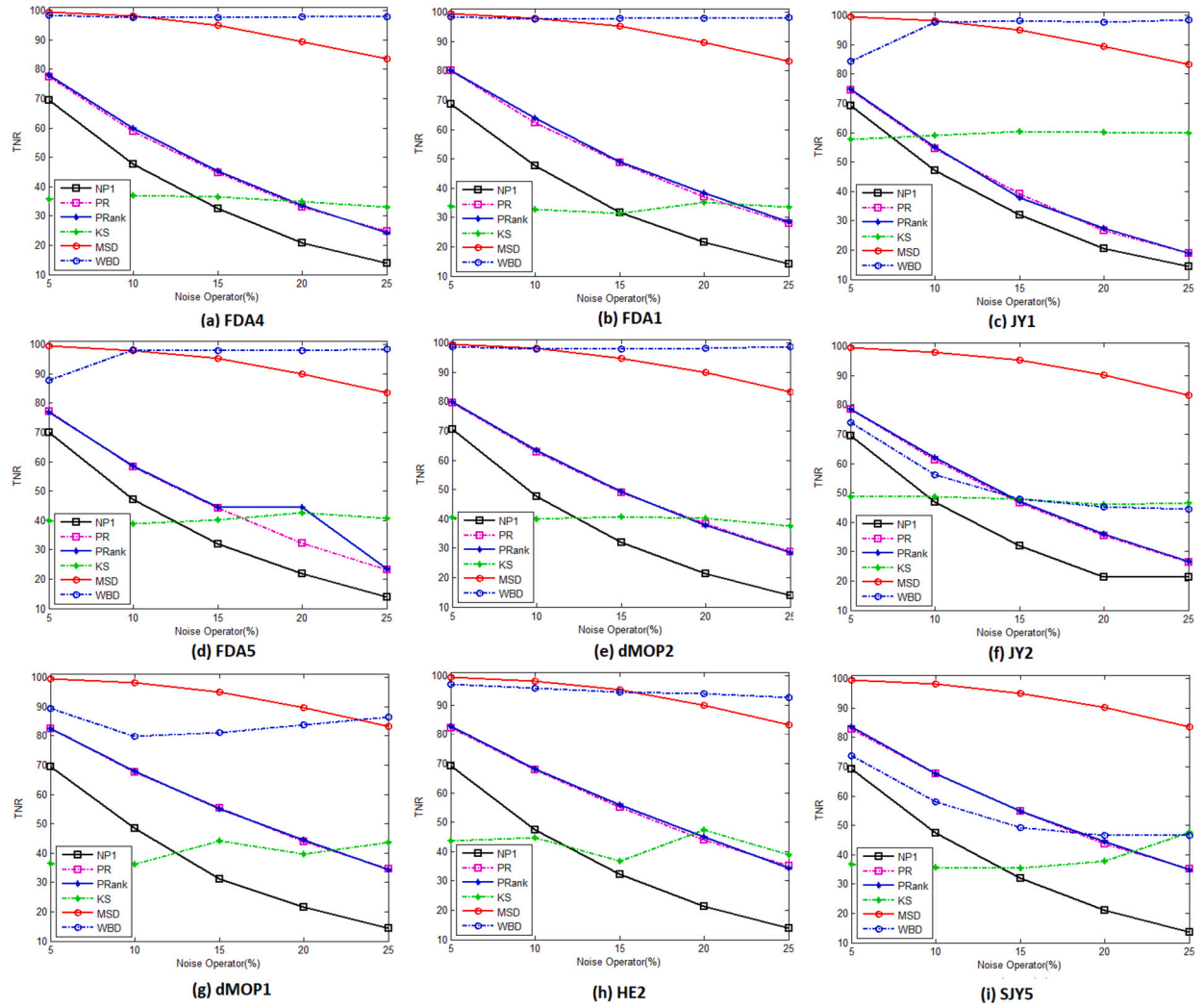



**Fig. 3.** Average values of TNR metric for each change detection scheme by using different values of noise percent (P).

which is represented with the "+" signs in the PR columns. Specifically, for the CR-NSGAII integration, our method statistically outperforms 12 out of 18 cases; whereas, it outperforms 17 out of 18 cases for the dCOEA integration. The results confirm that the proposed change detection mechanism is able to enhance both the convergence and the diversity of the DMOEAs when dealing with noisy environments.

## 7. Conclusions and future work

In this paper, we proposed a new noise model integration for dynamic multi-objective optimization problems to study the situation when two types of uncertainty occur simultaneously. Handling uncertainty in such a complex scenario becomes more challenging since the evolutionary algorithm should be able to distinguish between real environmental changes that resulted from a change in the optimization problem and false-alarm changes that resulted from stochastic noise. Moreover, a new flexible model was proposed to inject the noise into DMOPs and control its parameters. To efficiently handle these multi-uncertainty scenarios, two new change detection techniques were proposed to differentiate between real change points and noise points. They are evaluated and compared with other traditional change detection schemes from different approaches. Furthermore, the proposed change detection schemes were incorporated into a set of Dynamic Multi-objective Evolutionary Algorithms (DMOEAs) to measure the impact of these schemes on the performance of DMOEAs. The experimental results based on nine different noisy dynamic problems confirmed the efficiency of our techniques in accurately detecting the real environmental changes and filtering the noise points. Additionally, the results highlighted the positive impact of the proposed change detection techniques on the performance of DMOEAs when both noise and dynamism exist in the environment.

In future work, alternative noise models can be examined by integrating them into different sets of DMOPs. It can be useful to consider more types of DMOPs such as optimization problems with disconnected POFs and less-detectable DMOPs. Moreover,

**Table 4**
Average values of mIGD metric for four different DMOEAs after incorporating PR and WBD change detection schemes into each algorithm.

| Problem | ($n_t$,P) | NSGAII-A | | CR-NSGAII | | dCOEA | | SGEA | |
|---|---|---|---|---|---|---|---|---|---|
| | | PR | WBD | PR | WBD | PR | WBD | PR | WBD |
| FDA1 | (10,10) | 0.0788 | **0.0762** | 0.0775 | **0.0753** | 0.1309 + | **0.0978** | 0.0370 | **0.0369** |
| | (10,20) | 0.0861 + | **0.0760** | 0.0787 | **0.0789** | 0.1462 + | **0.0963** | 0.0390 | **0.0388** |
| FDA4 | (10,10) | 0.1273 | **0.1253** | 0.1260 | **0.1215** | 0.1474 + | **0.0951** | 0.0509 | **0.0501** |
| | (10,20) | 0.1331 + | **0.1282** | 0.1263 | **0.1257** | 0.1658 + | **0.1018** | 0.0547 | **0.0528** |
| JY1 | (10,10) | 0.4294 + | **0.2495** | 0.2222 + | **0.1821** | 0.1718 + | **0.1221** | 0.0506 + | **0.0468** |
| | (10,20) | 0.4523 + | **0.2701** | 0.3066 + | **0.2494** | 0.2093 + | **0.1350** | 0.0604 + | **0.0501** |
| FDA5 | (10,10) | 0.1333 | **0.1328** | 0.1229 | **0.1216** | 0.3262 + | **0.1959** | 0.7167 | **0.7121** |
| | (10,20) | **0.1337** | 0.1355 | **0.1217** | 0.1219 | 0.4549 + | **0.2092** | 0.7179 | **0.7144** |
| dMOP2 | (10,10) | **0.1152** | 0.1166 | **0.1104** | 0.1112 | 0.0611 + | **0.0402** | **0.1008** | 0.1020 |
| | (10,20) | **0.1162** | 0.1168 | 0.1126 | **0.1115** | 0.0762 + | **0.0370** | **0.1024** | 0.1028 |
| JY2 | (10,10) | 0.1305 + | **0.1190** | **0.1212** | 0.1215 | 0.1717 + | **0.1382** | 0.0703 + | **0.0658** |
| | (10,20) | 0.1557 + | **0.1210** | 0.1299 + | **0.1182** | 0.2030 + | **0.1319** | 0.0788 + | **0.0667** |
| dMOP1 | (10,10) | 0.4016 + | **0.3012** | 0.3578 + | **0.3270** | 0.1230 + | **0.0861** | **0.1710** | 0.3079 + |
| | (10,20) | 0.3801 + | **0.3369** | 0.3523 + | **0.3196** | 0.1477 + | **0.0879** | **0.1713** | 0.2680 + |
| HE2 | (10,10) | 0.2049 | **0.2041** | 0.2043 | **0.2024** | 0.2886 | **0.2762** | **0.1939** | 0.1974 |
| | (10,20) | 0.2116 | **0.2026** | 0.2109 + | **0.1979** | 0.3057 + | **0.2779** | 0.1978 | **0.1954** |
| SJY5 | (10,10) | 0.0628 + | **0.0505** | 0.0512 + | **0.0454** | 0.2328 | **0.2219** | **0.1296** | 0.1527 + |
| | (10,20) | 0.0625 + | **0.0581** | 0.0536 | **0.0514** | 0.2444 + | **0.2229** | 0.1860 + | **0.1663** |

**Table 5**
Average values of mIGDB metric for four different DMOEAs after incorporating PR and WBD change detection schemes into each algorithm.

| Problem | ($n_t$,P) | NSGAII-A | | CR-NSGAII | | dCOEA | | SGEA | |
|---|---|---|---|---|---|---|---|---|---|
| | | PR | WBD | PR | WBD | PR | WBD | PR | WBD |
| FDA1 | (10,10) | 0.0412 + | **0.0374** | 0.0379 + | **0.0365** | 0.1169 + | **0.0892** | 0.0199 | **0.0182** |
| | (10,20) | 0.0470 + | **0.0371** | 0.0400 | **0.0385** | 0.1270 + | **0.0871** | 0.0221 + | **0.0191** |
| FDA4 | (10,10) | 0.0657 | **0.0610** | 0.0602 | **0.0585** | 0.1292 + | **0.0808** | 0.0259 | **0.0246** |
| | (10,20) | 0.0715 + | **0.0633** | **0.0610** | 0.0613 | 0.1394 + | **0.0863** | 0.0297 + | **0.0259** |
| JY1 | (10,10) | 0.3829 + | **0.1858** | 0.1615 + | **0.1248** | 0.1552 + | **0.1100** | 0.0316 + | **0.0272** |
| | (10,20) | 0.4145 + | **0.2028** | 0.2403 + | **0.1847** | 0.1803 + | **0.1204** | 0.0424 + | **0.0301** |
| FDA5 | (10,10) | 0.0691 + | **0.0637** | 0.0575 | **0.0573** | 0.3319 + | **0.2105** | 0.7012 | **0.6967** |
| | (10,20) | 0.0716 + | **0.0668** | 0.0588 | **0.0578** | 0.4574 + | **0.2208** | 0.7023 | **0.6983** |
| dMOP2 | (10,10) | 0.0559 | **0.0548** | **0.0519** | 0.0521 | 0.0589 + | **0.0420** | 0.0751 | **0.0742** |
| | (10,20) | 0.0603 + | **0.0549** | 0.0539 | **0.0520** | 0.0732 + | **0.0378** | 0.0773 | **0.0762** |
| JY2 | (10,10) | 0.0842 + | **0.0724** | 0.0760 | **0.0742** | 0.1544 + | **0.1300** | 0.0548 + | **0.0506** |
| | (10,20) | 0.1071 + | **0.0739** | 0.0813 + | **0.0714** | 0.1761 + | **0.1210** | 0.0622 + | **0.0514** |
| dMOP1 | (10,10) | 0.2312 + | **0.1458** | 0.1887 + | **0.1512** | 0.1051 + | **0.0749** | **0.0719** | 0.1696 + |
| | (10,20) | 0.2001 + | **0.1713** | 0.1872 + | **0.1703** | 0.1232 + | **0.0776** | **0.0763** | 0.1568 + |
| HE2 | (10,10) | 0.2011 | **0.1996** | 0.2009 | **0.1974** | 0.2817 + | **0.2715** | **0.1918** | 0.1943 |
| | (10,20) | 0.2072 | **0.1977** | 0.2066 + | **0.1950** | 0.2999 + | **0.2739** | 0.1951 | **0.1925** |
| SJY5 | (10,10) | 0.0520 + | **0.0387** | 0.0400 + | **0.0340** | 0.2284 + | **0.2185** | **0.1201** | 0.1420 + |
| | (10,20) | 0.0514 | **0.0459** | 0.0422 | **0.0394** | 0.2415 + | **0.2192** | 0.1803 + | **0.1593** |

other types of uncertainty such as the uncertainty that occurs in the design and the environmental variables need to be studied and analyzed. These types can be tested independently and when the existence of other uncertainties such as stochastic noise in the objectives at the same time. In addition, the proposed DMOEAs and change detection schemes will be used to solve real-world optimization problems that have both noise and various forms of dynamism.

**Table 6**
Average values of SS metric for four different DMOEAs after incorporating PR and WBD change detection schemes into each algorithm.

| Problem | ($n_t$,P) | NSGAII-A | | CR-NSGAII | | dCOEA | | SGEA | |
|---|---|---|---|---|---|---|---|---|---|
| | | PR | WBD | PR | WBD | PR | WBD | PR | WBD |
| FDA1 | (10,10) | 0.0310 + | **0.0216** | 0.0251 | **0.0247** | 0.0329 + | **0.0307** | 0.0335 + | **0.0203** |
| | (10,20) | 0.0520 + | **0.0223** | 0.0361 + | **0.0255** | 0.0372 + | **0.0315** | 0.0638 + | **0.0246** |
| FDA4 | (10,10) | 0.0244 + | **0.0205** | 0.0221 | **0.0214** | 0.0392 + | **0.0350** | 0.0356 + | **0.0305** |
| | (10,20) | 0.0267 + | **0.0218** | 0.0235 | **0.0227** | 0.0470 + | **0.0376** | 0.0395 + | **0.0321** |
| JY1 | (10,10) | 0.1405 + | **0.0754** | 0.0662 + | **0.0604** | 0.0419 + | **0.0405** | 0.0350 + | **0.0294** |
| | (10,20) | 0.1614 + | **0.0777** | 0.0876 + | **0.0665** | 0.0595 + | **0.0427** | 0.0390 + | **0.0320** |
| FDA5 | (10,10) | 0.0290 | **0.0267** | 0.0268 | **0.0267** | 0.0592 + | **0.0569** | 0.0687 + | **0.0411** |
| | (10,20) | 0.0308 + | **0.0266** | 0.0275 | **0.0267** | 0.0584 + | **0.0546** | 0.0676 + | **0.0400** |
| dMOP2 | (10,10) | 0.0208 + | **0.0170** | 0.0200 + | **0.0180** | 0.0239 + | **0.0216** | 0.0310 + | **0.0248** |
| | (10,20) | 0.0245 + | **0.0182** | 0.0222 + | **0.0194** | 0.0297 + | **0.0226** | 0.0451 + | **0.0259** |
| JY2 | (10,10) | 0.0494 + | **0.0330** | 0.0421 + | **0.0392** | 0.0461 + | **0.0433** | 0.0341 + | **0.0277** |
| | (10,20) | 0.0762 + | **0.0337** | 0.0543 + | **0.0401** | 0.0543 + | **0.0414** | 0.0480 + | **0.0295** |
| dMOP1 | (10,10) | 0.2352 + | **0.0801** | 0.1032 + | **0.0507** | 0.0312 | **0.0307** | 0.2104 + | **0.0852** |
| | (10,20) | 0.3320 + | **0.1182** | 0.1608 + | **0.0692** | 0.0360 + | **0.0327** | 0.2886 + | **0.1888** |
| HE2 | (10,10) | 0.0317 + | **0.0157** | 0.0176 + | **0.0137** | 0.0391 + | **0.0360** | 0.0368 + | **0.0220** |
| | (10,20) | 0.0385 + | **0.0178** | 0.0227 + | **0.0154** | 0.0420 + | **0.0338** | 0.0545 + | **0.0312** |
| SJY5 | (10,10) | 0.0083 + | **0.0073** | 0.0074 | **0.0071** | 0.0155 + | **0.0117** | **0.0349** | 0.0385 + |
| | (10,20) | 0.0091 + | **0.0078** | 0.0080 + | **0.0074** | 0.0190 + | **0.0123** | 0.0458 + | **0.0233** |

## CRediT authorship contribution statement

**Shaaban Sahmoud:** Conceptualization, Methodology, Software, Validation, Writing – original draft. **Haluk Rahmi Topcuoglu:** Conceptualization, Methodology, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## References

[1] P.A. Castillo, J.L.J. Laredo, Applications of Evolutionary Computation: 24th International Conference, EvoApplications 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings, vol. 12694, Springer Nature, 2021.
[2] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, A. Cosar, A survey on new generation metaheuristic algorithms, Comput. Ind. Eng. 137 (2019) 106040.
[3] I. Boussaïd, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Inf. Sci. 237 (2013) 82–117.
[4] K. Hussain, M.N. Mohd Salleh, S. Cheng, Y. Shi, Metaheuristic research: a comprehensive survey, Artif. Intell. Rev. 52 (2019) 2191–2233.
[5] C.-K. Goh, K.C. Tan, Evolutionary multi-objective optimization in uncertain environments: issues and algorithms, in: Studies in Computational Intelligence, vol. 186, 2009, pp. 5–18.
[6] C. Villa, E. Lozinguez, R. Labayrade, Multi-objective optimization under uncertain objectives: application to engineering design problem, in: International Conference on Evolutionary Multi-Criterion Optimization, Springer, 2013, pp. 796–810.
[7] Y. Jin, J. Branke, Evolutionary optimization in uncertain environments-a survey, IEEE Trans. Evol. Comput. 9 (3) (2005) 303–317.
[8] P. Rakshit, A. Konar, S. Das, Noisy evolutionary optimization algorithms–a comprehensive survey, Swarm Evol. Comput. 33 (2017) 18–45.
[9] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Trans. Evol. Comput. 8 (5) (2004) 425–442.
[10] Y. Tian, L. Si, X. Zhang, R. Cheng, C. He, K.C. Tan, Y. Jin, Evolutionary large-scale multi-objective optimization: a survey, ACM Comput. Surv. 54 (8) (2021) 1–34.
[11] K. Deb, U. Bhaskara Rao N., S. Karthik, Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling, in: International Conference on Evolutionary Multi-Criterion Optimization, Springer, 2007, pp. 803–817.
[12] S. Sahmoud, H.R. Topcuoglu, Exploiting characterization of dynamism for enhancing dynamic multi-objective evolutionary algorithms, Appl. Soft Comput. 85 (2019) 105783.
[13] M. Liu, J. Zheng, J. Wang, Y. Liu, L. Jiang, An adaptive diversity introduction method for dynamic evolutionary multiobjective optimization, in: Evolutionary Computation (CEC), 2014 IEEE Congress on, IEEE, 2014, pp. 3160–3167.
[14] H.G. Cobb, An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments, Tech. rep., DTIC Document, 1990.

[15] C.-K. Goh, K.C. Tan, A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization, IEEE Trans. Evol. Comput. 13 (1) (2009) 103–127.
[16] Y. Wang, B. Li, Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment, in: 2009 IEEE Congress on Evolutionary Computation, IEEE, 2009, pp. 630–637.
[17] Z. Peng, J. Zheng, J. Zou, M. Liu, Novel prediction and memory strategies for dynamic multiobjective optimization, Soft Comput. 19 (9) (2015) 2633–2653.
[18] S. Sahmoud, H.R. Topcuoglu, A memory-based NSGA-II algorithm for dynamic multi-objective optimization problems, in: European Conference on the Applications of Evolutionary Computation, Springer, 2016, pp. 296–310.
[19] Q. Zhang, X. He, S. Yang, Y. Dong, H. Song, S. Jiang, Solving dynamic multi-objective problems using polynomial fitting-based prediction algorithm, Inf. Sci. (2022).
[20] S. Jiang, S. Yang, A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization, IEEE Trans. Evol. Comput. 21 (1) (2017) 65–82.
[21] R. Liu, Y. Chen, W. Ma, C. Mu, L. Jiao, A novel cooperative coevolutionary dynamic multi-objective optimization algorithm using a new predictive model, Soft Comput. 18 (10) (2014) 1913–1929.
[22] F. Wang, F. Liao, Y. Li, H. Wang, A new prediction strategy for dynamic multi-objective optimization using gaussian mixture model, Inf. Sci. 580 (2021) 331–351.
[23] J. Sun, X. Gan, D. Gong, X. Tang, H. Dai, Z. Zhong, A self-evolving fuzzy system online prediction-based dynamic multi-objective evolutionary algorithm, Inf. Sci. 612 (2022) 638–654.
[24] H. Weizhen, M. Jiang, X. Gao, K.C. Tan, Y.-m. Cheung, Solving dynamic multi-objective optimization problems using incremental support vector machine, in: 2019 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 2794–2799.
[25] M. Jiang, W. Hu, L. Qiu, M. Shi, K.C. Tan, Solving dynamic multi-objective optimization problems via support vector machine, in: 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI), IEEE, 2018, pp. 819–824.
[26] M. Jiang, Z. Huang, L. Qiu, W. Huang, G.G. Yen, Transfer learning-based dynamic multiobjective optimization algorithms, IEEE Trans. Evol. Comput. 22 (4) (2017) 501–514.
[27] J.Y. Chia, C.K. Goh, V.A. Shim, K.C. Tan, A data mining approach to evolutionary optimisation of noisy multi-objective problems, Int. J. Syst. Sci. 43 (7) (2012) 1217–1247.
[28] P. Rakshit, A. Konar, S. Das, L.C. Jain, A.K. Nagar, Uncertainty management in differential evolution induced multiobjective optimization in presence of measurement noise, IEEE Trans. Syst. Man Cybern. Syst. 44 (7) (2013) 922–937.
[29] J.E. Diaz, J. Handl, Implicit and explicit averaging strategies for simulation-based optimization of a real-world production planning problem, Informatica 39 (2) (2015).
[30] A.N. Aizawa, B.W. Wah, Scheduling of genetic algorithms in a noisy environment, Evol. Comput. 2 (2) (1994) 97–122.
[31] J.E. Fieldsend, Elite accumulative sampling strategies for noisy multi-objective optimisation, in: International Conference on Evolutionary Multi-Criterion Optimization, Springer, 2015, pp. 172–186.
[32] J.J. Merelo, F. Liberatore, A.F. Ares, R. García, Z. Chelly, C. Cotta, N. Rico, A.M. Mora, P. García-Sánchez, There is noisy lunch: a study of noise in evolutionary optimization problems, in: 2015 7th International Joint Conference on Computational Intelligence (IJCCI), vol. 1, IEEE, 2015, pp. 261–268.
[33] J. Branke, C. Schmidt, H. Schmeck, Efficient fitness estimation in noisy environments, in: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, 2001, pp. 243–250.
[34] U. Hammel, T. Bäck, Evolution strategies on noisy functions how to improve convergence properties, in: International Conference on Parallel Problem Solving from Nature, Springer, 1994, pp. 159–168.
[35] H.-G. Beyer, Mutate large, but inherit small! On the analysis of rescaled mutations in es with noisy fitness data, in: International Conference on Parallel Problem Solving from Nature, Springer, 1998, pp. 109–118.
[36] E. Mininno, F. Neri, A memetic differential evolution approach in noisy optimization, Memetic Comput. 2 (2) (2010) 111–135.
[37] M. Helbig, A.P. Engelbrecht, Benchmarks for dynamic multi-objective optimisation algorithms, ACM Comput. Surv. 46 (3) (2014) 1–39.
[38] H. Richter, Detecting change in dynamic fitness landscapes, in: 2009 IEEE Congress on Evolutionary Computation, IEEE, 2009, pp. 1613–1620.
[39] L. Altin, H.R. Topcuoglu, Impact of sensor-based change detection schemes on the performance of evolutionary dynamic optimization techniques, Soft Comput. 22 (14) (2018) 4741–4762.
[40] S. Sahmoud, H.R. Topcuoglu, Sensor-based change detection schemes for dynamic multi-objective optimization problems, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2016, pp. 1–8.
[41] A. Ladi, J. Timmis, A.M. Tyrrell, P.J. Hickey, Statistical hypothesis testing for chemical detection in changing environments, in: 2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), IEEE, 2014, pp. 77–84.
[42] S. Sahmoud, H.R. Topcuoglu, Hybrid techniques for detecting changes in less detectable dynamic multiobjective optimization problems, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2019, pp. 1449–1456.
[43] T.F. Chan, G.H. Golub, R.J. LeVeque, Algorithms for computing the sample variance: analysis and recommendations, Am. Stat. 37 (3) (1983) 242–247.
[44] K. Deb, Multi-objective genetic algorithms: problem difficulties and construction of test problems, Evol. Comput. 7 (3) (1999) 205–230.
[45] S. Jiang, S. Yang, Evolutionary dynamic multiobjective optimization: benchmarks and algorithm comparisons, IEEE Trans. Cybern. 47 (1) (2017) 198–211.
[46] S. Jiang, S. Yang, A framework of scalable dynamic test problems for dynamic multi-objective optimization, in: Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on, IEEE, 2014, pp. 32–39.
[47] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, moea/d and NSGA-II, IEEE Trans. Evol. Comput. 13 (2) (2009) 284–302.
[48] M. Helbig, A.P. Engelbrecht, Dynamic multi-objective optimization using pso, in: Metaheuristics for Dynamic Optimization, Springer, 2013, pp. 147–188.
[49] J.R. Schott, Fault tolerant design using single and multicriteria genetic algorithm optimization, Tech. rep., DTIC Document, 1995.
[50] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.