Adaptive Multilevel Prediction Method for Dynamic Multimodal Optimization

Ali Ahrari*, *Member, IEEE* Saber Elsayed*, *Member, IEEE,* Ruhul Sarker*, *Member, IEEE,* Daryl Essam*, *Member, IEEE,* and Carlos A. Coello Coello+, *Fellow, IEEE*

***Abstract*—This study develops an adaptive multilevel prediction (AMLP) method to detect and track multiple global optima over time. First, it formulates a multilevel prediction approach in which a higher-level prediction improves the accuracy of the lower-level prediction to reduce the prediction error, enabling it to capture more complex patterns in the changes. However, a higher-level prediction is more sensitive to input errors and the randomness in the pattern of the change. To overcome this challenge, this study employs an adaptive mechanism which can determine the near-optimal prediction level at each time step. At the same time, AMLP calculates the strength of the diversity introduced after a change based on the estimated prediction error. A successful static multimodal optimizer is augmented with AMLP, for which AMLP determines the location and the mutation strength of the initialized subpopulations. An existing dynamic benchmark generator is improved so that it can generate dynamic test problems with more complex patterns in their changes. In particular, this dynamic benchmark generator allows for controlling the randomness of the pattern in the change to simulate dynamic problems with different degrees of predictability. A few controlled experiments are first performed to provide insight into different components of AMLP. Then, AMLP is compared with some of the most successful prediction methods when they are incorporated into the developed dynamic multimodal optimization method. Eleven dynamic cases with different change severity, change frequency, predictability, problem dimensionality, and the number of global minima are considered. The numerical results show the superiority of AMLP over other prediction methods.**

***Index Terms*—Evolutionary algorithm, changing environment, niching, dynamic test problem, change pattern**

I. Introduction

MANY real-world problems change over time. This change often occurs in the relationship between the objective function(s) and decision parameters; however, other aspects of the problem, such as constraints [1], [2], number of decision parameters [3], or even the number of objectives [4] may change over time. Hence, the optimal solution(s) to a dynamic problem change(s) over time. Dynamic optimization aims to detect and track the global optimum of dynamic problems. There has been a growing number of studies on the application of different search methods to practical dynamic optimization problems. Some recent examples are the optimal control of time-varying systems [5], [6], [7], communication systems [8], and vehicle routing scenarios [9], [10], [11].

*Authors are with the School of Engineering and Information Technology, University of New South Wales, ACT, Australia (E-mail: {a.ahrari; s.elsayed; r.sarker; d.essam}@unsw.edu.au, aliahrari1983@gmail.com)

+The author is with CINVESTAV-IPN, Departamento de Computación, Mexico City, Mexico (E-mail:: ccoello@cs.cinvestav.mx)

The simplest approach to optimize a dynamic problem is restarting the optimization process from scratch after a change. This approach is a reasonable choice when the change is so radical that the problem landscape after the change does not correlate with the problem landscape before it. However, in many classes of dynamic problems, the changes are not radical [12], and the problem landscape after the change resembles the one before it. Steadily changing problems [13] are good examples of dynamic problems in this class. The change in these problems is continual but smooth and gradual. For example, the demand for electricity changes with time and temperature, which gradually changes the load on the electricity grid during the day. Ideally, these problems should be optimized on-line; however, they are generally formulated as series of problem snapshots which are separated by a short time interval [13]. There are two main reasons for using this type of formulation [13], [14]:

- The optimized solution should be implemented in the system for an interval, which is referred to as the *implementing window* [13]. The length of this interval is application-dependent. For example, for power systems, the response time interval can be from a few seconds (for short-term operational flexibility) to several months (for long-term planning flexibility)[15].
- Most optimization algorithms require some time to provide an approximation of the optimal solution(s), during which they assume the problem does not change [13].

The formulated problem is thus stationary at each interval while the changes occur between two successive intervals. Shorter intervals (and thus less severe changes) makes the formulated problem a closer approximation of the actual one [13]. The formulated problem should be re-optimized in the limited time between two successive changes, the length of which is determined by the *change frequency*. Methods tailored for dynamic optimization may achieve this goal by exploiting information available from historical solutions, e.g. how the optimal solution has previously relocated in two successive time steps.

A dynamic optimization method can be developed by reinforcing a static search method with a dynamic mechanism. A variety of such strategies exist in the literature, such as memory-based methods [16], [17], which maintain diversity during the optimization process [18], [19] by introducing diversity after the change (e.g., by hypermutation [13] or random immigrants [20], [21]), multi-population methods [22], [23], [24], [25], and prediction methods [26]. A detailed

analysis of each strategy can be found in [27]. Prediction methods, in particular, have gained much interest in the realm of dynamic optimization [26]. They can be interpreted as a more robust variant of memory-based strategies that intend to predict the location of a global optimum, or the Pareto-optimal solutions in the case of dynamic multi-objective optimization. When compared with other types of strategies for handling dynamic aspects of a dynamic problem, prediction methods enjoy one prominent advantage: they are separate modules that only get activated when a change is detected. This modularity results in several benefits. First, they do not interfere with the static optimizer. Second, they can be easily combined with other static optimizers, for which there are already many well-developed methods. Third, they are activated only when a change is detected.

When the problem is multimodal, even small changes in the problem landscape can result in a substantial change in the global optimum because the depth of the optima varies over time, and thus, a previous global minimum may become a local one at a later time [28]. This will result in a huge movement of the global optimum from one step to the next one, which causes difficulties for a prediction method. Such a challenge can be effectively addressed if all the candidate local minima, which may potentially become a global one in future time steps, are tracked. This goal can be reached by multimodal optimization [29], [30], which aims to detect all good optima of the problem. Multimodal optimization is, therefore, an indispensable component of dynamic optimization, especially if prediction methods are to be used.

A more recent perspective to dynamic multimodal optimization (DMMO) aims at tracking not only one global optimum but all global and near-global ones over time [31], [32], [33]. This class of problems relates more closely to the concept of multimodal optimization. The dynamic multipath routing problem [32] is a good example: The problem of finding the best route to the destination may change because of an accident, heavy rain, or road maintenance. At the same time, there may be multiple (almost-) equally good paths to the destination in the new problem. If multiple of these paths are available, the user may select any of them based on some preferences that were not included in the formulation of the problem. Other examples of relevant real-world problems are solving a time-dependent system of equations [34], dynamic tracking of multiple targets [32], and Adaptive Information Filtering (AIF) [35]. These problems have been discussed in their corresponding publications in detail.

In spite of its practical importance, studies on DMMO are comparatively scarce, at least when compared to the number of studies on dynamic multi-objective optimization and dynamic constrained optimization. For example, several methods for dynamic multi-objective optimization have been recently developed by augmentation of a static multi-objective optimizer with a prediction method. This prediction method may track and predict the centroid of the non-dominated set [12], [36], [37], specific points of the non-dominated set [18], [38], [39], or even each Pareto optimal solution independently [40], [41]. Some of these predictions methods can be applied to DMMO problems as well.

A major drawback of most existing prediction methods is the fixed and limited utilization of past information. Thus, they can capture only simple patterns in the movement of global optima, such as a translation with fixed length and fixed direction. On the other hand, more complicated prediction models may capture more complicated patterns in the movement of the global optimum but they are more sensitive to input error randomness in the movement of the global optimum since they rely on older information.

This study intends to develop an approach which overcomes these drawbacks of existing techniques. This prediction method is expected to make the most out of hidden information in the pattern of each change and remains robust against potential randomness in those patterns. The contributions of this study are as follows:

- It proposes an adaptive multilevel prediction (AMLP) method in which a higher prediction level can approximate complex patterns more accurately but becomes more sensitive to the random movement of the global optimum. ALMP can learn the near-optimal prediction level at each time step.
- It develops a dynamic variant of the covariance matrix self-adaptation evolution strategy with repelling subpopulation (RS-DCMSA-ES) [42], [43] for DMMO. This method can accept an arbitrary prediction method as an independent module.
- It improves an existing test suite for DMMO so that it can simulate complex patterns in the movement of the global optimum that cannot be easily identified by a human observer. The test suite also allows for controlling the randomness in the change pattern.

The remainder of this paper is organized as follows: Section II discusses the previous related work. Section III formulates the adaptive multilevel prediction method. RS-DCMSA-ES and the improved dynamic benchmark generator are developed in Section V. Section VI shows some descriptive experiments that aim to provide insights on different aspects of AMLP. Section VII compares the performance of AMLP with some successful prediction methods in different dynamic settings. Finally, our conclusions are drawn in Section VIII.

II. Related work

This section briefly reviews prediction methods for dynamic optimization as well as relevant test suites for DMMO.

A. Prediction Methods

Prediction methods can be very efficient for dynamic problems in which the movement of the global optimum follows some patterns [44][38]. They exploit the past information gathered during the optimization process to predict the location of the global minimum in the new time step. This information is generally the location of the global best in the previous time steps. Using this information, a prediction method approximates the location of the global optimum in the new time step $(t+1)$:

$$\widehat{\boldsymbol{x}}_{t+1} = p(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t) \tag{1}$$

in which p is the prediction function and $\boldsymbol{x}_t$ is the location of the global optimum at time step t. Many prediction methods utilize information from a few recent locations of global optima, enabling them to capture simple patterns. For example, a simple but effective prediction method predicts the new translation vector first, which is then added to the previous location of the global optimum:

$$\widehat{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_t + \widehat{\boldsymbol{v}}_{t+1},\ \widehat{\boldsymbol{v}}_{t+1} = \boldsymbol{x}_t - \boldsymbol{x}_{t-1} \tag{2}$$

in which $\widehat{\boldsymbol{v}}_{t+1}$ is the prediction translation vector for time step $t+1$ and $\widehat{\boldsymbol{x}}_{t+1}$ is the predicted global minimum for time step $(t+1)$. This prediction method has been used in many studies with minor variations [38], [45], [41], [18], [19]. Despite its simplicity, it is an effective prediction model which can partially capture more complex but smooth change patterns.

Generally, there is a difference between the predicted and actual global minimum. This difference is the prediction error $\boldsymbol{e}_{t+1} = \boldsymbol{x}_{t+1} - \widehat{\boldsymbol{x}}_{t+1}$, the norm of which is generally greater than zero. There are two sources for this error [44]:

- Model error, which is the inability of the prediction model to capture the pattern underlying the movement of the global optimum.
- Input error, which is the inaccuracy of the data provided for the prediction function. For example, the optimization method could not detect the accurate location of the global optimum in the previous time steps.

Besides, there might be some randomness in the movement of the global optimum, which makes its movement at least partially unpredictable. In such problems, even the most sophisticated prediction method cannot accurately predict the new global optimum even though the exact global optima of the previous time steps are known. Consequently, each prediction method should also introduce some diversity to sample the seed population around $\widehat{\boldsymbol{x}}_{t+1}$. The strength of this introduced variation should be determined by the prediction method. If it is too strong, the optimization algorithm will take a long time to converge. Conversely, if it is too weak, the method should spend some time to diversify first, which will again take a long time to converge. Most existing prediction methods set the strength of this diversity proportional to the length of the predicted translation vector [41], [46], [47], [48].

More sophisticated prediction methods, such as Autoregression (AR) [49], may utilize more hidden information gathered during the optimization process, which enables them to capture more complex patterns in the movement of the global optimum. On the downside, it makes the prediction method more sensitive to input error, as well as the randomness in the change patterns.

As discussed earlier, prediction methods use the history of the optimum in the previous time steps. Another valuable element is the prediction error, which, surprisingly, has been ignored in most existing prediction methods. There are only a few studies that have used prediction error in previous time steps for prediction of $\widehat{\boldsymbol{x}}_{t+1}$. For example, Chen et al. [48] used the previous prediction error to predict the new efficient set for dynamic multi-objective optimization. The prediction error can be used to quantify accuracy of each prediction method when an ensemble of prediction methods is used [50]. Besides these two applications of the prediction error, this study demonstrates its importance for determining the strength of the introduced diversity after the change, as a more reasonable alternative to existing methods which define the strength of the introduced diversity proportionally to the length of the translation vector [41], [46], [47], [48].

B. Test Suites for DMMO

There are several recently proposed test suites for dynamic multi-objective optimization [3], [51], [52] and Dynamic constrained optimization [1], [53], [54]. In contrast, the most commonly used test suite for DMMO is a two-decade old moving peak benchmark (MPB) [55], [16]. This test suite forms peaks by superposition of spherical functions, where the depths, widths, and heights of the peaks change randomly over time. It is generally employed when the objective is to track only the best solution [56][28][33][57], which may jump from one peak to another. Later variants of this test suite allow for varying the number of optima over time [58], the inclusion of dynamic constraints [53], or the generation of problems with multiple equally good global maxima [31]. Nevertheless, the number of optima is limited because the simulation of each peak requires an independent function for superposition, which would increase the computational time proportionally. More importantly, the problem landscape consists of isolated spherical peaks. Therefore, these problems may not simulate the presence of a global structure in the problem nor ill-conditioned problems, which are two common features of global optimization [59].

The dynamic benchmark generator [DBG], developed by Li et al. [60], can simulate two different types of dynamic landscapes. In real rotation DBG (RRDBG), the problem landscape is similar to that of MPB; however, the dynamic change is simulated by a time-dependent rigid rotation of the search space. The more interesting variant, Real Composition DBG (RCDBG), superposes different types of basic functions borrowed from the global optimization literature instead of the simple spherical function. This change allows for the simulation of more diverse types of challenges, such as many undesirable local peaks and ill-conditioned problems. However, the dynamic behavior is simulated by a time-dependent shift vector per basic function. This benchmark generator was also used to create test problems for the CEC 2009 competition on dynamic optimization [61].

III. Adaptive Multilevel Prediction method

This section elaborates the proposed multilevel prediction method. First, the concept of prediction level is defined, and then, a simple strategy is developed to find the best prediction level at each time step.

A. Multilevel prediction

At this time, the (approximate) optimal solutions of the problem at time steps $0, 1, 2, \ldots, t$ are known, which are denoted by $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$. The objectives of the multilevel prediction method are:

- to predict $\boldsymbol{x}_{t+1}$ accurately so that the population seed for time step $t+1$ is generated close to $\boldsymbol{x}_{t+1}$. This prediction is denoted by $\widehat{\boldsymbol{x}}_{t+1}$.
- to define a proper strength for the introduced diversity for the seed population at the onset of time step $t+1$.

In the proposed multilevel prediction, each prediction level provides a prediction for $\boldsymbol{x}_{t+1}$, which is denoted by $\widehat{\boldsymbol{x}}_{t+1}^{(L)}$, in which 'L' refers to the level of prediction. Besides, each prediction level has its own prediction error $\boldsymbol{e}_{t+1}^{(L)} = \boldsymbol{x}_{t+1} - \widehat{\boldsymbol{x}}_{t+1}^{(L)}$.

The level one prediction is the simplest possible prediction method. It considers the last detected optimum ($\boldsymbol{x}_t$) as an estimate for the global optimum at time step $t+1$:

$$\widehat{\boldsymbol{x}}_{t+1}^{(1)} = \boldsymbol{x}_t, \boldsymbol{e}_{t+1}^{(1)} = \boldsymbol{x}_{t+1} - \widehat{\boldsymbol{x}}_{t+1}^{(1)} = \boldsymbol{x}_{t+1} - \boldsymbol{x}_t. \tag{3}$$

Note that if $\boldsymbol{e}_{t+1}^{(1)}$ is known, then $\boldsymbol{x}_{t+1}$ can be easily calculated as follows:

$$\boldsymbol{x}_{t+1} = \widehat{\boldsymbol{x}}_{t+1}^{(1)} + \boldsymbol{e}_{t+1}^{(1)}. \tag{4}$$

However, the calculation of $\boldsymbol{e}_{t+1}^{(1)}$ requires $\boldsymbol{x}_{t+1}$, which is not known at this time of the optimization process (at the onset of time step $t+1$). Instead, an estimate for $\boldsymbol{e}_{t+1}^{(1)}$ can be made using $\boldsymbol{e}_t^{(1)}, \boldsymbol{e}_{t-1}^{(1)}, \ldots, \boldsymbol{e}_1^{(1)}$, which are available at the onset of time step $t+1$. There are a variety of possible ways to use this information; however, we adopt the simplest one which only uses $\boldsymbol{e}_t^{(1)}$:

$$\widehat{\boldsymbol{e}}_{t+1}^{(1)} = \boldsymbol{e}_t^{(1)}, \tag{5}$$

in which $\widehat{\boldsymbol{e}}_{t+1}^{(1)}$ is the estimated level one prediction error. Using this estimate for $\boldsymbol{e}_{t+1}^{(1)}$ in equation 4, the Level 2 prediction for the optimum and the corresponding prediction error can be calculated:

$$\widehat{\boldsymbol{x}}_{t+1}^{(2)} = \widehat{\boldsymbol{x}}_{t+1}^{(1)} + \widehat{\boldsymbol{e}}_{t+1}^{(1)};\ \boldsymbol{e}_{t+1}^{(2)} = \boldsymbol{x}_{t+1} - \widehat{\boldsymbol{x}}_{t+1}^{(2)} \tag{6}$$

It is worth noting that $\widehat{\boldsymbol{x}}_{t+1}^{(2)}$ is equivalent to the commonly used prediction method described in equation 2. This can be easily concluded by combining equations 6 and 3:

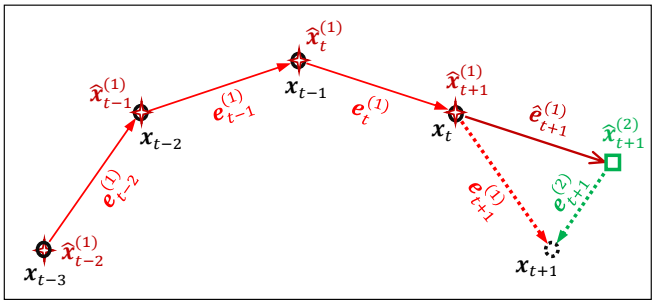
$$\widehat{\boldsymbol{x}}_{t+1}^{(2)} = \widehat{\boldsymbol{x}}_{t+1}^{(1)} + \widehat{\boldsymbol{e}}_{t+1}^{(1)} = \boldsymbol{x}_t + \boldsymbol{e}_t^{(1)} = \boldsymbol{x}_t + (\boldsymbol{x}_t - \boldsymbol{x}_{t-1}) \tag{7}$$

This procedure can be easily employed to calculate higher-level predictions for the optimum $\widehat{\boldsymbol{x}}_{t+1}^{(L)}$ and the estimated prediction error $\widehat{\boldsymbol{e}}_{t+1}^{(L)}$:

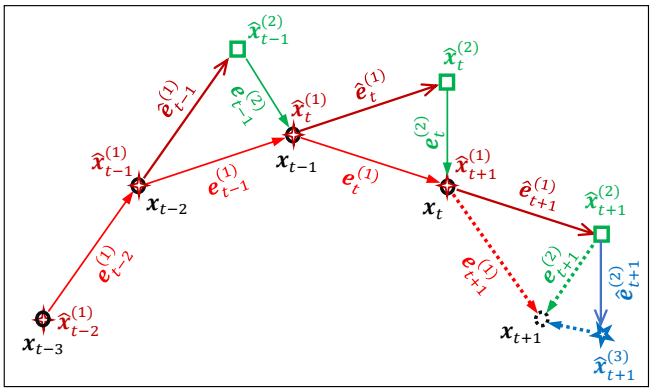
$$\widehat{\boldsymbol{e}}_{t+1}^{(L-1)} = \boldsymbol{e}_t^{(L-1)};\ \widehat{\boldsymbol{x}}_{t+1}^{(L)} = \widehat{\boldsymbol{x}}_t^{(L-1)} + \widehat{\boldsymbol{e}}_{t+1}^{(L-1)} \tag{8}$$

For example, Figure 1 illustrates how these vectors are calculated when $L = 3$. Figure 1a shows that Level 2 prediction has exploited the pattern in the prediction error of Level 1 to improve its prediction accuracy. Similarly, Figure 1b illustrates that Level 3 has learned and used the pattern in the prediction error of Level 2. Comparing the prediction error at different levels reveals that for this example, $\|\boldsymbol{e}_{t+1}^{(1)}\| > \|\boldsymbol{e}_{t+1}^{(2)}\| > \|\boldsymbol{e}_{t+1}^{(3)}\|$, indicating that Level 3 prediction is more accurate than level 2, which, in turn, is more accurate than Level 1 prediction.

Consequently, a higher-level prediction can capture complex patterns more accurately. However, we speculate that it is more sensitive to noise in the movement pattern of the optimum. Such noise can be caused by the dynamic problem itself or by the inability of the employed static search method to find the accurate location of the optimum in the previous time steps. Therefore, it is also important to explore the robustness of a prediction method against randomness in the movement pattern of the optimum. Another limitation of higher level predictions is that they can be calculated after a longer time. $\widehat{\boldsymbol{x}}_t^{(L)}$ and $\boldsymbol{e}_t^{(L)}$ can only be calculated if $t \geq L$.



(a)



(b)

Fig. 1: The predicted optimum and the corresponding prediction error at time step $t+1$ using a) Level 1 and Level 2 predictions, and b) Level 3 prediction

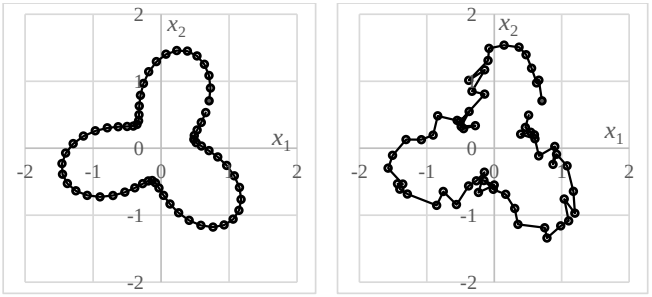
B. A Simple Simulation

The following prediction problem is considered to investigate this hypothesis:

$$\boldsymbol{x}_t = a(t)\ \mathbf{R}_0^t\ \boldsymbol{x}_0 + \sigma_{\mathrm{r}}\boldsymbol{r}$$
$$a(t) = 1 + 0.5\sin\left(\frac{\pi t}{10}\right),\ \boldsymbol{x}_0 = \frac{\mathbf{1}}{\sqrt{D}}, \tag{9}$$

in which $\boldsymbol{r}$ is a vector of independent and isotropic random numbers sampled from the standard normal distribution, σ_{r} specifies the strength of randomness in the movement pattern, $\mathbf{R}_0$ is a rigid rotation matrix which rotates $\boldsymbol{x}_0$ by the angle of $\alpha_0 = \frac{\pi t}{30}$. Figure 2 illustrates the location of the optimum for 60 time steps $(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{59})$.

When $\sigma_{\mathrm{r}} = 0$ (Figure 2a), the pattern in the movement of the optimum is smooth but not simple enough to be fully captured. The length of the translation vector and the rotation angle changes with time. Figure 2b depicts the movement pattern when $\sigma_{\mathrm{r}} = 0.1$. For this case, the randomness is too high to capture any useful pattern for prediction. Figure 3 illustrates the norm of the prediction error of different levels for different values of σ_{r} when $D = 10$. It reveals that:



(a) (b)

Fig. 2: Movement of the optimum over time in the simple test problem when a) $\sigma_r = 0$, and b) $\sigma_r = 0.1$

- The higher levels of prediction are more accurate when there is a smooth pattern in the movement of the optimum (σ_r= 0), even though this pattern is not simple to formulate.
- For moderate randomness in the movement of the optimum (σ_r= 0.001), prediction levels 3 and 5 can provide the most accurate prediction, rather than higher (levels 7, 9) and lower levels (level 1). For σ_r= 0.1, the lower the prediction level, the less the prediction error.
- Level L prediction can be used from time step L, since it requires the previous L-locations of the optimum. One simple strategy to address this issue is to use a lower level prediction for early time steps.

These observations reveal that a higher-level prediction can capture more complex patterns in the movement of the optimum, whereas a lower level prediction is more robust to irregular change patterns. Consequently, the optimal prediction level is problem-dependent. A single-level prediction method may not be a robust approach because it can be efficient only for specific types of dynamic problems. Furthermore, it is possible that a prediction level could be the optimal choice only for specific time steps of a dynamic problem. In such problems, an adaptive strategy that reliably determines the most promising level for each time step will surpass any fixed-level prediction method.

C. Adaptation Mechanism

The proposed strategy to overcome the limitation of a fixed level prediction is to use an adaptation strategy that selects the near-optimal level for prediction at the current time step $(t + 1)$. The best prediction level is not known at the onset of the current time step; however, the prediction error of each level can now be calculated for the previous time step. The prediction level with the least prediction error in time step t is deemed as the most reliable one for time step $t + 1$.

Algorithm 1 presents the proposed adaptive multilevel prediction (AMLP) method. Line 2 determines the highest prediction level that can be used, which is bounded by the size of data $(t + 1)$ and the user-defined parameter $L_{\max}$. Line 3 calculates t_{recent}. Locations of the solution $\boldsymbol{x}$ before time step t_{recent} are not needed for prediction; therefore, $\boldsymbol{e}_\tau^{(L)}$ and $\boldsymbol{x}_\tau^{(L)}$ are not calculated for $\tau < t_{\text{recent}}$. This can eliminate unnecessary calculations and improve time-complexity when $t \gg L_{\max}$. Lines 4-9 calculate $\boldsymbol{e}_\tau^{(L)}$ and $\boldsymbol{x}_\tau^{(L)}$ for level 1 prediction. Lines 10-19 calculate these values for higher level predictions (up to level $L'_{\max}$). It is worth mentioning that if $(L'_{\max} = t+1)$ and $t > 0$ (Line 20), the $\boldsymbol{x}_{t+1}^{(L'_{\max})}$ can be calculated and used for prediction; however, no estimation for prediction error can be provided since $\boldsymbol{e}_t^{(L'_{\max})}$ cannot be calculated. In this specific case, $\boldsymbol{e}_t^{(L'_{\max})}$ is set equal to $\boldsymbol{e}_t^{(L'_{\max}-1)}$ (Line 21).

The adaptation mechanisms triggers from line 20. If $t = 0$ (Only one time step has been finished), only $\boldsymbol{x}_1^{(L)}$ can be calculated, which is used for prediction (Lines 20-24). In this case, there is no estimate for the prediction error. For the general case of $t > 0$, the best prediction level is the one with the smallest prediction error norm at time step t (Line 25). Furthermore, if $L^* = L'_{\max} - 1$ and $L'_{\max} = t + 1$, then knowing that $\boldsymbol{e}_t^{(L'_{\max}-1)} = \boldsymbol{e}_t^{(L'_{\max})}$, the prediction level $L'_{\max}$ is equally good. In this case, AMLP favors the higher level prediction; therefore, it sets $L^* \leftarrow L^* + 1$ (Line 30). This preference can be helpful in early time steps when the predicted solution from level $L'_{\max}$ is available but there is no indicator for its accuracy; therefore, it will be always ignored if Line 30 is omitted. Finally, Lines 32 and 33 calculate the best predicted solution and the estimated error for this prediction given L^*.

To explore the importance of such an adaptive strategy, the previous example is extended to simulate time-dependent randomness in the movement of the optimum:

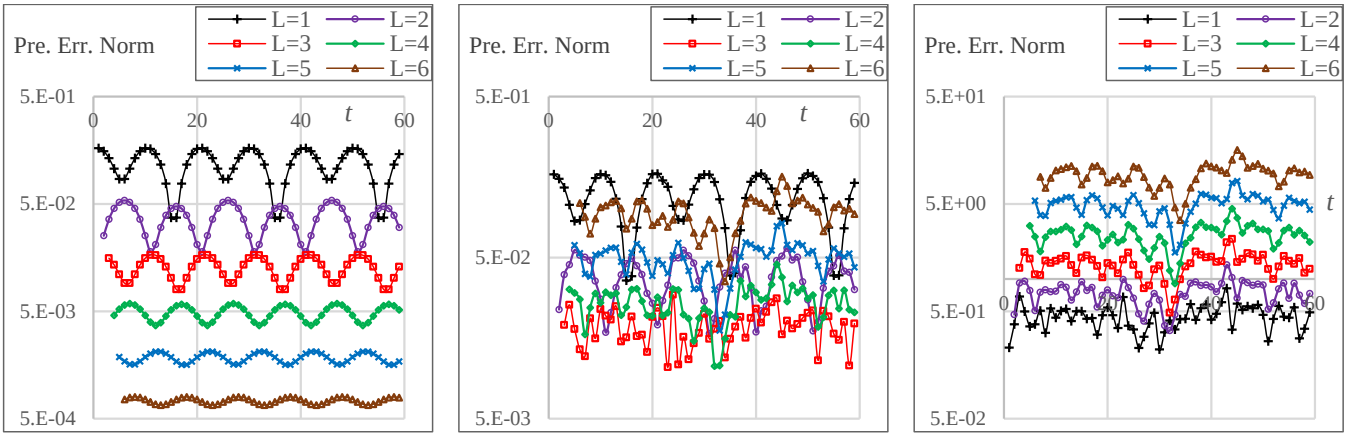
$$\sigma_r(t) = 0.001 \exp\left(5 \sin\left(\frac{\pi t}{30}\right)\right). \tag{10}$$

For this problem, the randomness in the moving pattern varies over time. It is negligible for certain time steps but dominant for some other time steps. For this example, $L_{\max} = 6$. Figure 4 shows the prediction error for each level when using a fixed level prediction (FLP) as well as the prediction error when an adaptive multilevel prediction (AMLP) is employed. This figure shows that the prediction error of the adaptive multilevel strategy is identical or very close to the smallest of all six levels. This example demonstrates that the adaptive strategy can detect the near-optimal prediction level on-the-fly. The code of AMLP in MATLAB® is provided in Supplementary Material **S1**.

D. Strength of the Introduced Diversity

The strength of the introduced diversity (s_{ID}) plays a critical role in the convergence process at each time step. As discussed in Section II, most existing prediction methods set $s_{\text{ID}_{t+1}} \propto \|\boldsymbol{v}_{t+1}\|$. However, this setting suffers from some theoretical issues. Figure 5a illustrated an example in which the prediction is relatively accurate (the norm of the prediction error is small). For this case, s_{ID} should be small since the center of the seed population ($\widehat{\boldsymbol{x}}_{t+1}$) is close to the new global optimum ($\boldsymbol{x}_{t+1}$). However, it will be large if s_{ID} is proportional to $\|\widehat{\boldsymbol{v}}_{t+1}\|$. In contrast, s_{ID} should be large for the case illustrated in Figure 5b, since the population center is far from the global optimum. For this case, the introduced diversity will be too weak.

Consequently, s_{ID} should be proportional to the norm of the prediction error, not to the length of the translation vector. The



(a) $\sigma_r = 0$ (b) $\sigma_r = 0.001$ (c) $\sigma_r = 0.1$

Fig. 3: Norm of the prediction error of different prediction levels (L) for the sample prediction problem with three different values of σ_r

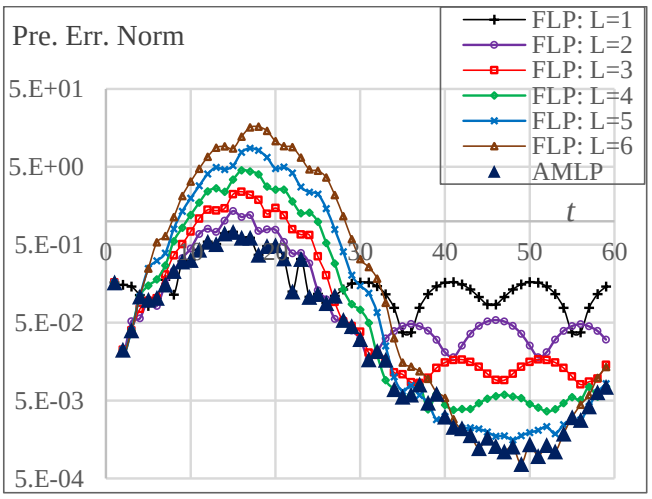
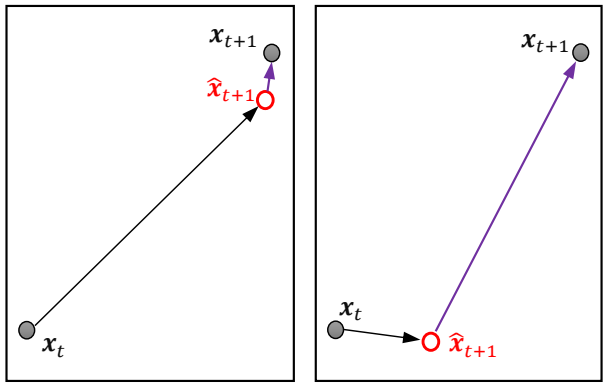


Fig. 4: Prediction error of the adaptive multilevel prediction (AMLP) and the fixed level prediction (FLP) method with different levels (L) for the defined test problem with variable randomness in the change pattern



(a) (b)

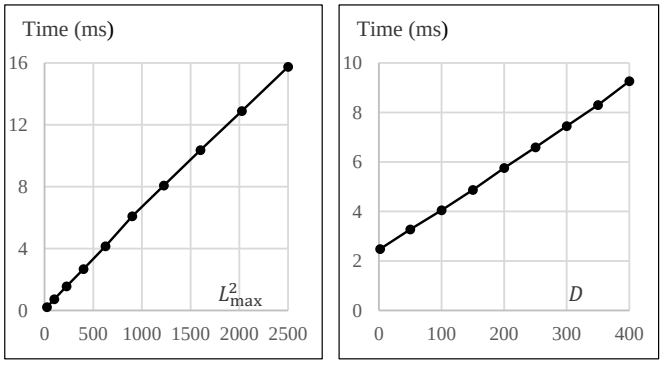
Fig. 5: Two sample cases and the outcome of the prediction in each case. a) The norm of the translation vector is large, but the prediction error is small. b) The norm of the translation vector is small, but the prediction error is large

proposed heuristic in this study sets s_{ID} proportionally to the norm of the estimated prediction error:

$$s_{ID_{t+1}} \propto \|\hat{e}_{t+1}\| = \|e_t\| \tag{11}$$

E. Time Complexity

AMLP does not require any additional function evaluations, nor does it involve any computationally expensive operators. Figure 6 shows the required computational time by AMLP for predicting the new location of a point given its locations in previous time steps. This figure shows that the time required by AMLP is proportional to the problem's dimension (D) and L_{max}^2. For typical values of $D = 10$ and $L_{max} = 20$, the required time is only 0.0027 second. This simulation was performed in a MATLAB R2017b environment using a personal laptop with 8 GB RAM and i7-8550U CPU (only one core was used for the simulation).



(a) (b)

Fig. 6: Computational time required for prediction by the proposed AMLP method as a function of a) the highest prediction level (L_{max}) when $D = 10$, and b) the problem's dimensionality (D) when $L_{max} = 20$

Algorithm 1: Adaptive Multilevel Prediction (AMLP) method.

Data: Maximum prediction level ($L_{\max}$); A time histroy of the solution $\boldsymbol{x}$ denoted by ($\mathbf{S_x} = \{\boldsymbol{x}_0, \boldsymbol{x}_1, ...\boldsymbol{x}_t\}$).
Result: The best prediction level (L^*), predicted solution for time step $t+1$ ($\widehat{\boldsymbol{x}}_{t+1}$), estimated prediction error for this prediction ($\widehat{\boldsymbol{e}}_{t+1}$)

Initialization
$L'_{\max} \leftarrow \min\{L_{\max}, t+1\}$
$t_{\text{recent}} = \max\{0, t - L_{\max}\}$
for $\tau = t_{\text{recent}}$ **to** t **do**
  $\widehat{\boldsymbol{x}}^{(1)}_{\tau+1} \leftarrow \boldsymbol{x}_\tau$
  if $\tau < t$ **then**
    $\boldsymbol{e}^{(1)}_{\tau+1} \leftarrow \boldsymbol{x}_{\tau+1} - \widehat{\boldsymbol{x}}^{(1)}_{\tau+1}$
  end
end
for $L = 2$ **to** $L'_{\max}$ **do**
  for $\tau = t_{\text{recent}}$ **to** t **do**
    if $\tau \geq L - 1$ **then**
      $\widehat{\boldsymbol{x}}^{(L)}_{\tau+1} \leftarrow \widehat{\boldsymbol{x}}^{(L-1)}_{\tau+1} + \boldsymbol{e}^{(L-1)}_{\tau}$
      if $\tau < t$ **then**
        $\boldsymbol{e}^{(L)}_{\tau+1} \leftarrow \boldsymbol{x}_{\tau+1} - \widehat{\boldsymbol{x}}^{(L)}_{\tau+1}$
      end
    end
  end
end
if $(L'_{\max} = t+1)$ **and** $(t > 0)$ **then**
  $\boldsymbol{e}^{(L'_{\max})}_{t} \leftarrow \boldsymbol{e}^{(L'_{\max}-1)}_{t}$
end
if $t = 0$ **then**
  $L^* \leftarrow 1$
  $\widehat{\boldsymbol{x}}_{t+1} \leftarrow \widehat{\boldsymbol{x}}^{(L^*)}_{t+1}$
  $\widehat{\boldsymbol{e}}_{t+1} \leftarrow \{\}$
else
  $L^* \leftarrow \text{argmin}_L\{\|\boldsymbol{e}^{(L)}_t\|\}$
  if $(L^* = L'_{\max} - 1)$ **then**
    $L^* \leftarrow L^* + 1$
  end
  $\widehat{\boldsymbol{x}}_{t+1} \leftarrow \widehat{\boldsymbol{x}}^{(L^*)}_{t+1}$
  $\widehat{\boldsymbol{e}}_{t+1} \leftarrow \boldsymbol{e}^{(L^*)}_{t}$
end

IV. RS-DCMSA WITH AMLP

This section integrates AMLP into the covariance matrix self-adaptation evolution strategy with repelling subpopulations (RS-CMSA), one of the most recent successful methods for static multimodal optimization [42], [43].

A. Brief description of RS-CMSA-ES

In RS-CMSA-ES, N_s subpopulations ($\mathbf{P}_k, k = 1, 2, \ldots, N_s$) search for the global optima in parallel. Those solutions that are deemed global minima are stored in an external archive denoted by **Archive**. RS-CMSA-ES encourages each subpopulation to converge to a different global minimum by defining taboo regions for each subpopulation and each archived solution. These taboo distances are adapted after each restart, and a new restart is performed when the subpopulation size is multiplied by c_{popSize}. This process continues until the evaluation budget is depleted.

B. Input Data for Prediction

Data provided for the prediction method is the solutions stored at **Archive** at the end of each time step (**ARCHIVE** = {**Archive** (0) , **Archive** (1) **, . . . ,Archive**(t)}). To be usable for prediction methods, the archived solutions in previous time steps that correspond to each solution in **Archive** (t) ($\boldsymbol{y}^{(t)}_m, m = 1, 2, \ldots, M$) should be determined, in which M specifies the number of solutions in **Archive** (t). This is required to form a history of locations of a specific optimum. This correspondence is determined based on the minimum Euclidean distance in the decision variable space, a strategy that has also been used for prediction for dynamic multi-objective problems [42]. Accordingly, for each $\boldsymbol{y}^{(t)}_m$, the corresponding solution in **Archive** $(t-1)$ is the nearest one to $\boldsymbol{y}^{(t)}_m$. Similarly, the nearest solution in **Archive** $(t-2)$ is the one that corresponds to $\boldsymbol{y}^{(t-1)}_m$. This process continues until a time history of $\boldsymbol{y}^{(t)}_m$ is identified and stored in a series denoted by $\mathbf{S}_m$, which ideally, but not necessarily, represents the movement of a specific optimum over time. This series is determined for each $\boldsymbol{y}^{(t)}_m \in$ **Archive** (t), resulting in M series of solutions ($\mathbb{S} = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_M\}$), which are provided for the prediction method.

C. Initialization using AMLP

In dynamic problems, the evaluation budget may be very limited; therefore, for RS-DCMSA-ES, $N_s = 1$. This means that at each restart, only one subpopulation evolves until it terminates. The seed population at each restart in time step $t+1$ may be initialized quasi-randomly or using AMLP, which are discussed in this section.

For the first M restarts in time step $t+1$, the parameters of the initial population are defined using AMLP. First, the predicted location of the optimum at time step $t+1$ and the estimate for the prediction error are calculated:

$$(\widehat{\boldsymbol{x}}_{m,t+1}, \widehat{\boldsymbol{e}}_{m,t+1}) = AMLP(\mathbf{S}_m)$$

Then, these two vectors are used to initialize the seed subpopulation:

$$\boldsymbol{x}_{\text{mean}_m} = \widehat{\boldsymbol{x}}_{m,t+1},\ \mathbf{C}_m = \mathbf{I}_D$$
$$s_{\text{mean}_m} = \begin{cases} k_{s_{\text{mean}}} \left\|\widehat{\boldsymbol{e}}_{m,t+1}\right| & \text{if } t = 0 \\ \dfrac{0.125}{D} \displaystyle\sum_{i=1}^{D} \left(X^{\text{U}}_i - X^{\text{L}}_i\right) & \text{if } t > 0 \end{cases} \tag{12}$$
$$m = 1, 2, \ldots, M,$$

in which $k_{s_{\text{mean}}}$ is a fixed number and I_D is the identity matrix of size D.X^{U}_i and X^{L}_i are the upper and lower limit of the i^{th} decision parameter. $\boldsymbol{x}_{\text{mean}_m}$, s_{mean_m}, and $\mathbf{C}_m$ are the center, mutation strength, and covariance matrix of the

initialized subpopulation, respectively. When $t = 0$, $\widehat{\boldsymbol{e}}_{m,t+1}$ is not available; therefore, for this specific time step, the mutation strength is determined using an alternative approach.

For the next M restarts in time step $t+1$, the subpopulation $\mathbf{P} = \{\boldsymbol{x}_{\text{mean}_m}, s_{\text{mean}_m}, \mathbf{C}_m\}$ is initialized using the default quasi-random initialization strategy of RS-CMSA (no prediction is used in this case). For the subsequent restarts, the realization strategy switches after M restarts alternately. Therefore, the initialization process is prediction-based for restarts in the range $[2kM+1, (2k+1)M]$, $(k = 0, 2, 4, \dots)$, which aims to maximize the exploitation of past information. In contrast, for $k = 1, 3, 5, \dots$, the initialization process is quasi-random which intends to maximize exploration. The latter goal is of particular importance if the dynamic optimization could not have detected some of the global minima in the previous time steps. The only exceptions are for time steps zero and one. For $t = 0$, there is no past information to use, and thus, the subpopulation is always initialized using the quasi-random method. For $t = 1$, the prediction based initialization is used only to generate $\boldsymbol{x}_{\text{mean}_m}$ while s_{mean_m} and $\mathbf{C}_m$ are still determined using the quasi-random method since the prediction error cannot be estimated. The flowchart of RS-DCMSA, when augmented with a prediction method, is provided in Fig. 7.

V. Dynamic Multimodal Benchmark Generator

This section extends RCDBG [60] to explore the potential of different prediction methods. In particular, this extension combines simple dynamic shift of the global optima with the rigid rotation of RRDBG [60]. Besides, it introduces a parameter for controlling the randomness in the change pattern.

A. General formulation

The formulation of the developed dynamic benchmark generator is as follow:

$$F(\boldsymbol{x}, t) = \min_i \left\{ f\left(\frac{\mathbf{R}(t)\,\boldsymbol{x} - \boldsymbol{O}_i(t)}{\lambda_i}\right)\right\} + h(t) \qquad i = 1, 2, \dots, N_{\text{gmin}}. \tag{13}$$

In this equation, $f(\boldsymbol{x})$ is a basic unimodal or multimodal function whose global minimum is $\mathbf{0}$, $\mathbf{R}(t)$ is a time dependent rigid rotation matrix, $\boldsymbol{O}_i$ is the dynamic shift vector for the i^{th} global minimum, λ_i are the scaling factors, $h(t)$ shifts the global minimum value, and N_{gmin} defines the number of global minima in the problem. The global minima of $F(\boldsymbol{x}, t)$ are

$$\boldsymbol{x}^*(t) = \mathbf{R}^T(t)\ \boldsymbol{O}_i(t),\ i = 1, 2, \dots, N_{\text{gmin}} \tag{14}$$

The search range is symmetric about the origin. To make sure that they remain within the search range after rotation, $\boldsymbol{O}_i(t)$ must remain inside the sphere inscribed by the search space.

The dynamic shift vectors are determined as follows: First, for each global minimum, two random points $\boldsymbol{p}_{i1}$ and $\boldsymbol{p}_{i2}$ are generated on the sphere inscribed in the search space. Then, $\boldsymbol{O}_i(t)$ is calculated as follows:

$$\boldsymbol{O}_i(t) = \boldsymbol{O}_i(0) + 0.5 \sin\left(\frac{2\pi t + r}{n_{ti}}\right)\boldsymbol{u}_i, r \sim \mathcal{N}\left(0, \sigma_{\text{r}}^2(t)\right) \tag{15}$$

in which $\boldsymbol{O}_i(0) = 0.5\left(\boldsymbol{p}_{i2} + \boldsymbol{p}_{i1}\right)$ and $\boldsymbol{u}_i = 0.5\left(\boldsymbol{p}_{i2} - \boldsymbol{p}_{i1}\right)$ are the shift vector at time step zero and the shift range vector for the i^{th} global minima, respectively (see Figure 8 for an example). $\sigma_{\text{r}}(t)$ is the standard deviation of randomness in the change pattern. r is sampled anew for each time step, and n_{ti} is the severity of the change of $\boldsymbol{O}_i(t)$. A reasonable choice is $n_{ti} = n_t$. However, in this study, a slightly different n_{ti} is used for each global minimum so that the relative locations of global minima do not return to the same distribution after n_t time steps:

$$n_{t_i} = \overline{n}_t \exp\left(\frac{i-1}{N_{\text{gmin}} - 1} - 0.5\right), \tag{16}$$

in which $\overline{n}_t$ is the average change severity. λ_i are selected using the same idea:

$$\boldsymbol{r_0} = randperm\left(N_{\text{gmin}}\right) \qquad \lambda_i = \exp\left(\frac{\tau_\lambda}{D}\left(\frac{r_{0_i} - 1}{N_{\text{gmin}} - 1} - 0.5\right)\right), \tag{17}$$

in which τ_λ controls the ratio of the largest to the smallest scaling factor, and $randperm(N_{\text{gmin}})$ returns a random permutation of natural numbers from 1 to N_{gmin} (inclusive). Performing this random permutation is recommended since otherwise the smallest basin will always be the one with the smallest n_{ti}.

To allow the test suite to simulate dynamic problems with variable randomness in the change pattern, $\sigma_{\text{r}}(t)$ is defined as follows:

$$\sigma_{\text{r}}(t) = \overline{\sigma}_{\text{r}} \exp\left(\sigma_{\sigma_{\text{r}}} \sin\left(\frac{\pi t}{\overline{n}_t}\right)\right), \tag{18}$$

in which $\overline{\sigma}_{\text{r}}$ and $\sigma_{\sigma_{\text{r}}}$ are the mean and standard deviation for the randomness in the movement of the minima. Finally, the offset in the objective value and the dynamic rotation matrix are defined as follows:

$$h(t) = 100 \sin\left(\frac{3\pi t}{n_t}\right);\ \mathbf{R}(t) = \mathbf{R}_0^t, \tag{19}$$

in which $\mathbf{R}_0$ is a randomly generated rigid rotation matrix which rotates the search space by the angle of $2\pi/\overline{n}_t$.

Remarkably, the movement of the global minima consists of oscillation along a randomly selected chord of the sphere plus a rigid rotation. This simulates relatively complex change patterns.

B. Test problems

The considered test problems are summarized in Table I. The small number of test problems allows for studying the effect of decisive factors, such as problem's dimensionality and N_{gmin} on the whole test suite. The definitions of the employed basic functions are provided in the Supplementary Material **S2**.

C. Performance Indicator

For multimodal optimization, the peak ratio (PR) is a commonly used indicator, which can be calculated at the end or during the optimization process. A reasonable approach to employ this indicator for dynamic problems is to average the

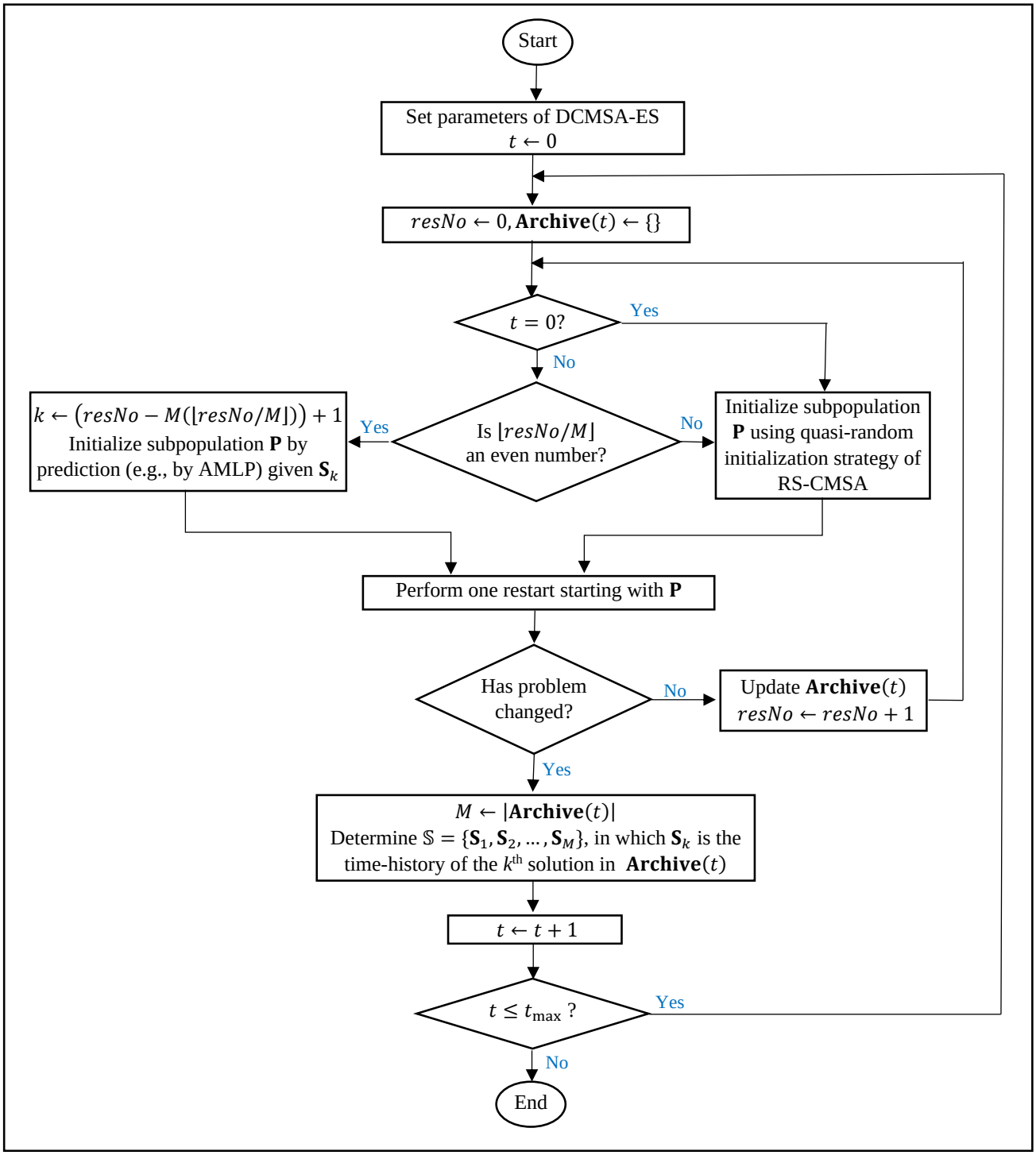


Fig. 7: Flowchart of RS-DCMSA

TABLE I: Description of the considered test problem. For all these problems, the search range is $[-5, 5]^D$.

Dynamic Function	Basic Function
$F_1(\boldsymbol{x}, t)$	Rotated Rastrigin
$F_2(\boldsymbol{x}, t)$	Rotated Ackley
$F_3(\boldsymbol{x}, t)$	Rotated Zakharov
$F_4(\boldsymbol{x}, t)$	Rotated Rosenbrock
$F_5(\boldsymbol{x}, t)$	Rotated Elliptic
$F_6(\boldsymbol{x}, t)$	Rotated Levy

PR at the end of each time step, as employed in [31]; however, this definition ignores when in the time steps the optima were detected. An optimization method that can identify optima faster requires fewer function evaluations, which is particularly essential for dynamic optimization because the time between two changes is limited.

Another performance indicator for multimodal dynamic optimization in this paper is as follows: For time step t, $PR(t, FE)$ denotes the peak ratio after FE solution evalu-

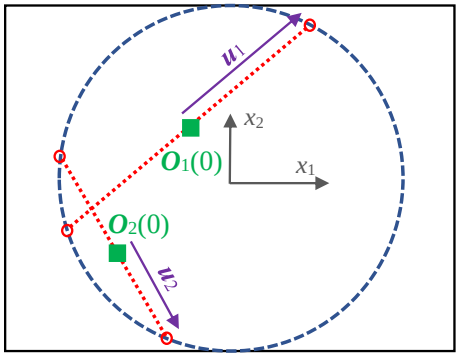


Fig. 8: Shift vector at time step zero ($\boldsymbol{O}_i(0)$) and the change extent vector ($\boldsymbol{u}_i$) for an exemplary problem with two global minima.

ations in that time step, given the target function tolerance of $\epsilon_f = 0.001$. Then, for time step t, the dynamic peak ratio ($0 \leq DPR(t) \leq 1$) is calculated as follows:

$$DPR(t) = \frac{1}{\tau_t} \sum_{FE=1}^{\tau_t} PR(t, FE). \tag{20}$$

A higher DPR denotes that the optimization method could detect more optima faster. The mean DPR is calculated by averaging DPR of each time step:

$$MDPR = \left(\frac{1}{t_{\max} - t_0 + 1} \right) \sum_{t=t_0}^{t_{\max}} DPR(t). \tag{21}$$

The initial time steps (time steps 0, 1, ..., $t_0 - 1$) were excluded in the calculation of MDPR as they were deemed the training period for the prediction method. MDPR with $t_0 = 20$ is then employed to compare the performance of different methods.

VI. Descriptive Experiments

Two descriptive experiments are performed in this section to provide insights into different components of AMLP. In this case, an ideal scenario is considered, in which the global optima and their correct correspondence in the previous time steps (time steps 0, $1, \ldots, t$) are provided for the prediction method. The prediction method is then used to initialize the subpopulation(s) for time step $t+1$. The ideal scenario allows for a more reliable comparison of different prediction methods when there is no input error. For the rest of this article, the default setting of $L_{\max} = 20$ is used for AMLP unless mentioned otherwise. For the experiments in this section, a maximum of 60 time steps are considered for each problem ($t_{\max} = 59$).

A. Strength of the Introduced Diversity

The introduced diversity after a change plays a significant role in the convergence rate towards the global optimum. As argued in Section III-D, the strength of this diversity should be proportional to the prediction error norm, and not to the length of the predicted translation vector. To check the importance of this issue, AMLP is employed with the following strategies:

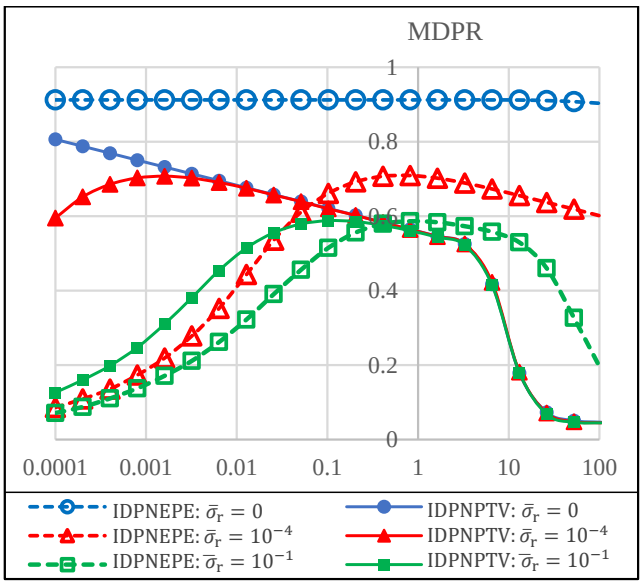


Fig. 9: MDPR as a function of $k_{s_{\text{mean}}}$ using two different strategies for setting the introduced diversity after the change. Results are for three dynamic problems with different amounts of randomness in the changes.

- Introduced diversity proportional to the norm of the predicted translation vector (IDPNPTV): $s_{\text{mean}} = k_{s_{\text{mean}}} \|\widehat{\boldsymbol{v}}_{t+1}\|$
- Introduced diversity proportional to the norm of the estimated prediction error (IDPNEPE): $s_{\text{mean}} = k_{s_{\text{mean}}} \|\widehat{\boldsymbol{e}}_{t+1}\|$

Figure 9 shows MDPR for these two strategies versus $k_{s_{\text{mean}}}$ for three selected values of $\overline{\sigma}_r$, representing dynamic problems with no, moderate and severe patternless changes. This figure shows that the optimal value of $k_{s_{\text{mean}}}$ strongly depends on the randomness in the movement pattern if IDPNPTV is employed. For a greater $\overline{\sigma}_r$, $k_{s_{\text{mean}}}$ should be greater. Therefore, there is no unique $k_{s_{\text{mean}}}$ that can work efficiently for IDPNPTV. In contrast, any $k_{s_{\text{mean}}} \in [0.4, 1.6]$ is a (near-) optimal choice for all values of $\overline{\sigma}_r$ if IDPNEPE is employed.

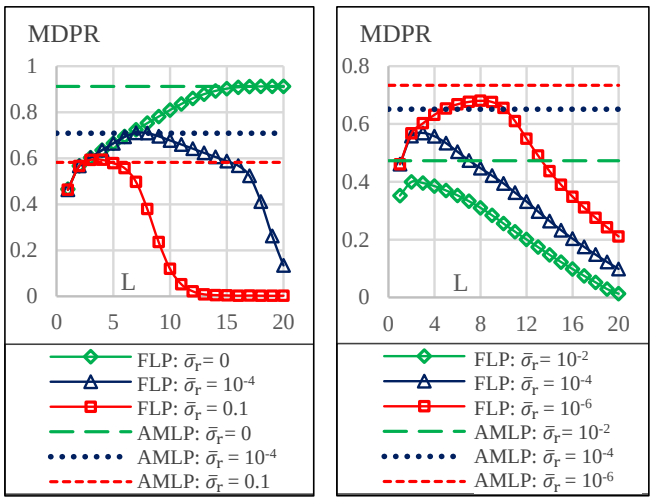
B. Adaptation Capability of AMLP

The performance of the fixed level prediction (FLP) with different levels is compared with that of AMLP. The objective is to explore whether AMLP can find the optimal prediction level, which depends on the randomness. For this experiment, $F_1(\boldsymbol{x}, t)$ is considered with $D = 10, \tau_t = 40000, \overline{n}_t = 40$. Two cases are studied:

- Fixed randomness: $\sigma_{\sigma_r} = 0, \overline{\sigma}_r = 0, 10^{-4}, 0.1$
- Variable randomness: $\sigma_{\sigma_r} = 10, \overline{\sigma}_r = 10^{-2}, 10^{-4}, 10^{-6}$

Figure 10 illustrates MDPR as a function of the prediction level when FLP is used, as well as MDPR of AMLP. Results for both cases are provided. This figure reveals that:

- For a fixed amount of randomness, the optimal prediction level depends on the randomness in the movement pattern of the global minima. When the randomness intensifies, the lower levels are more successful.



(a) Fixed randomness (b) Variable randomness

Fig. 10: MDPR for the fixed level prediction (FLP) method with different prediction levels (L) as well as AMLP for $F_1(\boldsymbol{x}, t)$ in the ideal scenario

- When randomness is fixed, the performance of AMLP is similar to the performance of FLP with the optimal level for all the tested values of $\overline{\sigma}_r$.
- When the randomness varies over time, AMLP outperforms FLP with any value for the prediction level.

Consequently, a fixed level prediction method can be neither robust nor efficient when dynamic problems with different randomness in the movements of the global optima are considered. The AMLP method can learn the optimal prediction level on-the-fly and choose its outcome for prediction and determining the strength of the introduced diversity.

VII. Numerical Comparison

This section compares the performance of AMLP with some of the most successful prediction methods in the literature. All these prediction methods are incorporated into RS-DCMSA to suppress the effect of the static optimizer. Besides, for all the prediction methods, the introduced diversity is proportional to the estimated prediction error (IDPNEPE), which was motivated by findings in Section VI-A. The changes are assumed to be informed in order to suppress the difference that may be caused by the change detection mechanism. This means that the optimization process is notified when a change occurs. Consequently, the gap between the performance of the tested methods solely originates from the difference in the prediction methods. All the test problems are optimized in real scenarios, which means that the prediction method has access only to the archive at previous time steps.

A. Selected Methods

The following prediction methods were selected for comparison:

- PRE: This prediction method was originally developed for dynamic multi-objective optimization [41]. The same idea has been employed in a recent study [40] as well. PRE simply calculates the translation vector as the difference between a global optimum in the two last time steps:

$$\widehat{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_t + \widehat{\boldsymbol{v}}_{t+1}, \widehat{\boldsymbol{v}}_{t+1} = \boldsymbol{x}_t - \boldsymbol{x}_{t-1},$$

  PRE is equivalent to the prediction level being fixed at level two.
- Autoregression (AR): The autoregressive (AR) model, developed by Schneider and Neumaier [49], is a well-known prediction method. It has been successfully employed in the feed-forward prediction strategy [44], as well as in several subsequent prediction methods [45], [46]. The univariate AR model, which assigns an independent model to each variable, is selected for comparison. This preference is based on its advantages over the multivariate vector autoregressive model [44].
- Decomposition-based difference model (D-DM) [12]: This prediction method utilizes the location of the global optimum in the last three time steps:

$$\widehat{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_t + \widehat{\boldsymbol{v}}_{t+1}$$

$$\widehat{\boldsymbol{v}}_{t+1} = 2\,(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}) - (\boldsymbol{x}_{t-1} - \boldsymbol{x}_{t-2})$$

None of these predictions methods require additional function evaluations nor involve any time-consuming operators. Therefore, the time required for prediction is negligible when compared with the whole optimization process.

B. Settings for the Test Problems

Different aspects of the generated test problems can be controlled by tuning parameters of the test suite such as D, N_{gmin}, $\overline{n}_t$, $\overline{\sigma}_r$, σ_{σ_r}. Except for D, these aspects are not known a priori. Therefore, the overall performance of a DMMO method in all the settings should be regarded when comparing different strategies or methods. In this study, a base case is defined, and the effect of each aspect is studied by changing one tunable parameter at a time, resulting in eleven cases of six test problems. Table II presents the settings used for each case. For all cases, $\sigma_{\sigma_r} = 0$, $t_{\max} = 100$, and the first change occurs after $10000DN_{\text{gmin}}$ evaluations.

C. Results and Discussion

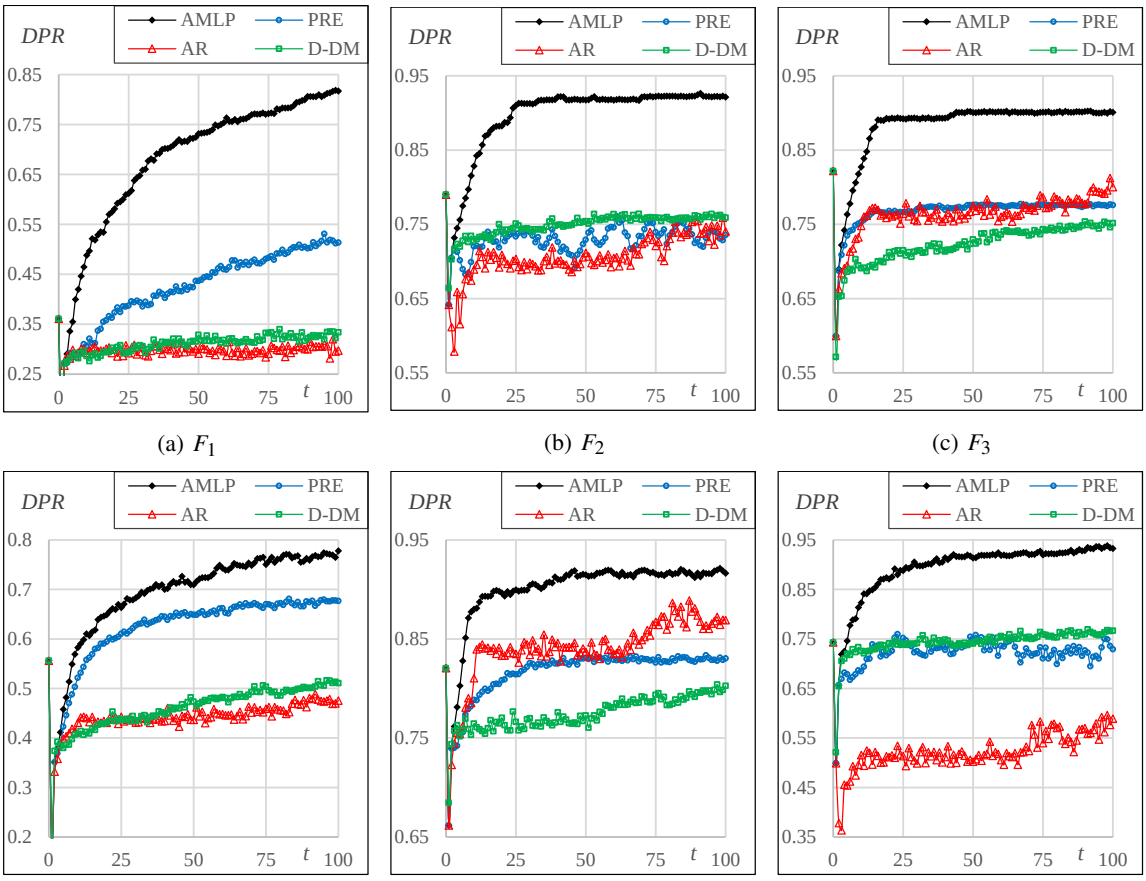
Since the test problems have some randomness, each problem is instantiated 20 times with seeds 1-20. Each prediction method is tested once on each instantiated problem, and the performance indicator is averaged over these 20 runs.

Table III presents the average MDPR, which is obtained by calculating the average MDPR of the six test problems in each case. The standard error of the average MDPR is also provided. MDPR and the corresponding standard error for individual test problems are provided in the supplementary material **S3**. For Case I, the history of DPR for each problem is provided in Figure 11. The obtained results are summarized below:

- Figure 11 shows that DPR gradually improves over time when AMLP is used for prediction. This trend is particularly observable for F_1. Two factors account for this improvement: First, a higher prediction level can be used

TABLE II: Settings for the parameters of the dynamic multimodal test problems

Case	D	N_{gmin}	n_t	$\overline{\sigma}_{\text{r}}$	$\tau_t/(DN_{\text{gmin}})$	Problem Aspect Studied
I	10	10	40	0	1000	Base setting
II	5	10	40	0	1000	Problem Dimension
III	20	10	40	0	1000	Problem Dimension
IV	10	5	40	0	1000	No. of global minima
V	10	20	40	0	1000	No. of global minima
VI	10	10	20	0	1000	Change severity
VII	10	10	80	0	1000	Change severity
VIII	10	10	40	0.0001	1000	Randomness in the pattern of the change
IX	10	10	40	0.1	1000	Randomness in the pattern of the change
X	10	10	40	0	200	Change Frequency
XI	10	10	40	0	5000	Change Frequency



(a) F_1 (b) F_2 (c) F_3

(d) F_4 (e) F_5 (f) F_6

Fig. 11: DPR of the tested prediction method when incorporated to RS-DCMSA-ES for each problem in Case I.

TABLE III: MDPR ± standard error for the tested prediction methods. The values were calculated over six test problems in each case.

Case	Employed Prediction Method			
	AMLP	PRE	AR	D-DM
I	0.852 ± 0.0040	0.695 ± 0.0042	0.603 ± 0.0069	0.635 ± 0.0050
II	0.757 ± 0.0044	0.589 ± 0.0034	0.548 ± 0.0066	0.547 ± 0.0042
III	0.873 ± 0.0040	0.733 ± 0.0040	0.624 ± 0.0075	0.647 ± 0.0055
IV	0.849 ± 0.0042	0.680 ± 0.0047	0.622 ± 0.0087	0.636 ± 0.0064
V	0.802 ± 0.0037	0.658 ± 0.0051	0.537 ± 0.0072	0.564 ± 0.0058
VI	0.867 ± 0.0021	0.715 ± 0.0025	0.673 ± 0.0062	0.677 ± 0.0050
VII	0.829 ± 0.0033	0.707 ± 0.0026	0.550 ± 0.0062	0.608 ± 0.0045
VIII	0.748 ± 0.0032	0.636 ± 0.0026	0.564 ± 0.0060	0.593 ± 0.0048
IX	0.717 ± 0.0038	0.690 ± 0.0033	0.528 ± 0.0053	0.622 ± 0.0051
X	0.370 ± 0.0033	0.190 ± 0.0026	0.200 ± 0.0037	0.189 ± 0.0035
XI	0.969 ± 0.0016	0.936 ± 0.0021	0.848 ± 0.0060	0.886 ± 0.0038
Avg.	0.785 ± 0.0011	0.657 ± 0.0011	0.572 ± 0.0020	0.600 ± 0.0015

when the time step increases. Second, the prediction can be more effective when more global minima have been identified in previous time steps since the location of more global minima in the new time step can be predicted. This trend is observable for other prediction methods to some extent, but it is not as significant as in the case of AMLP. As a result, the superiority of AMLP intensifies for later time steps.

- In case I (Figure 11), AMLP outperforms all the prediction methods in all six test problems by a clear margin. The second-best prediction method is problem-dependent. For example, for F_5, it is AR, while the second most successful prediction method is PRE for F_1 and F_4
- Table III shows that AMLP is the dominant approach in all cases. In Case IX, PRE is a close competitor. When all cases are considered, MDPR of AMLP is 0.128 higher than PRE, which is its closest competitor.
- Comparing the results of an arbitrary prediction method in Cases I, X, and XI (Table III) reveals that the performance of all the prediction methods substantially improve when the change frequency is increased. All variants of RS-DCMSA have $DMPR > 0.84$ in case XI (compared with Case X, in which the problem changes faster). Nevertheless, a better method is the one that can detect more optimal solutions faster. This is particularly important for fast changing or steadily changing environments [13]. Besides, in practice, the solution from dynamic optimization should be implemented for a time-window. The faster the optimal solution(s) are found, the longer they can be implemented in the actual problem. The practical importance of this issue has been discussed in detail by Deb et al. [13].

VIII. Summary and Conclusion

This study proposed an adaptive multilevel prediction (AMLP) method which utilizes not only the pattern in the movement of the global optima but also the pattern in the prediction error. A higher level makes more use of past information, and thus, can capture more complex change patterns, whereas a lower prediction level is more robust to input error. The proposed adaptation strategy helps the multilevel prediction method to learn the near-optimal prediction level on-the-fly by checking the success of each level of prediction in the previous time step. The drawback of conventional approaches that set the introduced diversity strength proportionally to the norm of the translation of the global optimum was demonstrated. Alternatively, AMLP determines the introduced diversity strength proportionally to the norm of the estimated prediction error.

This study also extended an existing dynamic benchmark generator to enable it to simulate complex and partially random patterns in the movement of their global minima. The dynamic covariance matrix self-adaptation evolution strategy with repelling subpopulation (RS-DCMSA-ES) was also developed by modification of RS-CMSA-ES. This modification enables it to use the outcome of an arbitrary prediction method to set the center and mutation strength of the initialized subpopulation. AMLP was evaluated and compared with some of the most successful prediction methods when they were all incorporated into RS-DCMSA-ES. Eleven cases of six dynamic test problems were tested to form a comprehensive test suite with different values for the problem's dimensionality, number of global minima, change severity, change frequency, and randomness in the change pattern. Our comparison of results showed the superiority of AMLP, especially when there is a complex but smooth change in the problem.

The current study highlighted the importance of the prediction error in previous time steps as valuable information which has been generally overlooked in existing prediction methods. In particular, the prediction error was used in AMLP in three places: First, it was used to formulate the multilevel prediction method by learning the pattern in the past prediction errors. Second, AMLP sets the introduced diversity strength proportionally to the norm of the prediction error. Third, the prediction error in the latest time step was used to identify the near-optimal prediction level at each time step.

Multimodal optimization is an indispensable component of dynamic optimization, especially if a prediction method is to be used, because the size and depths of the minima may change over time, resulting in a significant change in the global minimum, even if the change in the landscape is small. This phenomenon may also occur for multi-objective problems, e.g. a local Pareto optimal solution may become a global Pareto-optimal solution after a change. There are shared challenges and benefits between prediction methods for dynamic multimodal and dynamic multi-objective problems. A prediction method which is successful for one is likely to be successful for the other, at least when some minor adjustments are made. This is another reason that encourages more research on dynamic multimodal optimization. Besides, the proposed methodology in this study paves the way towards a more complicated class of (dynamic) multimodal multi-objective problems. For example, it is possible to direct each subpopulation to a distinct subregion of the Pareto set. If the problem is dynamic, AMLP can be used to provide a reasonably good seed population for the new time step by tracking the centers of these subregions over time.

IX. Acknowledgments

This study was funded by Australian Research Council Discovery Project DP190102637. The last author acknowledges support from CONACyT project no. 2016-01-1920 and from a 2018 SEP-Cinvestav grant (application no. 4). Computational work in this study was supported by National Computational Infrastructure of Australia.

References

[1] R. Azzouz, S. Bechikh, L. B. Said, and W. Trabelsi, "Handling time-varying constraints and objectives in dynamic evolutionary multi-objective optimization," *Swarm and evolutionary computation*, vol. 39, pp. 222–248, 2018.

[2] C. Bu, W. Luo, and L. Yue, "Continuous dynamic constrained optimization with ensemble of locating and tracking feasible regions strategies," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 14–33, 2016.

[3] S. Jiang, M. Kaiser, S. Yang, S. Kollias, and N. Krasnogor, "A scalable test suite for continuous dynamic multiobjective optimization," *IEEE transactions on cybernetics*, 2019.

[4] R. Chen, K. Li, and X. Yao, "Dynamic multiobjectives optimization with a changing number of objectives," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 157–171, 2017.

[5] S. Scolan, S. Serra, S. Sochard, P. Delmas, and J.-M. Reneaume, "Dynamic optimization of the operation of a solar thermal plant," *Solar Energy*, vol. 198, pp. 643–657, 2020.

[6] D. Rohde, B. R. Knudsen, T. Andresen, and N. Nord, "Dynamic optimization of control setpoints for an integrated heating and cooling system with thermal energy storages," *Energy*, vol. 193, p. 116771, 2020.

[7] M. Lucio and L. A. Ricardez-Sandoval, "Dynamic modelling and optimal control strategies for chemical-looping combustion in an industrial-scale packed bed reactor," *Fuel*, vol. 262, p. 116544, 2020.

[8] Z. Wang and M. Gong, "Dynamic deployment optimization of near space communication system using a novel estimation of distribution algorithm," *Applied Soft Computing*, vol. 78, pp. 569–582, 2019.

[9] Y. Liu, "An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones," *Computers & Operations Research*, vol. 111, pp. 1–20, 2019.

[10] N. R. Sabar, A. Bhaskar, E. Chung, A. Turky, and A. Song, "A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion," *Swarm and evolutionary computation*, vol. 44, pp. 1018–1027, 2019.

[11] C. Yin, Z. Xiao, X. Cao, X. Xi, P. Yang, and D. Wu, "Offline and online search: Uav multiobjective path planning under dynamic urban environment," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 546–558, 2017.

[12] L. Cao, L. Xu, E. D. Goodman, and H. Li, "Decomposition-based evolutionary dynamic multiobjective optimization using a difference model," *Applied Soft Computing*, vol. 76, pp. 473–490, 2019.

[13] K. Deb, S. Karthik *et al.*, "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.

[14] W. T. Koo, C. K. Goh, and K. C. Tan, "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment," *Memetic Computing*, vol. 2, no. 2, pp. 87–110, 2010.

[15] A. Akrami, M. Doostizadeh, and F. Aminifar, "Power system flexibility: an overview of emergence to evolution," *Journal of Modern Power Systems and Clean Energy*, vol. 7, no. 5, pp. 987–1007, 2019.

[16] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, vol. 3. IEEE, 1999, pp. 1875–1882.

[17] Z. Peng, J. Zheng, J. Zou, and M. Liu, "Novel prediction and memory strategies for dynamic multiobjective optimization," *Soft Computing*, vol. 19, no. 9, pp. 2633–2653, 2015.

[18] Q. Li, J. Zou, S. Yang, J. Zheng, and G. Ruan, "A predictive strategy based on special points for evolutionary dynamic multi-objective optimization," *Soft Computing*, pp. 1–17, 2018.

[19] G. Ruan, G. Yu, J. Zheng, J. Zou, and S. Yang, "The effect of diversity maintenance on prediction in dynamic multi-objective optimization," *Applied Soft Computing*, vol. 58, pp. 631–647, 2017.

[20] R. Liu, J. Li, C. Mu, L. Jiao *et al.*, "A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization," *European Journal of Operational Research*, vol. 261, no. 3, pp. 1028–1051, 2017.

[21] L. Shi, Y. Wu, and Y. Zhou, "A hybrid immigrants strategy for dynamic multi-objective optimization," in *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*. IEEE, 2018, pp. 589–593.

[22] C. Li and S. Yang, "A general framework of multipopulation methods with clustering in undetectable dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 556–577, 2012.

[23] R. Vafashoar and M. R. Meybodi, "A multi-population differential evolution algorithm based on cellular learning automata and evolutionary context information for optimization in dynamic environments," *Applied Soft Computing*, vol. 88, p. 106009, 2020.

[24] R. Liu, J. Li, J. Fan, and L. Jiao, "A dynamic multiple populations particle swarm optimization algorithm based on decomposition and prediction," *Applied Soft Computing*, vol. 73, pp. 434–459, 2018.

[25] X.-W. Luo, Z.-J. Wang, R.-C. Guan, Z.-H. Zhan, and Y. Gao, "A distributed multiple populations framework for evolutionary algorithm in solving dynamic optimization problems," *IEEE Access*, vol. 7, pp. 44 372–44 390, 2019.

[26] A. Meier and O. Kramer, "Prediction in nature-inspired dynamic optimization," in *Frontier Applications of Nature Inspired Computation*. Springer, 2020, pp. 34–52.

[27] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 6, pp. 1–24, 2012.

[28] T. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments," in *Workshops on Applications of Evolutionary Computation*. Springer, 2004, pp. 489–500.

[29] X. Li, M. G. Epitropakis, K. Deb, and A. Engelbrecht, "Seeking multiple solutions: an updated survey on niching methods and their applications," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 518–538, 2016.

[30] S. Das, S. Maity, B.-Y. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimization-a survey of the state-of-the-art," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 71–88, 2011.

[31] W. Luo, X. Lin, T. Zhu, and P. Xu, "A clonal selection algorithm for dynamic multimodal function optimization," *Swarm and Evolutionary Computation*, vol. 50, p. 100459, 2019.

[32] S. Cheng, H. Lu, Y.-n. Guo, X. Lei, J. Liang, J. Chen, and Y. Shi, "Dynamic multimodal optimization: A preliminary study," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 279–285.

[33] S. Kundu, S. Biswas, S. Das, and P. N. Suganthan, "Crowding-based local differential evolution with speciation-based memory archive for dynamic multimodal optimization," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*. ACM, 2013, pp. 33–40.

[34] S. Cheng, H. Lu, W. Song, J. Chen, and Y. Shi, "Dynamic multimodal optimization using brain storm optimization algorithms," in *International Conference on Bio-Inspired Computing: Theories and Applications*. Springer, 2018, pp. 236–245.

[35] N. Nanas and A. De Roeck, "Multimodal dynamic optimization: from evolutionary algorithms to artificial immune systems," in *International Conference on Artificial Immune Systems*. Springer, 2007, pp. 13–24.

[36] Z. Liang, S. Zheng, Z. Zhu, and S. Yang, "Hybrid of memory and prediction strategies for dynamic multiobjective optimization," *Information Sciences*, vol. 485, pp. 200–218, 2019.

[37] A. Ahrari, S. Elsayed, R. Sarker, and D. Essam, "A new prediction approach for dynamic multiobjective optimization," in *Proceedings of the Congress on Evolutionary Computation*. IEEE, 2019, p. in press.

[38] M. Rong, D. Gong, Y. Zhang, Y. Jin, and W. Pedrycz, "Multidirectional prediction approach for dynamic multiobjective optimization problems," *IEEE transactions on cybernetics*, vol. 49, no. 9, pp. 3362–3374, 2018.

[39] X.-F. Liu, Y.-R. Zhou, and X. Yu, "Cooperative particle swarm optimization with reference-point-based prediction strategy for dynamic multi-objective optimization," *Applied Soft Computing*, vol. 87, p. 105988, 2020.

[40] J. Ou, J. Zheng, G. Ruan, Y. Hu, J. Zou, M. Li, S. Yang, and X. Tan, "A pareto-based evolutionary algorithm using decomposition and truncation for dynamic multi-objective optimization," *Applied Soft Computing*, p. 105673, 2019.

[41] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 832–846.

[42] A. Ahrari, K. Deb, and M. Preuss, "Multimodal optimization by covariance matrix self-adaptation evolution strategy with repelling subpopulations," *Evolutionary Computation*, vol. 25, no. 3, pp. 439–471, 2017.

[43] A. Ahrari and K. Deb, "A novel class of test problems for performance evaluation of niching methods," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 6, pp. 909–919, 2017.

[44] I. Hatzakis and D. Wallace, "Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 2006, pp. 1201–1208.

[45] J. Zou, Q. Li, S. Yang, H. Bai, and J. Zheng, "A prediction strategy based on center points and knee points for evolutionary dynamic multi-objective optimization," *Applied Soft Computing*, vol. 61, pp. 806–818, 2017.

[46] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization," *IEEE transactions on cybernetics*, vol. 44, no. 1, pp. 40–53, 2013.

[47] S. Jiang and S. Yang, "A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 65–82, 2017.

[48] D. Chen, F. Zou, R. Lu, and X. Wang, "A hybrid fuzzy inference prediction strategy for dynamic multi-objective optimization," *Swarm and evolutionary computation*, vol. 43, pp. 147–165, 2018.

[49] T. Schneider and A. Neumaier, "Algorithm 808: Arfit—a matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models," *ACM Transactions on Mathematical Software (TOMS)*, vol. 27, no. 1, pp. 58–65, 2001.

[50] R. Rambabu, P. Vadakkepat, K. C. Tan, and M. Jiang, "A mixture-of-experts prediction framework for evolutionary dynamic multiobjective optimization," *IEEE transactions on cybernetics*, 2019.

[51] S. B. Gee, K. C. Tan, and H. A. Abbass, "A benchmark test suite for dynamic evolutionary multiobjective optimization," *IEEE transactions on cybernetics*, vol. 47, no. 2, pp. 461–472, 2017.

[52] S. Jiang and S. Yang, "Evolutionary dynamic multiobjective optimization: Benchmarks and algorithm comparisons," *IEEE transactions on cybernetics*, vol. 47, no. 1, pp. 198–211, 2017.

[53] Y. Wang, J. Yu, S. Yang, S. Jiang, and S. Zhao, "Evolutionary dynamic constrained optimization: Test suite construction and algorithm comparisons," *Swarm and Evolutionary Computation*, vol. 50, p. 100559, 2019.

[54] S. Zeng, R. Jiao, C. Li, X. Li, and J. S. Alkasassbeh, "A general framework of dynamic constrained multiobjective evolutionary algorithms for constrained optimization," *IEEE transactions on cybernetics*, vol. 47, no. 9, pp. 2678–2688, 2017.

[55] I. Moser and R. Chiong, "Dynamic function optimization: the moving peaks benchmark," in *Metaheuristics for Dynamic Optimization*. Springer, 2013, pp. 35–59.

[56] R. Mukherjee, G. R. Patra, R. Kundu, and S. Das, "Cluster-based differential evolution with crowding archive for niching in dynamic environments," *Information Sciences*, vol. 267, pp. 58–82, 2014.

[57] F. B. Ozsoydan and A. Baykasoğlu, "Quantum firefly swarms for multimodal dynamic optimization problems," *Expert Systems with Applications*, vol. 115, pp. 189–199, 2019.

[58] C. Li, S. Yang, and M. Yang, "An adaptive multi-swarm optimizer for dynamic optimization problems," *Evolutionary computation*, vol. 22, no. 4, pp. 559–594, 2014.

[59] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," 2009.

[60] C. Li and S. Yang, "A generalized approach to construct benchmark problems for dynamic optimization," in *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer, 2008, pp. 391–400.

[61] C. Li, S. Yang, T. Nguyen, E. L. Yu, X. Yao, Y. Jin, H. Beyer, and P. Suganthan, "Benchmark generator for cec 2009 competition on dynamic optimization," Tech. Rep., 2008.

Ali Ahrari received his Ph.D. degree in mechanical engineering from Michigan State University, Michigan, USA. He is now a postdoc research associate at the University of New South Wales, ACT, Australia. His research concentrates on evolutionary algorithms and engineering optimization. He has won GECCO competition in multimodal optimization in 2016 and 2017, and ISCSO student competition in structural optimization in 2017 and 2018. He is now a member of the IEEE CIS Task Force on Multimodal Optimization.

Saber Elsayed received the Ph.D.degree in Computer Science from the University of New South Wales Canberra, Australia, in 2012. Currently, Saber is a Senior Lecturer with the School of Engineering and Information Technology, University of New South Wales Canberra. His research interests include the areas of evolutionary algorithms, constraint-handling techniques for evolutionary algorithms, scheduling, big data and cybersecurity using computational intelligence. Saber won several IEEE-CEC competitions. Dr. Elsayed is serving as the chair of the IEEE Computational Intelligence Society (ACT Chapter), and has held organizational roles in several conferences.

Ruhul Sarker received his Ph.D. in 1992 from Dalhousie University, Halifax, Canada. He is currently a Professor in the School of Engineering and IT, Canberra, Australia. His main research interests are Evolutionary Optimization, and Applied Operations Research. He is the lead author of the book *Optimization Modelling: A Practical Approach*. Prof. Sarker is a member of IEEE and INFORMS.

Daryl Essam received his B.Sc. degree in computer science from University of New England, Australia in 1990 and his Ph.D. degree from University of New South Wales, Australia, in 2000. Since 1994, he has been with the Canberra campus, UNSW, where he is currently a Senior Lecturer. His research interests include genetic algorithms, with a focus on both evolutionary optimisation and large scale problems.

Carlos A. Coello Coello (M'98-SM'04-F'11) received a PhD in computer science from Tulane University, USA, in 1996. He is currently Professor with Distinction (CINVESTAV-3F Researcher) at the Computer Science Department of CINVESTAV-IPN, in Mexico City, Mexico. Dr. Coello has authored and co-authored over 450 technical papers and book chapters. He has also co-authored the book *Evolutionary Algorithms for Solving Multi-Objective Problems* (Second Edition, Springer, 2007). His publications currently report over 44,800 citations in Google Scholar (his h-index is 83). Currently, he is associate editor of the IEEE Transactions on Evolutionary Computation and serves in the editorial board of several other international journals. He received the 2007 National Research Award from the Mexican Academy of Sciences in the area of Exact Sciences, the Medal to the Scientific Merit 2009, granted by Mexico City's congress and the National Medal of Science and Arts in the area of Physical, Mathematical and Natural Sciences. He is a Fellow of the IEEE, and a member of the ACM, Sigma Xi, and the Mexican Academy of Science. His major research interests are: evolutionary multi-objective optimization and constraint-handling techniques for evolutionary algorithms.