A Vector Autoregression-Based Algorithm for Dynamic Many-Objective Optimization Problems

Kalthoum Karkazan[1], Haluk Rahmi Topcuoglu[1] and Shaaban Sahmoud[2]
[1]*Faculty of Engineering, Computer Engineering Department, Marmara University, Istanbul, Turkey*
[2]*Faculty of Engineering, Computer Engineering Department, Fatih Sultan Mehmet Vakif University, Istanbul, Turkey*

Keywords: Dynamic Many-Objective Optimization, Many-Objective Evolutionary Algorithms, Change Detection, Prediction-Based Optimization.

Abstract: Dynamic Many-Objective Optimization Problems (DMaOPs) represent a significant challenge due to their inherent dynamism and the presence of a large number of objectives. In addressing this complexity, this paper proposes a new prediction-based strategy tailored to managing detected changes in such problems, which is one of the first attempts to address the DMaOPs. Our proposed algorithm constructs a Vector Autoregressive (VAR) model within a dimensionality-reduced space. This model effectively captures the mutual relationships among decision variables and enables an accurate prediction of the initial positions for the evolving solutions in dynamic environments. To accelerate the convergence process, the algorithm demonstrates adaptability by responding multiple times to the same detected change. In our empirical study, the performance of the proposed algorithm is evaluated using four selected test problems from various benchmarks. Our proposed approach shows competitive results compared to the other algorithms in most test instances.

1 INTRODUCTION

Many-objective optimization problems (MaOPs) are defined as optimization problems that have four or more conflicting objectives to be optimized simultaneously (Farina and Amato, 2002). Many real-world problems from diverse domains belong to this type of optimization problem where they sometimes involve up to 15 objectives or more (Coello and Lamont, 2004). One illustrative example is explored in (Zhu et al., 2019), where the authors argue that the evolving societal landscape necessitates increasing demand for high-quality power. Recognizing the limitations of single-objective economic optimization, there is a persistent need for a shift toward many-objective power flow optimization. These objectives primarily encompass economic, environmental, reliability, and stability factors, aiming for comprehensive modern power system management. Another instance is presented in (Li et al., 2008), where the authors investigate the power generation loading optimization problem with four objectives: minimize total cost, minimize fuel consumption, minimize emissions, and maximize output.

The primary distinction between a dynamic optimization problem (DOP) and a stationary optimization problem lies in the presence of dynamism within the environment (Jin and Branke, 2005). Dynamism can manifest in various forms, encompassing changes in one or more objective functions, modifications in constraints, and/or shifts in problem parameters across time (Jin and Branke, 2005). Solutions deemed optimal or the best before a change may not retain their optimality or superiority in the new environment. Consequently, the main objective of a dynamic optimization problem is to track the global optima closely. This dynamic nature presents additional complexities when dealing with such problems.

While most of the current research on MaOPs assumes that the environment is stationary, in real-world scenarios, many factors can affect the environment or the optimization problem objectives and variables (Gupta and Nanda, 2021). To the best of our knowledge, this paper is one of the first to try to handle this type of optimization problem called Dynamic Many-Objective Optimization Problems (DMaOPs).

The complexity of dynamic many-objective optimization problems requires advanced optimization algorithms and strategies to explore the high-dimensional solution space efficiently and quickly adapt to new environmental changes. Therefore, the algorithms designed for static many-objective prob-

Karkazan, K., Topcuoglu, H. and Sahmoud, S.
A Vector Autoregression-Based Algorithm for Dynamic Many-Objective Optimization Problems.
DOI: 10.5220/0013009200003837

In *Proceedings of the 16th International Joint Conference on Computational Intelligence (IJCCI 2024)*, pages 279-287
ISBN: 978-989-758-721-4; ISSN: 2184-3236

lems are not adequate for handling the changing nature of DMaOPs. As a result, these problems need the development of adaptive algorithms that can consistently update the estimated Pareto-optimal front (POF) to align it as quickly, accurately, and uniformly diversified as possible to the actual POF.

In this paper, we present the design and development of a dynamic many-objective evolutionary algorithm (DMaOEA) that effectively solves dynamic many-objective optimization problems (DMaOPs). Our algorithm adapts to changes in the problem environment and efficiently handles the trade-offs between multiple conflicting objectives over time to ensure the attainment of an accurately estimated Pareto-optimal front (POF). The proposed DMaOEA consistently tracks the shifting Pareto-optimal fronts (POFs) and produces a widely spread set of solutions across it. The algorithm's effectiveness is assessed using different metrics that measure the estimated POFs' convergence and diversity.

The rest of this paper is organized as follows: Section 2 defines the dynamic many-objective optimization problem and summarizes the related work. Section 3 describes and explains the proposed algorithm in detail. Experimental settings and results are given in Section 4. Section 5 concludes the paper and proposes some future research directions.

2 DYNAMIC MANY-OBJECTIVE OPTIMIZATION

In this section, we focus on many-objective optimization and dynamic multi-objective optimization because they are considered the two origins from which Dynamic Many-Objective Optimization Problems (DMaOPs) are derived. A DMaOP is an optimization problem characterized by two merged key attributes: the existence of dynamism within the problem environment and the involvement of many conflicting objectives.

As in dynamic multi-objective optimization problems, the dynamism of DMaOP is exemplified by variations over time in one or more aspects of the optimization problem, such as alterations in objective functions, constraints, or problem parameters. Concurrently, the optimization problem is more complex by the inclusion of a significant number of conflicting objectives, typically exceeding four. As a result of this large number of objectives, the Pareto-dominance becomes ineffective in differentiating between individuals.

In MaOPs, obtaining a well-distributed set of non-dominated solutions on the POF becomes increasingly challenging as the number of objectives rises. Moreover, the visualization for this type of problem is complicated as the traditional graphical methods struggle to represent high-dimensional Pareto fronts (Deb and Jain, 2014). In addition, evaluating many-objective optimization problems is computationally expensive since the performance metrics require larger computational effort as the number of objectives increases (C.M. Fonseca and Lopez-Ibanez, 2006).

Although the performance of the existing MOEAs has dropped dramatically when handling MaOPs due to the difficulties mentioned above, they can be used as a baseline to develop effective MaOEAs. Many successful MOEAs have been adapted in literature and have proven their effectiveness in solving various real-world MaOPs such as the NSGA-III (Deb and Jain, 2014) and the MOEA/D (Zhang and Li, 2007). The main idea in these two algorithms is to divide the search space into predefined multiple targeted searches. As a result, an individual will compete with only the individuals that are associated with its search target rather than compete with all the other individuals in the population. Consequently, optimal points corresponding to each targeted search task are identified, easing the difficulty associated with handling a large non-dominated set as well as maintaining a good level of diversity in the population. Unfortunately, most MaOEAs are not directly applicable to DMaOPs because of the existence of various forms of dynamism which dramatically degrade the performance.

In the literature, there are many approaches have been proposed to solve DMOPs including diversity-based (Deb et al., 2007), problem type-based (Sahmoud and Topcuoglu, 2018), memory-based (Sahmoud and Topcuoglu, 2020), and prediction-based (Azzouz et al., 2017; Karkazan et al., 2023).

On the other hand, there are many attempts to benefit from Machine Learning models to improve the prediction results such as using Support Vector Machine (SVM) (Weizhen et al., 2019) and transfer learning (Jiang et al., 2017). In a recent research work, the authors of (Jiang et al., 2023) introduced an algorithm called Vector Autoregressive Evolution (VARE), which combines Vector Autoregression (VAR) and environment-aware hypermutation (EAH) techniques to track the changing POFs in DMOPs. The VARE constructs a VAR model that captures the mutual relationships between decision variables, enabling accurate prediction of the locations of new solutions in the evolving environment. Additionally, the VARE incorporates environment-aware hypermutation techniques to enhance popu-

lation diversity and improve the algorithm's performance in cases where the prediction approach is unsuitable.

3 A NEW PREDICTION-BASED ALGORITHM

Our proposed algorithm utilizes the NSGA-III algorithm (Deb and Jain, 2014) as a baseline, enhancing its ability to deal with environmental changes. It also applies the Principle Component Analysis (PCA) for constructing a Vector Auto Regression (VAR) model. In the following subsections, the VAR and the PCA are explained briefly, which is followed by the steps of our algorithm.

3.1 Vector Auto Regression

Multivariate time series is a special case of time series where multiple variables are recorded over time, creating a sequence of data points. These variables can be interrelated and impact each other's values. In other words, each variable is influenced not only by its past values but also by other variables. This interdependency is utilized to predict future values. Analyzing those time series and creating models from them opens doors to comprehending and predicting trends and patterns effectively (Jonathan D. Cryer, 2008).

Vector Auto Regression (VAR) is a widely utilized approach for forecasting multivariate time series data. This technique has consistently demonstrated its effectiveness across various fields by capturing intricate relationships among multiple time series variables and accurately predicting future values. Some of its applications include but are not limited to economic forecasting, financial market analysis, climate and environmental studies, and social sciences. In a VAR model, each variable is modeled as a linear combination of its past values as well as the past values of other variables in the system (Jin and Chen, 2013).

According to (S. Huband and While, 2006), the decision variables in a MOP/MaOP are often correlated and interrelated. This characteristic makes VAR models a highly suitable technique to address these problems. However, there is a critical consideration before applying VAR to MaOPs. Although VAR models deliver accurate prediction results, their performance can be significantly impacted when dealing with high-dimensional problems, as in the case of MaOP decision variables. To mitigate this effect, a dimension reduction technique is employed as explained in the following section.

3.2 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical method used for dimensionality reduction, outliers detection, data analysis, and graphical clustering or classification (Geladi and Linderholm, 2020). It identifies the most significant patterns in a dataset by transforming the original variables into a new set of uncorrelated variables called principal components. These components capture the maximum variance in the data, allowing for a reduction in dimensionality while retaining essential information. Principal Component Analysis (PCA) is a commonly employed technique in multivariate data analysis (Geladi and Linderholm, 2020).

Figure 1 provides a simple example illustrating the process of dimensionality reduction using PCA. The first sub-figure on the left shows the position of the data points in the original space. PCA identifies the best-fitting lower-dimensional surface in which the projection error is minimized, as demonstrated in the middle sub-figure. The final sub-figure represents the points in the new surface after the reduction process.

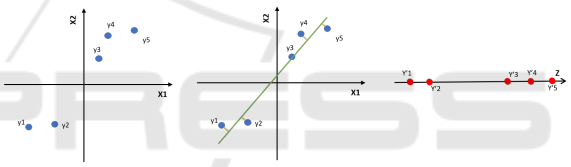

Figure 1: Converting 2-D space to a 1-D space using PCA.

Let $POP_{ml} = x_1, x_2, ..., x_n$ (where each x_i is a $D \times 1$ vector) represent all archived solutions up to the current environment. We aim to map any data point $x_i \in pop_{ml}$ from the $D - Dimentions$ space to a much lower $k - Dimentions$ $(k \ll n)$ space, namely,

$$x_i = (x_i^1, x_i^2, ..., x_i^D) \Rightarrow y_i = (y_i^1, y_i^2, ..., y_i^k) \quad (1)$$

One advantage of the PCA algorithm is the ability to reconstruct the points in the original space (i.e. from y_i back to x_i), as demonstrated in Figure 2. This feature is essential in the context of our algorithm and makes PCA a good choice for dimensionality reduction.

3.3 Non-Dominated Sorting Genetic Algorithm III (NSGA-III)

The NSGA-III (Deb and Jain, 2014) algorithm is an extension of the popular NSGA-II algorithm to adapt for many-objective optimization problems. The fundamental structure remains same to the original NSGA-II algorithm, with substantial modifications in

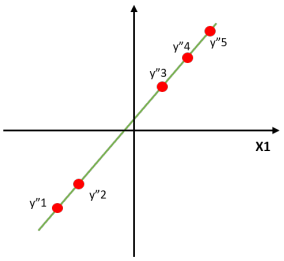

Figure 2: Reconstruction of 2-D space using PCA.

its selection mechanism which is designed to overcome the main difficulty mentioned in Section 2. This new approach also helps in maintaining a diverse set of solutions. The main idea behind the modifications in NSGA-III is to initialize the algorithm with reference points (RPs) at the start of the run. The positions of individuals concerning these RPs serve as the foundation for the selection process, diverging from the approach of using crowding distance as in NSGA-II.

3.4 Change Response Strategy

To effectively characterize changes in dynamic multi-objective optimization problems, it is vital to both detect the changes and evaluate their severity. The severity of a change typically refers to the magnitude or extent of modifications made to the problem's parameters, constraints, or objective function. When a dynamic multi-objective evolutionary algorithm incorporates a precise mechanism for detecting the severity of changes, it becomes beneficial to enhance the algorithm with dynamic and efficient mechanisms to appropriately respond to the detected severity of change (Sahmoud and Topcuoglu, 2019). In this paper, we will adapt the same strategy for dynamic many-objective optimization problems.

When a change is detected, the severity of change is calculated according to Equation 2. It is important to highlight that employing this equation for calculating the severity of change resolves issues encountered when determining the overall severity of change (SC) by choosing the maximum value among the calculated severities for each objective function independently (Sahmoud and Topcuoglu, 2019).

$$SC_j = \sum_{i=1}^{S} \frac{F_{i,j}(t) - F_{i,j}(t-1)}{F_{i,j}(t) + s} \qquad (2)$$
$$SC = \lambda \times \max(SC_1, SC_2, ..., SC_M)$$

In this equation, SC_j denotes the severity of change for the objective function j, and S represents the number of selected indicators for detecting environmental changes. $F_{i,j}(t)$ and $F_{i,j}(t-1)$ signify the values of objective function j for sensor i in the current and previous generations, respectively. The term s is a small constant utilized to prevent undefined results when $F_{i,j}(t)$ is zero. Additionally, λ is employed to adapt the equation for varying numbers of objective functions. In this study, it is set to $M-1$ as recommended by the reference study (Sahmoud and Topcuoglu, 2019), where M represents the objectives of the given problem.

If the severity of change is greater than or equal to the average value, ε% of the population is replaced by randomly generated individuals. Otherwise, the algorithm will replace ε% of the population with mutated solutions. It is noteworthy that this technique is employed because a significant change suggests that the new POF's location is likely far from the old POF's position. Hence, to explore new locations in the search space, the re-initialization procedure is implemented. Conversely, a small severity of change value indicates a slight shift in the new POF's location, and the mutation operation is sufficient to track it. In our empirical study, ε% is equal to 20% of the population.

In the predetermined number of generations following the detected change, the algorithm systematically archives the decision variable values of the closest individual associated with each reference point, if there are any, for subsequent usage. These values serve as training samples when constructing the vector autoregression model. In this study, we collect samples for five generations after each change. After that, the collected training data are used to construct and train a vector autoregression model. The process for doing that is explained in detail in the following subsection. Algorithm 1 explains how the proposed algorithm reacts to a detected change.

3.5 Prediction Strategies for Individuals

After collecting a sufficient number of training samples, our algorithm utilizes these samples to construct a vector autoregression model. In the preparatory phase, the algorithm employs PCA for dimensionality reduction, constructing a VAR model within the reduced space of time series solutions. Subsequently, this VAR model is utilized to predict new individuals, and these newly predicted individuals are reconstructed with PCA to revert to the original number of dimensions before being added to the population. To maintain a high level of diversity in the population, a fixed percentage of the population is substituted with randomly generated individuals. In this empirical study, a 5% of the population is replaced each time the algorithm responds.

The visualization in Figure 3 illustrates the con-

if *ChangeDetected* **then**
  Calculate the *SeverityOfChange*
  if *SeverityOfChange* $>=$ *AvgSeverityOfChange* **then**
    Reinitialize ε% of the population
  else
    Mutate ε% of the population
  end if
  Update *AvgSeverityOfChange*
  count $\leftarrow 0$
end if
if *count* $<=$ *TS* **then**
  Store the closest individual of each reference point in the archive
else
  if *TS* mod $\lambda \equiv 0$ **then**
    Reinitialize α% of the population
    Construct VAR and employ it to forecast new population members
  end if
end if
count $\leftarrow$ *count* $+ 1$

Algorithm 1: The Change Reaction Strategy.

cept. The black points represent the reference points generated by the NSGA-III algorithm at the beginning of the run. Within the objective function space, the green points denote the selected individuals to be stored in the archive and utilized as training samples. When a point is chosen for archiving, values of its decision variables are stored and all of the archived information is employed to construct a model that predicts new individuals, which is indicated by the green points in the decision variable space in Figure3. The resulting individuals predicted by the VAR model are represented as blue points in the decision variable space.

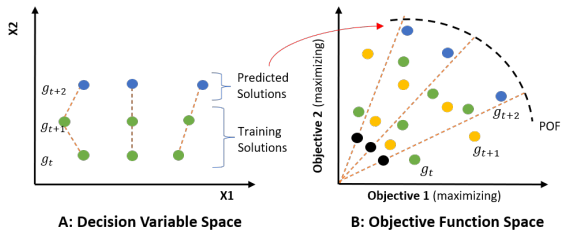

Figure 3: The proposed prediction strategy for individuals in the new detecting environments.

3.6 Considering Multiple Reactions

Finally, periodic reactions are applied after the new environment has been running for a specific number of generations. In our empirical study, the algorithm reacts periodically after every five generations. This step helps accelerate the convergence process and prevents the solutions from sticking to the local optimum. This step is very important for the performance of the algorithm. After each generation more information is collected about the new environment and the prediction process will be more accurate, which can help catch up any error in the previous prediction process.

Figure 4 summarizes the proposed algorithm in a flowchart. As depicted in the flowchart, the algorithm initiates by randomly initializing individuals for the first generation, configuring the algorithm's parameters, and generating reference points. Subsequently, the environment is sensed at the inception of each generation. If no change is detected, the static MOEA, specifically NSGA-III, is activated, and the population evolves through its genetic operators. Conversely, upon detecting a change, the algorithm computes the severity of the change and updates a portion of the population accordingly. Simultaneously, it begins collecting training samples by storing the closest individual for each reference point (RP) in the archive. When the number of generations used for collecting training samples reaches a specific threshold called *TS*, the algorithm employs the dimensional reduction technique, namely Principle Component Analysis (PCA) on the training samples. The output of PCA is then used to construct a VAR model, which is used to predict the new individuals that are injected into the population.

4 EXPERIMENTAL DESIGN

This section outlines the experimental design, encompassing information about the test problems, performance evaluation metrics, and the algorithms in our comparative study.

4.1 Test Problems

The performance of algorithms is evaluated using a total of four test instances. Two of these problems, FDA4 and FDA5, are from the FDA test problems (Farina et al., 2004), and the other two, SJY4 and SJY5, are from the SJY test problems (Jiang and Yang, 2014). These specific test instances were selected because they are among the limited number of scalable test instances available in the literature. Due to their scalability, these instances can be tested with more than 3-objective functions, making them suitable for evaluating Dynamic Many-Objective Problems (DMaOPs).

These test suites encompass all types of DMOPs, based on the classification presented by Farina et al. (Farina et al., 2004). Notably, these instances, fea-

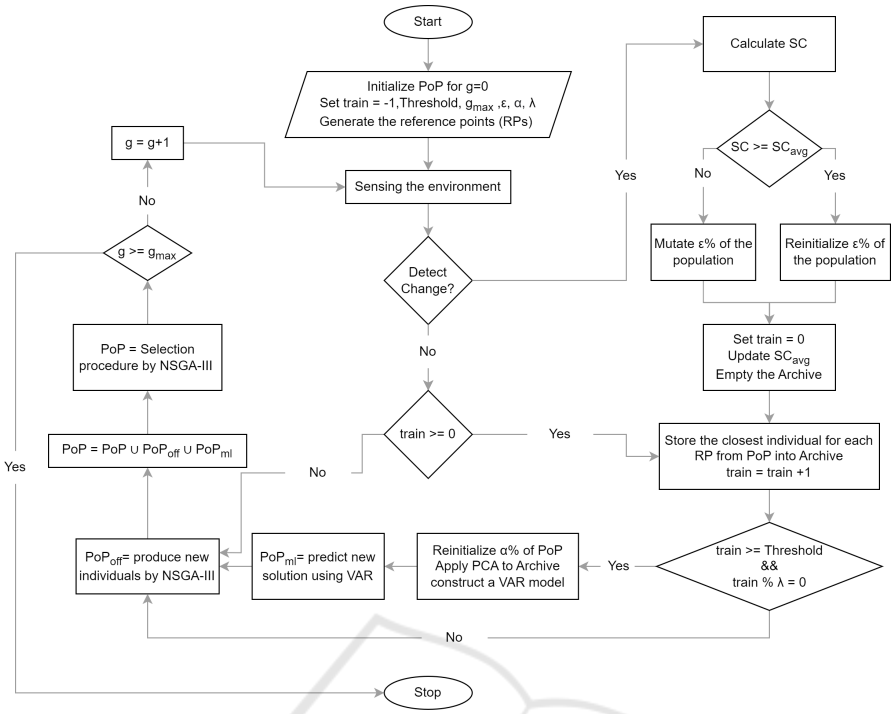

Figure 4: The flowchart of the proposed algorithm for DMaOPs.

Table 1: The used benchmarks types.

Problem	Type
FDA4	Type 1
FDA5	Type 2
SJY4	Type 3
SJY5	Type 4

turing known Pareto-optimal fronts, are specifically designed to assess the performance of algorithms designed to react to the dynamic changes in many-objective optimization problems.

The details of the selected test problems are summarized in Table 1, where the time used in these tests is defined as

$$t = \frac{1}{n_t} * \left\lfloor \frac{\tau}{fr} \right\rfloor \tag{3}$$

where fr is the change frequency, τ is the generations counter, and n_t is the severity of change.

4.2 Performance Metrics

Within this section, the definitions of the utilized metrics are introduced, clarifying their importance in evaluating the outcomes of the developed algorithms. Two of the extensively acknowledged performance metrics are used and their overviews are presented below.

- Mean Inverted Generational Distance (mIGD): This metric is a modified form of the inverted generational distance (IGD) (Li and Zhang, 2009). In this approach, the mean value of the IGD metric across all generations is computed, as illustrated below.

$$mIGD = \frac{1}{|T|} \sum_{t \in T} IGD(PF_t, PF_t^*) \tag{4}$$

The IGD metric (Coello and Cortes, 2005) is employed to assess the diversity and convergence of algorithms designed for static multi-objective optimization problems.

$$IGD(PF_t, PF_t^*) = \frac{\sum_{v \in PF_t} d(v, PF_t^*)}{|PF_t|} \tag{5}$$

In this context, PF_t denotes the set representing uniformly distributed points in the true Pareto front and PF_t^* is the set comprising the estimated Pareto optimal front generated by the algorithm at the t-th time step. The minimum Euclidean distance between a point v in PF_t and the points in PF_t^* is expressed as $d(v, PF_t^*)$.

- Average Maximum Spread (aMS): This metric quantifies the diversity of the estimated Pareto optimal front by assessing its dispersion across the true Pareto optimal front. Originally designed for application in steady-state multi-objective environments, it is adjusted for dynamic environments

by computing the average of its values across all generations, as demonstrated below (Goh and Tan, 2009):

$$aMS = \sqrt{\frac{1}{M}\sum_{k=1}^{M}\left[\frac{min\left[\overline{POF_k},\overline{POF_k^*}\right]-max\left[\underline{POF_k},\underline{POF_k^*}\right]}{\overline{POF_k}-\underline{POF_k}}\right]} \quad (6)$$

In this equation, $\underline{POF_k^*}$ denotes the minimum value of the k-th objective in the estimated Pareto optimal front. M denotes the count of fitness functions, and $\overline{POF_k}$ refers to the maximum value of the k-th objective in the true Pareto optimal front.

4.3 Algorithms in the Empirical Study

Two DMaOEAs are used in the comparison study: the dynamic version of the NSGA-III algorithm (NSGA-III-Reinitialize), and the Dynamic non-dominated sorting genetic algorithm III (DNSGA-III) (Gupta and Nanda, 2021).

- NSGA-III-Reinitialize: This algorithm is adapted for dynamic problems by introducing new individuals when a change in the environment is detected. The newly introduced solutions are chosen to be 20% of the population size and they are generated by random initialization.
- Dynamic Non-dominated Sorting Genetic Algorithm III (DNSGA-III): The only reference algorithm found in the literature proposed to handle DMaOPs is the DNSGA-III, which is adapted from the NSGA-III algorithm. To address the dynamism in the problem, the authors suggested using machine learning (ML) techniques to predict the initial location of individuals in future environments (Gupta and Nanda, 2021). The authors studied the effect of applying different ML techniques for the prediction process. Those techniques include support vector regression (SVR) with linear kernel and radial basis function (RBF) kernel, polynomial interpolation, and cubic spline-based prediction. According to the authors, the algorithm delivers its best results when the prediction is conducted using the support vector regression with radial basis function (SVR-RBF) kernel ML technique. Consequently, this technique was utilized during the simulation of this algorithm in our empirical study.

4.4 Parameters Setting

Following are the values for some of the key parameters as been set in the experiment:

- `Number of Objectives (M)`: Three different values have been chosen to effectively test the algorithms' ability to deal with MaOPs.
- `Number of Reference Points (H)`: This value has been calculated according to the number of objectives as suggested by the authors of NSGA-III (Deb and Jain, 2014).
- `Population size (N)`: This value is set for each test case as the smallest multiple of four greater than H (Deb and Jain, 2014).
- `Number of Decision Variables (D)`: The No. of decision variable is suggested in (Deb and Jain, 2014) to be calculated by the following equation:

$$D = M + k - 1 \quad (7)$$

where M is the number of objectives, and $k = 10$ for the problems extended from DTLZ2, namely, FDA4 and FDA5.

- `Change Frequency`: For each test instance, each algorithm has been tested with three different values for change frequency τ_t, those values are 10, 20, and 30.
- `Number of Generations`: At each run, the number of generations has been specified according to the change frequency for that run. 10 generations have been given between each two consecutive changes to give the algorithms the chance to converge to the real POF. This is represented by, $10 * \tau_t$
- `Number of Executions`: Each algorithm was executed 30 independent times on each test instance.

4.5 Experimental Results

In this section, the effectiveness of our algorithm is evaluated and compared with other DMaOEAs within the context of our experimental framework. All of the experiments ran on the same experimental platform PlatEMO (Tian et al., 2017). This platform is an evolutionary multi-objective optimization platform developed in MATLAB for free use in research and educational purposes. It supports diverse benchmark problems, performance metrics, and visualization tools.

Table 2 shows the results of the mIGD metric. The best results from each of the three algorithms are denoted in bold. As seen in Table 2, the proposed algorithm outperforms the other algorithms in most of the test instances (23 out of 36 cases). The mIGD metric evaluates both convergence and diversity when measuring an algorithm's performance, so getting competitive results in this metric means that the proposed

Table 2: Mean inverted generational distance values of algorithms.

Problem	M	tau	Proposed Algorithm	NSGA-III -Reinitialize	DNSGA-III
FDA4	4	10	**1.28E-01**	1.31E-01	1.34E-01
		20	1.23E-01	**1.21E-01**	**1.21E-01**
		30	**1.10E-01**	1.11E-01	1.12E-01
	5	10	2.18E-01	2.13E-01	**2.11E-01**
		20	**1.91E-01**	1.93E-01	1.92E-01
		30	1.77E-01	1.77E-01	**1.76E-01**
	6	10	**2.76E-01**	2.95E-01	2.91E-01
		20	**2.20E-01**	2.26E-01	2.25E-01
		30	**2.04E-01**	2.08E-01	2.06E-01
FDA5	4	10	**2.61E-01**	2.70E-01	2.74E-01
		20	**2.39E-01**	2.43E-01	2.44E-01
		30	2.23E-01	**2.22E-01**	2.26E-01
	5	10	4.52E-01	**4.48E-01**	4.50E-01
		20	3.95E-01	**3.92E-01**	3.92E-01
		30	3.58E-01	3.57E-01	**3.55E-01**
	6	10	**6.20E-01**	6.62E-01	6.47E-01
		20	**4.59E-01**	4.79E-01	4.68E-01
		30	**4.18E-01**	4.27E-01	4.22E-01
SJY4	4	10	**1.12E-01**	1.13E-01	**1.12E-01**
		20	**1.08E-01**	**1.08E-01**	1.09E-01
		30	**1.08E-01**	**1.08E-01**	**1.08E-01**
	5	10	1.65E-01	**1.63E-01**	1.63E-01
		20	**1.59E-01**	**1.59E-01**	**1.59E-01**
		30	1.59E-01	1.59E-01	**1.58E-01**
	6	10	**1.78E-01**	**1.78E-01**	**1.78E-01**
		20	1.73E-01	1.73E-01	**1.72E-01**
		30	**1.72E-01**	**1.72E-01**	**1.72E-01**
SJY5	4	10	9.48E-02	**8.95E-02**	9.56E-02
		20	8.21E-02	**8.20E-02**	8.25E-02
		30	**8.17E-02**	**8.17E-02**	8.24E-02
	5	10	**1.39E-01**	1.42E-01	1.43E-01
		20	**1.26E-01**	**1.26E-01**	**1.26E-01**
		30	**1.25E-01**	1.25E-01	**1.25E-01**
	6	10	1.50E-01	**1.48E-01**	1.52E-01
		20	**1.39E-01**	**1.39E-01**	1.40E-01
		30	**1.38E-01**	**1.38E-01**	**1.38E-01**

Table 3: Mean average maximum spread values of algorithms.

Problem	M	tau	Proposed Algorithm	NSGA-III -Reinitialize	DNSGA-III
FDA4	4	10	**1.00E+00**	9.99E-01	9.99E-01
		20	9.92E-01	**9.93E-01**	**9.85E-01**
		30	**9.99E-01**	**9.99E-01**	**9.99E-01**
	5	10	**1.00E+00**	**1.00E+00**	**1.00E+00**
		20	**9.94E-01**	9.91E-01	**9.94E-01**
		30	9.95E-01	9.96E-01	**9.97E-01**
	6	10	**1.00E+00**	**1.00E+00**	**1.00E+00**
		20	**9.98E-01**	**9.98E-01**	9.96E-01
		30	**9.98E-01**	**9.99E-01**	9.98E-01
FDA5	4	10	**9.98E-01**	**9.99E-01**	**9.99E-01**
		20	**9.95E-01**	9.93E-01	9.88E-01
		30	**9.97E-01**	**9.97E-01**	9.96E-01
	5	10	**1.00E+00**	**1.00E+00**	**1.00E+00**
		20	9.91E-01	**9.93E-01**	9.90E-01
		30	9.92E-01	**9.95E-01**	**9.93E-01**
	6	10	**1.00E+00**	**1.00E+00**	**1.00E+00**
		20	**9.98E-01**	**9.98E-01**	**9.98E-01**
		30	**9.96E-01**	**9.97E-01**	9.96E-01
SJY4	4	10	**9.97E-01**	9.94E-01	**9.94E-01**
		20	**9.96E-01**	**9.95E-01**	9.94E-01
		30	**9.96E-01**	**9.96E-01**	**9.95E-01**
	5	10	**1.00E+00**	**9.97E-01**	9.96E-01
		20	**9.94E-01**	**9.94E-01**	**9.93E-01**
		30	**9.94E-01**	**9.94E-01**	**9.93E-01**
	6	10	**1.00E+00**	**1.00E+00**	**1.00E+00**
		20	**9.96E-01**	**9.96E-01**	**9.96E-01**
		30	**9.95E-01**	**9.95E-01**	**9.95E-01**
SJY5	4	10	**9.92E-01**	**9.90E-01**	9.90E-01
		20	**9.96E-01**	**9.94E-01**	9.94E-01
		30	**9.95E-01**	**9.95E-01**	9.94E-01
	5	10	**9.99E-01**	9.90E-01	9.85E-01
		20	**9.93E-01**	**9.93E-01**	**9.93E-01**
		30	**9.94E-01**	9.93E-01	**9.94E-01**
	6	10	**9.99E-01**	**9.99E-01**	**9.99E-01**
		20	**9.93E-01**	**9.94E-01**	**9.95E-01**
		30	**9.95E-01**	**9.95E-01**	**9.94E-01**

algorithm is able to converge successfully to the true POF quickly and maintains the diversity, as well.

Table 3 shows the results of the aMS metric which evaluates the diversity level of the obtained solutions. Although the proposed algorithm obtains the best results, the other algorithms provide competitive results. When those results are compared, it can be observed that there is no significant difference for most of the test instances.

5 CONCLUSION AND FUTURE WORK

The primary objective of this paper is to create an algorithm capable of effectively tackling dynamic many-objective optimization problems. This algorithm adaptively responds to the change according to the severity of the detected change by either mutating or re-initializing a portion of the population. It then preserves the Pareto-optimal front (POF) for certain subsequent populations, utilizing them as training samples to construct a VAR model. This VAR model is then used to predict new individuals that are supposed to be closer to the real POF. The algorithm responds multiple times for the same change to speed up the convergence process. Our algorithm demonstrates better performance results in comparison with the other algorithms utilized in this study.

It is evident that Dynamic Many-Objective Problems (DMaOPs) are relatively new and remain underexplored. Therefore, exploring this area represents a potential research direction. This exploration process may involve proposing new scalable and dynamic test instances, devising more effective performance measures, developing efficient algorithms, and studying the effectiveness of proposed methodologies on real-world problems.

REFERENCES

Azzouz, R., Bechikh, S., and Ben Said, L. (2017). Dynamic multi-objective optimization using evolutionary algorithms: a survey. *Recent advances in evolutionary multi-objective optimization*, pages 31–70.

C.M. Fonseca, L. P. and Lopez-Ibanez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1157–1163.

Coello, C. A. and Cortes, N. C. (2005). Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*, 6(2):163–190.

Coello, C. A. C. and Lamont, G. B. (2004). *Applications*

of multi-objective evolutionary algorithms, volume 1. World Scientific.

Deb, K., Bhaskara Rao N., U., and Karthik, S. (2007). 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*, pages 803–817. Springer.

Deb, K. and Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601.

Farina, M. and Amato, P. (2002). On the optimal solution definition for many-criteria optimization problems. In *2002 Annual Meeting of the North American Fuzzy Information Processing Society Proceedings. NAFIPS-FLINT 2002 (Cat. No. 02TH8622)*, pages 233–238.

Farina, M., Deb, K., and Amato, P. (2004). Dynamic multi-objective optimization problems: Test cases, approximations, and applications. *Evolutionary Computation, IEEE Transactions on*, 8:425 – 442.

Geladi, P. and Linderholm, J. (2020). 2.03 - principal component analysis. In Brown, S., Tauler, R., and Walczak, B., editors, *Comprehensive Chemometrics (Second Edition)*, pages 17–37. Elsevier, Oxford, second edition edition.

Goh, C.-K. and Tan, K. C. (2009). A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(1):103–127.

Gupta, R. and Nanda, S. J. (2021). Solving dynamic many-objective tsp using nsga-iii equipped with svr-rbf kernel predictor. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 95–102.

Jiang, M., Huang, Z., Qiu, L., Huang, W., and Yen, G. G. (2017). Transfer learning-based dynamic multiobjective optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 22(4):501–514.

Jiang, S., Wang, Y., Hu, Y., Zhang, Q., and Yang, S. (2023). Vector autoregressive evolution for dynamic multi-objective optimisation.

Jiang, S. and Yang, S. (2014). A framework of scalable dynamic test problems for dynamic multi-objective optimization. In *2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, pages 32–39.

Jin, J. and Chen, Y. (2013). Var-based research on energy consumption in china. In *2013 International Conference on Computational and Information Sciences*, pages 1746–1749.

Jin, Y. and Branke, J. (2005). Evolutionary optimization in uncertain environments-a survey. *Trans. Evol. Comp*, 9(3):303–317.

Jonathan D. Cryer, K.-S. C. (2008). *Introduction*, pages 1–10. Springer New York, New York, NY.

Karkazan, K., Topcuoglu, H. R., and Sahmoud, S. (2023). A new prediction-based algorithm for dynamic multi-objective optimization problems. In *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*, pages 194–209. Springer.

Li, H. and Zhang, Q. (2009). Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302.

Li, L. D., Li, X., and Yu, X. (2008). Power generation loading optimization using a multi-objective constraint-handling method via pso algorithm. In *2008 6th IEEE International Conference on Industrial Informatics*, pages 1632–1637.

S. Huband, P. Hingston, L. B. and While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Sahmoud, S. and Topcuoglu, H. R. (2018). A type detection based dynamic multi-objective evolutionary algorithm. In *International Conference on the Applications of Evolutionary Computation*, pages 879–893. Springer.

Sahmoud, S. and Topcuoglu, H. R. (2019). Exploiting characterization of dynamism for enhancing dynamic multi-objective evolutionary algorithms. *Applied Soft Computing*, 85:105783.

Sahmoud, S. and Topcuoglu, H. R. (2020). Memory-assisted dynamic multi-objective evolutionary algorithm for feature drift problem. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Tian, Y., Cheng, R., Zhang, X., and Jin, Y. (2017). PlatEMO: A MATLAB platform for evolutionary multi-objective optimization. *IEEE Computational Intelligence Magazine*, 12(4):73–87.

Weizhen, H., Jiang, M., Gao, X., Tan, K. C., and Cheung, Y.-m. (2019). Solving dynamic multi-objective optimization problems using incremental support vector machine. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 2794–2799. IEEE.

Zhang, Q. and Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.

Zhu, X., Zhang, J., and Wu, Z. (2019). Many-objective power flow optimization problems based on an improved moea/d with dynamical resource allocation strategy. In *2019 Chinese Automation Congress (CAC)*, pages 3680–3685.