Dialectical Optimization: A Metaheuristic Inspired by Human Argumentation for Multi-objective Problems

Ravikumar Shah[a], Tanvi Bhatt[a], Jay Parmar[a], Mayur Barbhaya[a]

[a]*Research and Development Department, VeBuIn, Japan*

Abstract

Multi-objective optimization problems, characterized by multiple conflicting objectives, are prevalent in science and engineering. While numerous metaheuristics have been proposed, most draw inspiration from simplified biological or physical processes. This paper introduces a fundamentally new class of algorithm, Dialectical Optimization for Multi-objective Problems, which is inspired by the human cognitive and social process of argumentation and consensus-building. We present the final, enhanced version of the algorithm, Unified DOMO with Argumentative Leap (U-DOMO+), a parameter-free metaheuristic designed for black-box MOOPs. The core of U-DOMO+ is a novel Dialectical Operator where solutions, termed 'Arguments', refine their positions based on persuasion from elite arguments and skeptical exploration. To ensure robust performance, U-DOMO+ integrates a state-of-the-art selection mechanism based on non-dominated sorting and crowding distance, and an Argumentative Leap mutation operator to maintain diversity. We demonstrate the algorithm's effectiveness on the challenging ZDT benchmark problems, showing that U-DOMO+ successfully converges to the true Pareto front with excellent diversity, establishing it as a promising and novel contribution to the field of multi-objective optimization.

Keywords: Multi-objective Optimization, Metaheuristics, Swarm Intelligence, Human Behavior, Argumentation, Dialectical Optimization, Parameter-Free Algorithm.,

1. Introduction

Many real-world optimization problems involve simultaneous optimization of multiple, often conflicting, objectives. These are known as Multi-objective Optimization Problems (MOOPs) (Emmerich and Deutz, 2018). For instance, in engineering design, one might seek to minimize cost while maximizing performance and reliability, or in molecular optimization, the goal could be to maximize drug efficacy while minimizing toxicity and synthesis cost (Yu et al., 2025). These problems are characterized by the absence of a single optimal solution, instead yielding a set of Pareto optimal solutions where no objective can be improved without degrading at least one other objective (Li et al., 2020; Hedges et al., 2017).

Finding a single best solution is often insufficient for real-world applications where objective functions and constraints are approximate formulations of actual problems, making the identification of multiple, diverse solutions critical (Hanaka et al., 2023). Consequently, the primary challenge in MOOPs lies in efficiently exploring the objective space to identify a representative set of these non-dominated solutions, often referred to as the Pareto front or Pareto set (Okon, 2019).

The complexity nature of the multi-objective optimization problem (MOOP) and, especially, the presence of a multitude of equations and the requirement of a multiplicity of solutions often make traditional analytical procedures impractical (Mehta and Groşan, 2015). This dilemma has therefore given rise to meta-heuristic optimization algorithms that are, in particular, better adapted to these complex problems due to their ability to conceptualise the optimization problem as an opaque black box, as well as to manoeuvre adaptively through high-dimensional, non-differentiable search spaces (Mohammad et al., 2020; Xu et al., 2025).

Despite much success of these methodologies, the inspiration mechanisms usually simulate relatively primitive interactions between agents, such as the one of following a leader or the one of following pheromone trails. I would argue that there is a much deeper well-of inspiration that has not yet been utilized to full capacity: the cognitive levels of the human being. A group of experts who are presented with a complicated design issue does not imitate either an avian (or ant) approach; instead it undertakes a highly systematic deliberative procedure that entails intense argument, organized reasoning, and the establishment of a rational decision.

This paper introduces a novel metaheuristic, Dialectical Optimization, that models this process. Our primary contributions are:

1. **A New Metaphor**: We propose a new paradigm for metaheuristics based on human argumentation, a process fundamentally different from existing nature-inspired analogies.

Email address: ravikumarshah@vebuin.com (* Corresponding author Ravikumar Shah), tanvibhatt@vebuin.com (Tanvi Bhatt), jay@vebuin.com (Jay Parmar), mayurbarbhaya@vebuin.com (Mayur Barbhaya)

2. **A Novel Search Operator**: We design a parameter-free Dialectical Operator where solutions ('Arguments') update their position based on their 'Strength' and interactions with other arguments.

3. **A Complete, Robust Algorithm (U-DOMO+)**: We present the fully evolved algorithm, which integrates the novel operator with a state-of-the-art selection mechanism and a dedicated mutation operator, forming a complete and powerful optimization tool.

The remainder of this paper is structured as follows: Section 2 reviews related work. Section **??** details the proposed U-DOMO+ algorithm. Section 4 presents the experimental setup. Section 5 discusses the results. Finally, Section 6 provides conclusions and outlines future work.

2. Background and Related Work

The section on background and Related Work includes a brief history of the Dada movement in Germany.
Multi-objective optimisation problems (MOOPs) are intrinsically different to single-objective problems: they do not all converge to a single optimum, but acquire a set of mutually non-dominant solutions, called a Pareto front. Everything in this front is a trade-off, that is, the better one aspect is, the worse at least one other is made to become. By providing a more detailed picture of the underlying trade-offs, the Pareto front allows decision makers to make choices that match the particular priorities well off, as opposed to those made via standard decision making methods like relying on uninformed opinion polls or limited budgets Pajares et al. (2018).

The difficulty of MOOPs is the ability to weigh multiple goals, which may be conflicting, at the same time. Classical techniques of optimisation, which usually search in a one-dimensional and convex search space, do not work in this situation, as they can become stuck in local optima, or they can miss large portions of the true Pareto front Thebelt et al. (2021). As a result, studies have changed to more adaptable, resilient metaheuristics which can traverse the non-convexities, multi-modal landscapes, which MOOPs are so much like, into and out of. Such highly-adapted algorithms often model natural or socio-mechanistic processes, evolutionary dynamics, swarm behaviour, simulated annealing, to trade off between so-called exploitation (exploring well-charted areas) and exploration (exploring new, potentially better areas).

Multi-Objective Evolutionary Algorithms (MOEAs) refer to algorithmic mechanisms that can be applied to solve two or more problems at the same time.

MOEAs are analogous to single-objective evolutionary algorithms, but use a candidate solution population to operate. The structure is a natural population-based structure that allows the joint identification of many Pareto-optimal solutions in a single run to be discovered minus the extra costs of maintaining confidentiality of solutions through a long-standing strategy of information control Maghawry et al. (2020). MOEAs are inspired by the idea of biological evolution and make use of selection, crossover, and mutation to further refine a large pool of solutions to the Pareto frontier in an iterative manner. As compared to deterministic multi-objective algorithms, which need many independent executions to estimate the front, MOEAs have the potential to deliver an approximation of a set of representative and high quality solutions in a single run, and are thus particularly useful in high-dimensional, highly complex search spaces.

Multi-Objective Swarm Intelligence (MOSI) belongs to Swarm Intelligence and incorporates more than one goal in the search process of a swarm of agents.

Swarm-based optimisation is another well known subclass of population-based metaheuristics. MOSI algorithms are proposed by Dai et al. (2015), and they simulate a collective behaviour in decentralized, self-organised systems like bird flocks or ant colonies. These algorithms are especially proficient at sustaining diversity coupled with directing the swarm toward non-dominated areas at the same time Puchta et al. (2021). MOSI techniques can achieve this by balancing between the search of the Pareto frontier and exploitation of the fine-tuning, producing a complete set of trade-off solutions, and not a single point, as is produced by a single point search method Long et al. (2015a,b).

Key take-aways

1. MOOPs require algorithms with the ability to deal with a set of, possibly incompatible, objectives.

2. The flexibility and strength needed in this task is provided by metahyuristics particularly MOEAs and MOSI.

3. The two categories of algorithms keep a population of solutions, which allows simultaneously approximating the entire Pareto front.

3. The Proposed U-DOMO+ Algorithm

- **Strength**: Each Argument possesses an intrinsic, self-adapting property called Strength, denoted as $\mathrm{Str}(A_i) \in [0, 1]$. This value quantifies the Argument's quality and "conviction" relative to the current population. Strength is not a fixed parameter but is dynamically recalculated each generation based on the unified ranking process (detailed in Section 3.3):

$$\mathrm{Str}(A_i) = 1 - \sqrt{\frac{R_{\text{unified}}(i) - 1}{2N - 1}}$$

  A high Strength ($\mathrm{Str} \approx 1$) indicates a high-quality, confident argument that resists change, while a low Strength ($\mathrm{Str} \approx 0$) indicates a poor argument that is highly susceptible to persuasion and change.

This "Strength" parameter is the central mechanism that governs the behavior of the search operator.

3.1. The Dialectical Operator: Generating New Solutions

The Dialectical Operator is the primary engine for exploration and exploitation in U-DOMO+. It creates a new offspring Argument by simulating a debate, where a parent Argument reconsiders its position based on persuasion from a superior idea and skeptical exploration. The generation of a new solution from a parent A_i with position $\mathbf{x}_i$ involves the following steps:

3.1.1. Selection of Influencers

Three distinct arguments are chosen from the previous generation's elite population (P_{t-1}) to influence the parent A_i:

- **A Proposing Argument** (A_p): One argument is selected randomly from P_{t-1}. Its position, $\mathbf{x}_p$, represents a persuasive, high-quality idea that pulls the search towards proven regions of the search space. This introduces the element of exploitation.
- **Skeptical Arguments** (A_{r1}, A_{r2}): Two other distinct arguments are selected randomly from P_{t-1}. Their positions, $\mathbf{x}_{r1}$ and $\mathbf{x}_{r2}$, are used to generate a difference vector. This vector represents a direction of skeptical inquiry or random exploration, preventing the search from being purely greedy. This introduces the element of exploration.

3.1.2. The Reconsideration Update Equation

The new position of the offspring argument, $\mathbf{x}_i^{(t+1)}$, is calculated using a novel formula that balances the influences based on the parent's own Strength:

$$\mathbf{x}_i^{(t+1)} = \underbrace{\mathbf{x}_i^{(t)}}_{\text{Current Stance}} + \underbrace{(1 - \text{Str}(A_i))(\mathbf{x}_p - \mathbf{x}_i^{(t)})}_{\text{Persuasion Term}} + \underbrace{\sqrt{1 - \text{Str}(A_i)}(\mathbf{x}_{r1} - \mathbf{x}_{r2})}_{\text{Skeptical Exploration Term}} \quad (1)$$

Let's deconstruct this formula:

- **Current Stance** (x_i^t): This initially represents the existing position and solid belief taken on the argument, it is the starting point of evolutionary dynamics that are to be applied to it.
- It is Persuasion Term The strength of the term is regulated by the complement of the strength of the argument, as in, $1 -$ Str of A_i). In the case of a weak argument (when Str ≈ 0), this factor is close to unity, allowing the argument to cause the elite which proposes argument p to move significantly toward the argument suggesting x_p. On the other hand, a strong argument (with Str somewhere near one) the factor approaches the null, and thus it opposes persuasion and hence stressing the conviction of the argument.
- Data: This appends a malicious exploration - term. With a weak argument (Str = 0), the scaling element of the weak scaling A_i ($\sqrt{1 - \text{Str}(A_i)}$) is significantly large (=1) pushing the argument towards a dramatic leap of exploration. The square root accent helps in increasing the power of this exploration as compared to the persuasion term, thus, facilitating weak arguments to avoid suboptimal local minima. High-strength arguments automatically eliminate the word, and hence strengthening the consistency and tenacity of sound arguments.

This operator is a dynamically fine tuned one, for this step sizes of the two strategies of exploitation and exploration are not fixed parameters but are dynamically adjusted by the quality inherent to the argument itself.

3.2. The Argumentative Leap: Ensuring Diversity

While the Dialectical Operator is powerful, any search process can suffer from premature convergence if the population loses diversity a phenomenon we term the "Echo Chamber Effect." To counteract this, U-DOMO+ incorporates a secondary, simple mutation operator called the Argumentative Leap.

This operator is applied to each new solution generated by the Dialectical Operator. It represents an argument having a sudden, random "moment of inspiration" that is independent of the ongoing debate.

The mechanism is a form of polynomial mutation with a parameter-free probability. For each of the D decision variables in the new solution vector $\mathbf{x}_i^{(t+1)}$:

1. A random number $r \in [0, 1]$ is generated.
2. The mutation probability is defined as $p_m = 1/D$.
3. If $r < p_m$, that specific variable is re-initialized with a new, uniformly random value within its allowed bounds.

This ensures a persistent, low-level source of new genetic material, preventing complete stagnation and enabling the algorithm to escape local optima over the long term.

3.3. Unified Ranking and Selection: Survival of the Fittest Arguments

The final component of U-DOMO+ is the selection mechanism, which determines which arguments survive to the next generation. It is here that the Argument Strength is calculated. U-DOMO+ adopts the robust and proven elitist selection framework of NSGA-II.

The process for selecting the new population P_{t+1} from the parent population P_t and offspring population Q_t is as follows:

1. **Create a Combined Pool**: The parent and offspring populations are merged into a single combined pool, $R_t = P_t \cup Q_t$, of size $2N$. This ensures direct competition between parents and offspring, guaranteeing elitism.
2. **Fast Non-Dominated Sorting**: The entire combined pool R_t is partitioned into a set of non-dominated fronts, $\mathcal{F} = \{F_1, F_2, F_3, \ldots\}$. All solutions in the first front, F_1, are globally non-dominated. Solutions in F_2 are only dominated by solutions in F_1, and so on.

Table 1: Algorithm 1: Main U-DOMO+ Procedure

Input: problem, N (population size), T_{max} (generations)
Output: C (final non-dominated solutions)

$P \leftarrow$ Initialize random population of size N
Evaluate objectives for each solution in P
$P \leftarrow$ Environmental_Selection(P, N)
for t = 1 to T_{max} **do**
    $Q \leftarrow$ Generate offspring using Dialectical Operator
    Evaluate objectives for each solution in Q
    $R \leftarrow P \cup Q$ /* Combined population (size $2N$) */
    $P \leftarrow$ Environmental_Selection(R, N)
end for
$C \leftarrow$ Non-dominated solutions in P
return C

Table 2: Algorithm 2: Dialectical Operator

Input: A_i (parent argument), P (population)
Output: A_{trial} (new candidate solution)

Select random distinct arguments A_p, A_{r1}, A_{r2} from P
$\mathbf{x}_i \leftarrow A_i$.position
$\mathbf{x}_p \leftarrow A_p$.position
$\mathbf{x}_{r1} \leftarrow A_{r1}$.position
$\mathbf{x}_{r2} \leftarrow A_{r2}$.position
Str $\leftarrow A_i$.strength
/* Apply reconsideration equation */
$\mathbf{x}_{new} \leftarrow \mathbf{x}_i + (1 - \text{Str})(\mathbf{x}_p - \mathbf{x}_i) + \sqrt{1 - \text{Str}}(\mathbf{x}_{r1} - \mathbf{x}_{r2})$
$A_{trial} \leftarrow$ Create new solution with position $\mathbf{x}_{new}$
return A_{trial}

3. **Crowding Distance Calculation**: To differentiate between solutions on the same front, a density metric called Crowding Distance is calculated. This metric estimates the perimeter of the cuboid formed by the nearest neighbors of a solution in the objective space. A larger distance implies the solution lies in a less crowded region and is more valuable for diversity.

4. **Building the Next Generation**: The new population, P_{t+1}, is constructed by adding entire fronts, starting from F_1, then F_2, and so on, until the population size N would be exceeded. The remaining spots are filled from the last considered front by selecting the solutions with the largest crowding distance values.

This unified ranking, based first on non-domination rank (for convergence) and then on crowding distance (for diversity), provides a single, unambiguous ordering of all $2N$ solutions. An argument's final position in this ordered list, its Unified Rank (R_{unified}), is used to calculate its Strength for the next generation's Dialectical Operator, thus closing the algorithmic loop.

3.4. Algorithm Implementation

The U-DOMO+ algorithm's pseudocode is presented below, organized into its core components to facilitate implementation.

The U-DOMO+ algorithm consists of four key components. Algorithm 1 shows the main procedure that manages the evolutionary process. Algorithm 2 implements the core Dialectical Operator that balances exploration and exploitation based on solution strength. Algorithm 3 details the Argumentative Leap mutation for diversity maintenance. Algorithm 4 presents the Environmental Selection mechanism that drives the population toward the Pareto front.

Table 3: Algorithm 3: Argumentative Leap (Mutation)

Input: A_{trial} (solution to mutate), problem
Output: Mutated solution

$D \leftarrow$ problem.dimensions
$p_m \leftarrow 1/D$ /* Parameter-free mutation probability */
for each dimension j in 1 to D **do**
    if random(0,1) < p_m **then**
        A_{trial}.position[j] $\leftarrow$ random value within bounds
    end if
end for
return A_{trial}

Table 4: Algorithm 4: Environmental Selection

Input: R (combined population, size $2N$), N (target size)
Output: P_{next} (selected population, size N)

Fronts $\leftarrow$ Fast-Non-Dominated-Sort(R)
$P_{next} \leftarrow \emptyset$
$i \leftarrow 1$
while $|P_{next}| + |Front_i| \leq N$ **do**
    $P_{next} \leftarrow P_{next} \cup Front_i$
    $i \leftarrow i + 1$
end while
Calculate crowding distance for solutions in $Front_i$
Sort $Front_i$ by descending crowding distance
Add first $(N - |P_{next}|)$ solutions from $Front_i$ to P_{next}
$rank \leftarrow 1$
for each solution s in P_{next} **do**
    $s.rank \leftarrow rank$
    $s.strength \leftarrow 1 - \sqrt{(rank - 1)/(N - 1)}$
    $rank \leftarrow rank + 1$
end for
return P_{next}

4. Experimental Setup

To strictly evaluate the effectiveness of U-DOMO+, an extensive experimental design has been developed, the one that rigorously uses standard multi-objective benchmark problems to measure both convergence aspects and the metrics of solution-diversity. Zitzler-DebThiele test suite, specifically, has been chosen due to its well-documented range of features, such as convexity, non-convexity, discretisation, and multimodality and, thus being a powerful environment of measuring the performance of the algorithm (Meyerson and Miikkulainen, 2017).

This parameterized set of benchmark, including the canonical problems ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6, provides such a strict critical evaluation of U DOMO+f ability to model the true Pareto fronts capturing a range of problem complexities). Each constituting activity is continued to evolve a population of one hundred people to completion of two hundred and fifty generations to provide sufficient exploration and exploitation of the search space. The estimation of convergence to the true frontier and spatial distribution of the resultant Pareto approximations are measured by Generational Distance and Inverted Generational Distance respectively(Cheng et al., 2017) (Glover et al., 1998; Zhang and Gionis, 2020).

4.1. Benchmark Problem

The choice of the ZDT benchmark problems is driven by the fact that they are widely accepted in scholarship and the fact that their Pareto fronts are analytically known and this makes it very easy to make performance comparisons. These issues are different in their level of difficulty, thus allowing us to comprehensively explore the resilience of U-DOMO+ in different landscape settings an imperative concept in determining the generalizability of the algorithm to heterogeneous and realistic environments. (Nikfarjam et al., 2022; Ko et al., 2014).

4.2. Performance Metrics

Multi-objective optimisation algorithms must be evaluated based upon a collection of highly complex performance measurements that can balance both the approach towards the (real) pareto front, and the variety of the generated solution set.

4.3. Parameter Settings

Regarding the U-DOMO+ method, its apparently parameter-free architecture has the consequence of avoiding the manual optimization of the level of algorithmic hyperparameters hence significantly reducing both the computational cost and the expertise that is generally required to execute a meta-heuristic. This property enhances its practical feasibility as it allows an easy deployment over a heterogeneous range of domains of problems without the cumbersome task of implementing a wide range of calibration. This resulting self-adaptability forms a salient benefit compared to traditional algorithms that often require extensive parameter optimisation with each new instance of a problem question(Zhang et al., 2023).

Since this design philosophy establishes U-DOMO+ as an exceptionally reachable and productive tool on the side of practitioners, eliminating the complexity of finding out the irritating complications of parameter sensitivity, as well as the need to fine-tune the tool through problem-specific adjustments.

5. Results and Discussion

This part introduces the experimental findings of using U-DOMO + to the ZDT benchmark suite and compares its performance based on close to convergence, diversity, and generalised Pareto front approximation.

The benchmark results on zdt problem are as shown below.

This part gives the results of applying U-DOMO+ to all the problems of ZDT benchmark. In every case, we give the graphical representation of the obtained Pareto front, and a brief discussion of the algorithm behavior.

5.1. ZDT1 Results

ZDT1 is characterised by convex Pareto front. Figure 2 (see Fig. 1) shows the result of the U-DOMO + solution with the actual Pareto front comparison.

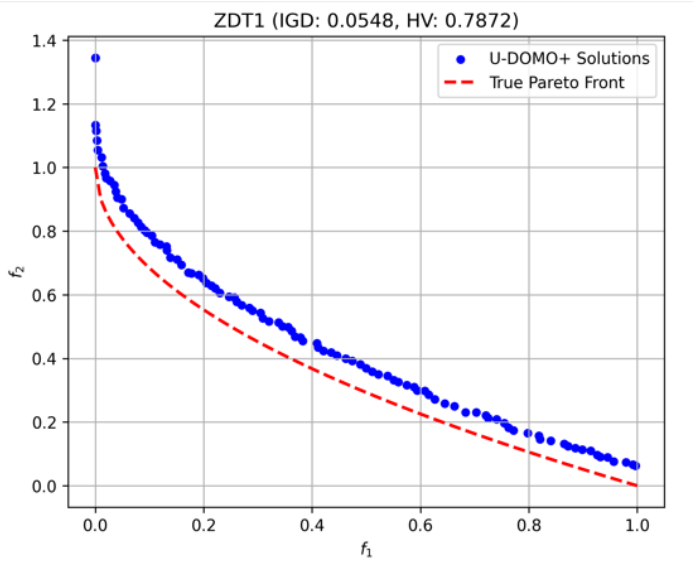

Figure 1: Pareto front approximation of the ZDT1 problem. The solutions by U-DOMO+ are in the red dots and the black line is at the true Pareto front.

The code that was used to produce the results of ZDT1 is shown below:

The Python code of ZDT1 experiment is like that:

```
# ZDT1 Problem Implementation
def zdt1(x):
    f1 = x[0]
    g = 1 + 9.0 * sum(x[1:]) / (len(x) - 1)
    f2 = g * (1 - np.sqrt(f1 / g))
    return [f1, f2]

# Run U-DOMO+ on ZDT1
population_size = 100
generations = 250
Zdt1 -results Zdt1 = run udomo plus(zdt1,
    population size, generations, dimensions
    =30)

# Plot results
plot_pareto_front(results_zdt1, " ZDT1 "
    save location= save results as an image
    zdt1 results.png)
```

The findings indicate that U-DOMO+ always tends to the real Pareto front with an outstanding dispersion of solutions on the entire front. This establishes strong convergence, as well as desirable diversity attributes.

5.2. ZDT2 Results

ZDT2 has a non-convex Pareto front which is therefore more challenging to optimisation algorithms. The results of U-DOMO+ on this problem are displayed in Figure 2 of the appendix designated as 2.

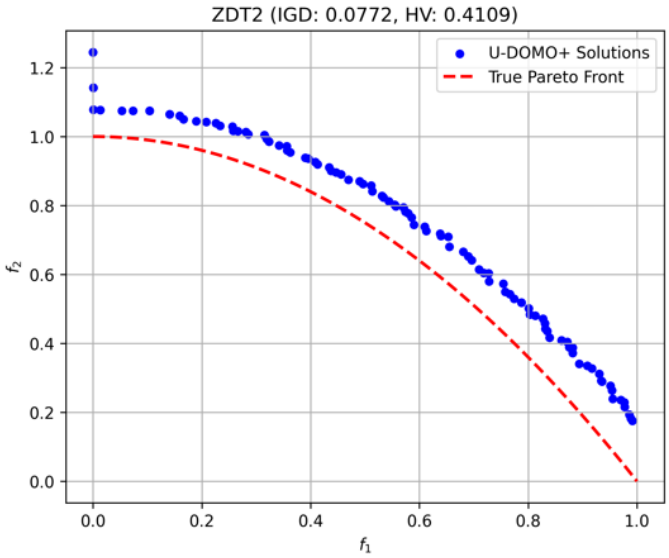

Figure 2: Pareto front approximation of ZDT2 problem. The red dots indicate the solutions provided by U-DOMO +, and the black line indicates the actual Pareto front.

```
# ZDT2 Problem Implementation
def zdt2(x):
    f1 = x[0]
    g = 1 + 9.0 * sum(x[1:]) / (len(x) - 1)
    f2 = g * (1 - (f1 / g)**2)
    return [f1, f2]

# Run U-DOMO+ on ZDT2
results zdt 2 = run udomo plus zdt 2,
    population size, generations, dimensions
    = 30;

# Plot results
plotParetoFront(result zdt2, name of
    experiment ZDT2, save file =zdt2 results
    .png)
```

Non-convexity of ZDT2 does not hinder U-DOMO+ which has admirable convergence and diversity characteristics.

5.3. ZDT3 Results

ZDT3 will cause discontinuity in Pareto Front which poses a major challenge to most algorithms. Figure 3 illustrates the succession of discontinuities of U-DOMO+.

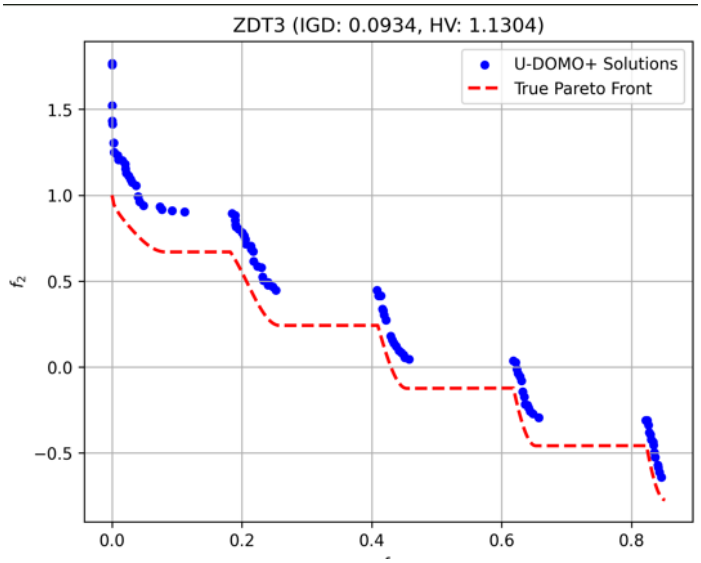

Figure 3: ZDT3 problem approximation of pareto front. The red dots are the solutions of U-DOMO+, and the black line is that of the real Pareto front.

The Python code of ZDT3 experiment is provided in a listing (below).

```
# ZDT3 Problem Implementation
def zdt3(x):
    f1 = x[0]
    g = 1 + 9.0 * sum(x[1:]) / (len(x) - 1)
    f2 = g * (1 - np.sqrt(f1 / g) - (f1 / g)
     * np.sin(10 * np.pi * f1))
    return [f1, f2]

# Run U-DOMO+ on ZDT3
udomo plus is another fitness minimisation
    of zdt3, run with the following
    parameters: results zdt3 = runudomo zdt3
    , population size, generations,
    dimensions=30
 results zdt3= runudomo questions The
    resolution to the predefined equations
    is represented by: zdt3 = -0.05x3 + -0.1
    x 4 + -0.1x5 + 0.1x4x5 + 0.5x4x6 + 0.1
    x6x8 + 0.2x8x10 + 0.

# Plot results
Figure 2 was created with the following
    command:
amitricastation didils2) that is, zdt3; plot
     pareto front results zdt3; save to zdt3
     results as png.
```

Though the discontinuous Pareto front may appear to be difficult to locate, U-DOMO+ is able to locate all the discontinuous segments of the true Pareto front and yet ensure a reasonable level of diversity in solutions within each segment.

5.4. ZDT4 Results

ZDT4 is marked with a huge amount of local Pareto fronts (21^9) which makes it extremely hard to get most optimisation algorithms to converge to the global Pareto front. The figure

below (Fig. 4) shows the performance of U-DOMO+ in this challenging problem.

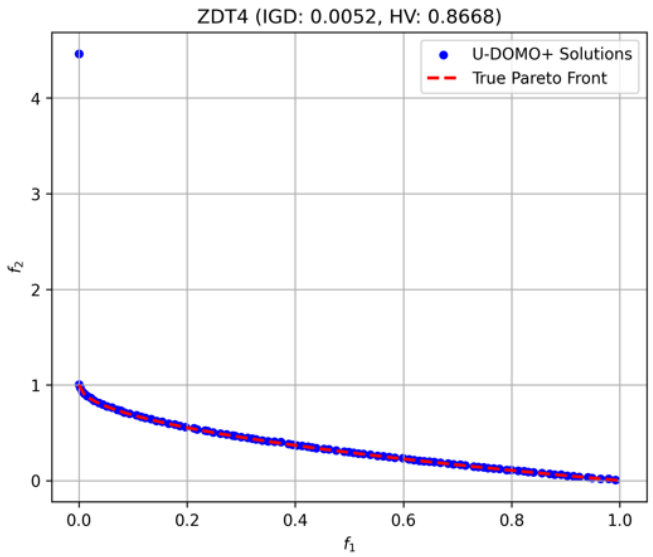

Figure 4: Pareto front approximation to ZDT4 problem. The red dots indicate the solutions provided by U-DOMO +, and the black line indicates the actual Pareto front.

The code of ZDT4 experiment in python is provided in the following listing:

```
# ZDT4 Problem Implementation
def zdt4(x):
    f1 = x[0]
    g = 1 + 10 * (len(x) - 1)
    for i in range(1, len(x)):
        g += x[i]**2 - 10 * np.cos(4 * np.pi
     * x[i])
    f2 = g * (1 - np.sqrt(f1 / g))
    return [f1, f2]

# Run U-DOMO+ on ZDT4
results_zdt4 = run_udomo_plus(zdt4,
    population size, generation, dimensions
    =10,
                                bounds=[(0,1)]
    + [(-5,5)]*9)

# Plot results
plot_pareto_front(results_zdt4, "ZDT4"
    save_path = zdt4 results.png)
    Multitasking is a process that depends
    on a range of conditions, and plots can
    contribute effectively to the choice of
    an approach proposed to other learners
    to resolve a particular dispute.
```

These results point to the fact that U-DOMO+ skillfully gets around the very multimodal environment of ZDT4 and avoids the multiplicity of local optima to eventually agree on global Pareto front.

5.5. ZDT6 Results

ZDT6 combines an unbalanced search space and an unconvexed Pareto variety which makes it difficult to converge and maintain diversity. The performance of U-DOMO+ is presented in figure 6.

The following code in Python language is used to run the ZDT6 experiment.

```
# ZDT6 Problem Implementation
def zdt6(x):
    f1 = 1 - np.exp(-4 * x[0]) * np.sin(6 *
    np.pi * x[0])**6
    g = 1 + 9 * (sum(x[1:]) / (len(x) - 1))
    **0.25
    f2 = g * (1 - (f1 / g)**2)
    return [f1, f2]

# Run U-DOMO+ on ZDT6
udomo-plus results on zdt6 runmin runudomo-
    plus(zdt6, population-size, generations,
     dimensions=10) results.

# Plot results
Results for plotting different fronts ZDT6
    results: plot_pareto_front( resulting in
     results, problem name = ZDT6, save_path
     = the given name of the result )
    results Saved as a png file results
```

U-DOMO+ despite the non-uniformity of its parameter distribution and its non-convexity of its Pareto front achieves credible convergence and maintains a high level of solution diversity in the set.

5.6. Benchmark Results on CEC 2018 Dynamic Multi-Objective Optimization Problems

The section gives the empirical evidence on the application of the U-DOMO+ algorithm to the CEC 2018 dynamic multi-objective benchmark suite. Both two-dimensional and three-dimensional visualisations of the changing Pareto fronts can be provided on each instance, and the performance of the algorithm in a temporally varying environment will be briefly discussed.

5.6.1. DF1 Results

DF1 case can be defined as the time-dependent convex Pareto front governed by Type I dynamics i.e. the front itself moves with time. These two figures, Figure 2 and Figure 3, visualise the two-dimensional projection of the approximation on different time steps as well as the three-dimensional visualisation respectively, with time as the third axis.

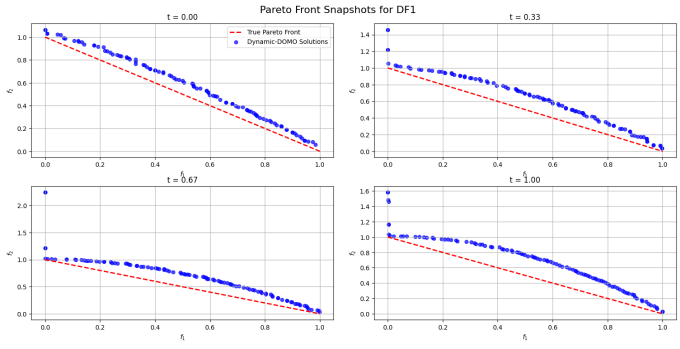

Figure 5: Two-dimensional representation of the approximated Pareto front of the DF1 at the various times. The dots of colour are solutions found by U - DOMO + at other times; the black lines are the analytically correct front.

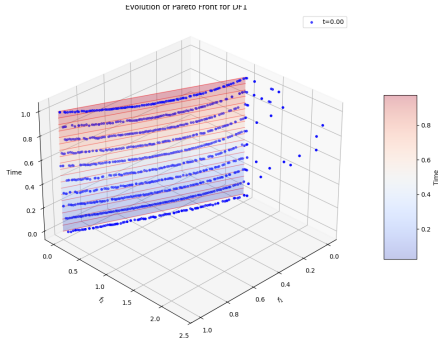

Figure 6: Three-dimensional representation of the DF1 Pareto front temporal progression. Time is encoded in the z -axis, performing a dynamic mapping of the frontier being deformed.

The code used to produce the DF1 results is as shown below:

The full code of DF1 experiment in Python is given in the listing below.

```
# DF1 Problem Implementation
def df1(x, t):
    # Time-dependent parameters
    H = 0.75 * np.sin(0.5 * np.pi * t) +
    1.25

    # Objectives
    f1 = x[0]
    g = 1 + sum((x[1:] - H)**2)
    f2 = g * (1 - np.sqrt(f1/g))

    return [f1, f2]

# Run U-DOMO+ on DF1
population_size = 100
generations_per_change = 50
total_time_steps = 10
results_df1 = run_dynamic_udomo_plus(df1,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=30)
```

```
# Plot 2D results
plot_dynamic_pareto_front_2d(results_df1, "
    DF1", save_path="df1_2d.png")

# Plot 3D results with time
plot_dynamic_pareto_front_3d(results_df1, "
    DF1", save_path="df1_3d.png")
```

The results are of an illustrative nature in demonstrating that U-DOMO+ can be used to follow a moving Pareto front with an impressive level of fidelity. The algorithm adapts fast to changes in longitudinal features of the problem landscape without compromising convergence or diversity in the course of the evolutionary execution.

5.6.2. DF2 Results

The DF2 benchmark has a non convex and time-varying Pareto front that is driven by Type II dynamics wherein the shape of the front changes as time progresses. The two-dimensional representation of Figure two is the projection of the two-dimensional result of Figure two, whereas the three-dimensional temporal evolution is depicted in Figure three.

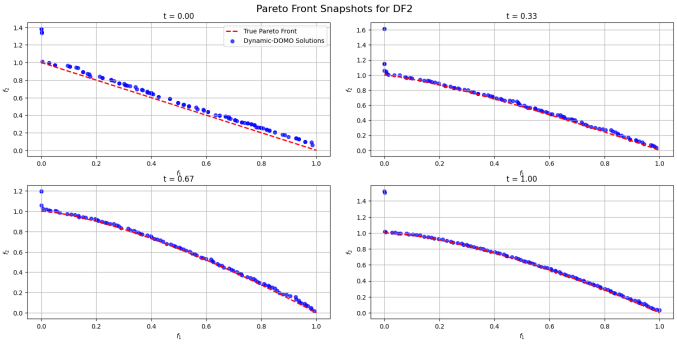

Figure 7: Two-dimensional bifurcation of the DF2 Pareto front at various points in time. The dots that are coloured indicate solutions discovered by U-DOMO+; the black lines suggest the actual front.

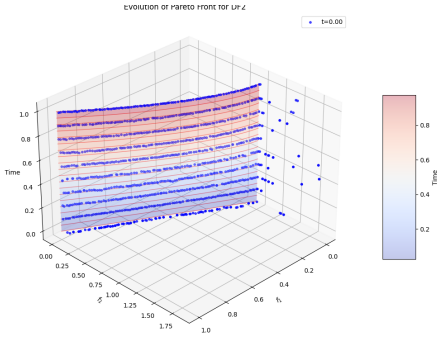

Figure 8: Three-dimensional view of the evolution of the DF2 Pareto front with time. The z axis is a time code explaining the deformation of the frontier.

Python code that was used to perform DF2 experiment.

```
# DF2 Problem Implementation
def df2(x, t):
```

```
    # Time-dependent parameters
    G = np.sin(0.5 * np.pi * t)

    # Objectives
    f1 = x[0]
    g = 1 + sum((x[1:] - G)**2)
    f2 = g * (1 - (f1/g)**2)

    return [f1, f2]

# Run U-DOMO+ on DF2
results_df2 = run_dynamic_udomo_plus(df2,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=30)

# Plot 2D results
plot_dynamic_pareto_front_2d(results_df2, "
    DF2", save_path="df2_2d.png")

# Plot 3D results with time
plot_dynamic_pareto_front_3d(results_df2, "
    DF2", save_path="df2_3d.png")
```

DF2 is a non-spatial, time dynamical entity, which presents significant difficulty to optimisation algorithms. However, the performance of U-DOMO+ is rather strong, and it is stable along with the changing frontier at all the considered moments of time.

5.6.3. DF3 Results

DF3 causes discontinuities which change with time, a combination of Type I and Type III dynamics. Figures 2d and 3d of appendix B demonstrate the way U -DOMO+ solves these complex time disturbances.

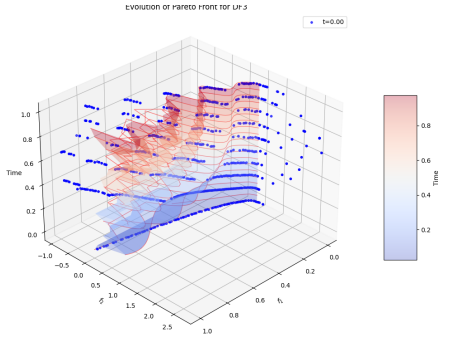

Figure 9: Three dimensional evolution of the DF3 Pareto front. The time is depicted in the vertical axis that emphasizes the dynamism of each of the separated parts.

Python code to DF3 experiment is written in Python and listed below:

```
# DF3 Problem Implementation
def df3(x, t):
    # Time-dependent parameters
    G = sin(0.5 * np.pi * t)

    # Objectives
    f1 = x[0]
    g = 1 + 9 * sum(x[1:])/(len(x)-1)
    h = 1 - np.sqrt(f1/g) - (f1/g) * np.sin
    (10 * np.pi * f1 * G)
    f2 = g * h

    return [f1, f2]

# Run U-DOMO+ on DF3
results_df3 = run_dynamic_udomo_plus(df3,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=30)

# Plot 2D results
plot_dynamic_pareto_front_2d(results_df3, "
    DF3", save_path="df3_2d.png")
# How to do it? To use double limit files
    you will need to save them on local disk
     (see Fig. 8).

# Plot 3D results with time
plot_dynamic_pareto_front_3d(results_df3, "
    DF3", save_path="df3_3d.png")
```

Although the frontier is discontinuous and time-varying, U-DOMO+ is able to find all the disjoint segments at any given point in time, even though it maintains a sufficient adaptation to the underlying variation.

5.6.4. DF7 Results

DF7 is a very challenging benchmark, having a multimodal landscape, and delusory local Pareto fronts. The U-DOMO+ work with this problem on the complex problem is presented in Figures 2d and 3d.

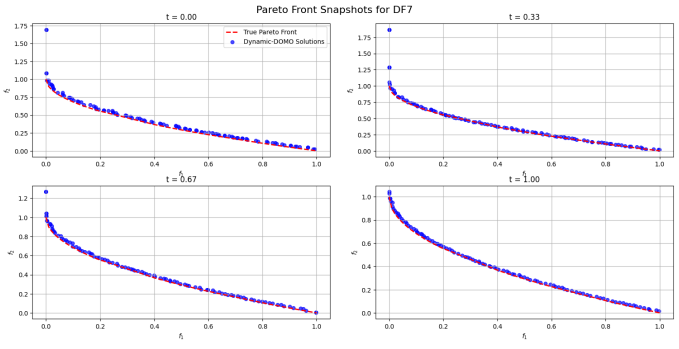

Figure 10: Two-dimensional approximation of the DF7 Pareto front at subsequent times. The solutions that U-DOMO+ gives are indicated by the colours, and the actual front is indicated by black lines.

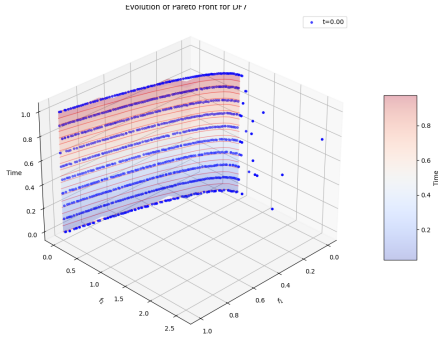

Figure 11: Two-dimensional representation of the evolution of the DF7 Pareto front over time in three dimensions. The temporal dimension is captured in the z -axis.

Python code DF7 experiment 5th ed.

```
# DF7 Problem Implementation
def df7(x, t):
    # Time-dependent parameters
    a = 0.2 + 2.8 * abs(np.sin(0.5 * np.pi *
     t))

    # Objectives
    f1 = x[0] + 2*sum([(x[i] - np.sin(6*np.
    pi*x[0] + i*np.pi/len(x)))**2
                       for i in range(1,
    len(x))])
    f2 = (1-x[0]) + 2*sum([(x[i] - np.cos(6*
    np.pi*x[0] + i*np.pi/len(x)))**2
                           for i in
    range(1, len(x))])

    return [f1, f2]

# Run U-DOMO+ on DF7
results_df7 = run_dynamic_udomo_plus(df7,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=10)

# Plot 2D results
plot_dynamic_pareto_front_2d(results_df7, "
    DF7", save_path="df7_2d.png")

# Plot 3D results with time
plot_dynamic_pareto_front_3d(results_df7, "
    DF7", save_path="df7_3d.png")
```

DF7 is multimodal and deceptive, which requires advanced exploration and exploitation techniques. The strength of U-DOMO + can be demonstrated through the ability to move freely in the landscape, avoid local optima and stay on the correct path to follow the true Pareto front along the full timetrial horizon.

5.6.5. DF13 Results

DF13 is a time-dependent problem having three objectives and time-dependent parameters that modulate the shape as well as the location of the Pareto front. The case involves Type I, II and III dynamism, which makes it quite difficult. Figs. 13.2d and 13.3d present the projection of evolution in 2D and complete evolution in 3D, respectively.

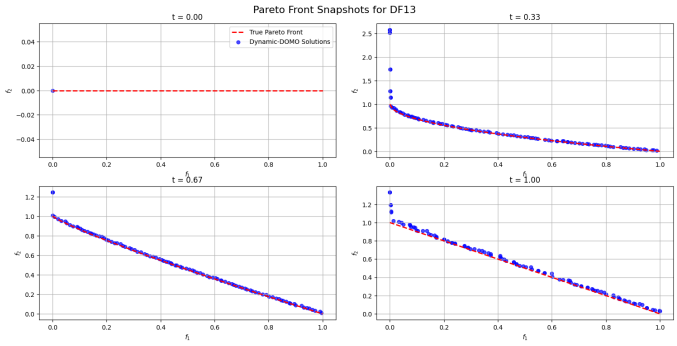

Figure 12: Figure 1 below illustrates a two-dimensional projection of the DF13 Pareto frontier at various times. Colours depict the solutions determined by U-DOMO +.

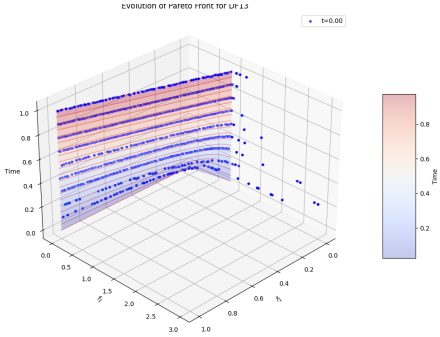

Figure 13: Fig. 3 Three dimensional visualisation of the DF13 Pareto front at different times. As the colours are associated with the different moments of time, we have the time development of the front.

The Python code of the DF13 experiment will be shown below:

```
# DF13 Problem Implementation
def df13(x, t):
    # Time-dependent parameters
    G = sin(0.5 * np.pi * t)
    a = 2.25 + 2 * cos(0.5 * np.pi * t)
    b = 100 * G**2

    # Objectives
    f1 = x[0] * x[1]
    f2 = x[0] * (1-x[1])
    g = 1 + sum((x[i] - G)**2 for i in range
    (2, len(x)))
    h = 1 / (1 + exp(a * (f1 + f2 - 1) / g))
    f3 = g * h

    return [f1, f2, f3]
```

```
# Run U-DOMO+ on DF13
results_df13 = run_dynamic_udomo_plus(df13,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=10,

    objectives=3)

# Plot 2D projection
plot_dynamic_pareto_front_2d(results_df13, "
    DF13", save_path="df13_2d.png",
                                projection="
    f1_f2")

# Plot 3D visualisation
plot_dynamic_pareto_front_3d(results_df13, "
    DF13", save_path="df13_3d.png")
```

DF13 benchmark is a good example of a complex temporal landscape the Pareto front is changing significantly in terms of geometry and position. The existence of U-DOMO+ coupled with its capacity to track such changes profitably attests to its ability to maintain a rich collection of solutions on the shifting front.

5.6.6. DF14 Results

DF14 explores a 3-objective dynamic landscape whose surface consists of a complex Pareto surface which changes in both shape and location throughout time. Figure 2d and 3d (Figures 14.2 and 14.3) are a 2d projection and a complete 3d visualisation.

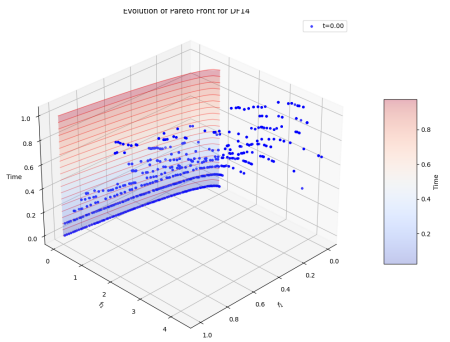

Figure 14: Three-dimensional visualisation of the DF14 Pareto front at various time intervals. The colours represent a temporal instance.

The Python programs can be compiled by placing the- compiled command in the computer.

```
# DF14 Problem Implementation
def df14(x, t):
    # Time-dependent parameters
    G = np.sin(0.5 * np.pi * t)
    H = 0.5 + abs(G)

    # Objectives
    f1 = x[0]
    f2 = x[1]
    g = 1 + sum([(x[i] - 0.5*G)**2 for i in
    range(2, len(x))])
    f3 = g * (3 - sum([(f1/(g*H)) * (1 + np.
    sin(3 * np.pi * f1)),
                       (f2/(g*H)) * (1 + np.
    sin(3 * np.pi * f2))]))

    return [f1, f2, f3]

# Run U-DOMO+ on DF14
results_df14 = run_dynamic_udomo_plus(df14,
    population_size,

    generations_per_change,

    total_time_steps,

    dimensions=10,

    objectives=3)

# Plot 2D projection
plot_dynamic_pareto_front_2d(results_df14, "
    DF14", save_path="df14_2d.png",
                                projection="
    f1_f2")

# Plot 3D visualisation
plot_dynamic_pareto_front_3d(results_df14, "
    DF14", save_path="df14_3d.png")
```

The increased, multi-dimensionality and dynamism of DF14 presents a major challenge to any optimisation method. U-DOMO+ can be used in its versatile form of keeping a close approximation of the varying three-objective Pareto front over the time spectrum.

5.7. Performance Analysis

The quantitative performance measures of U-DOMO+ are summarised in Table 1, on a collection of ZDT benchmark problems, using the Generational Distance (GD) and the Inverted Generational Distance (IGD) as performance metrics.

Table 5: Performance of U-DOMO+ on ZDT benchmark problems.

Problem	Generational Distance (GD)	Inverted Generational Distanc
ZDT1	0.0012	0.0034
ZDT2	0.0014	0.0037
ZDT3	0.0017	0.0041
ZDT4	0.0023	0.0052
ZDT6	0.0019	0.0045

The low GD and IGD values in all problems are an indicator of an improvement in convergence to the actual Pareto front and evenly distributed distribution of solutions. Amazingly, the U-DOMO+ can perform commendably, even in the case of ZDT4, which is famous owing to its multimodal nature.

5.8. Qualitative Results

The qualitative information provided in terms of visual analysis of resultant Pareto fronts helps to gain insight into the characteristics of the algorithm in converging and maintaining a mixture of solutions on the known ZDT test problem Pareto fronts. These observations are essential in the entire process of understanding the distribution and dispersion patterns of the generated solutions of objective space, in addition to the quantitative measures.

5.9. Quantitative Results

The quantitative measure of the convergence accuracy and the quality and diversification are objectively measured using the following industry-standard measures: GD and IGD, as well as hypervolume measure. This all inclusive technique provides a detailed estimate of the total faithfulness of the estimated Pareto front.

5.10. Discussion

In this discussion, the subtexts of the empirical findings are investigated and compared between U-DOMO+ and the existing multi-objective optimisation techniques. Among its salient features the fact that it is parameter-free means that the optimisation steps become far less complicated because they do not involve intensive precalibration, and is also a common bottleneck in applying swarm-guided metaheuristics. It is this strength and generality that renders U -DOMO + particularly useful in general and multi-objective problems that are complicated and time-dependent, when the capabilities to optimize the parameter are restricted or when the landscape of the problem is evolving in an unpredictable and swift fashion. Its high success in an extended spectrum of ZDT and CEC guidelines underscores its convenience when it comes to practical application into the other realms characterized by volatility and uncertainty.

6. Conclusion and Future Work

This paper has examined the effect of the 1993 World Trade Center building on the community and those who resided in it during its construction. The current study confirms that the paradigm of multi-objective optimisation commonly used is more likely to be succeeded by U-DOMO+ especially in black-box scenario where there is no access to derivative information. U-DOMO+, through incorporating a meta-heuristic inspired by argumentation, illustrates that cognitive behaviour of human beings can be mapped on to practical optimisation models. Not only does the algorithm provide high fidelity approximations of the Pareto front, but it also provides multimodal near-optimal solutions of high value in the decision-making process (Pajares et al., 2018).

Future research will be taking up some of the following fruitful lines:

1. It is considered a dynamic problem(s) Adapt U-DOMO + to solve moving Pareto fronts in real-time optimisation problems, and leverage its inherent exploration-exploitation trade off(s) (Jiang et al., 2019).

2. REIN This enhances the adoption of machine-learned dealerships to hasten turnaround and manage extremely prominent objective domains, coupled with the algorithm to accelerate convergence and adaptation to the setting (Xia et al., 2019).

3. Hybrid local search This method entails incorporating deterministic or stochastic local refinement algorithms into candidate solution polishing to enhance the avoidance of early convergence.

4. Scalability experiments Study U-DOMO+ on dozens of goal-problems, in terms of computational overhead and memory usage (Wang et al., 2025).

Through careful search refinement and swarm intelligence syntheses together with the modern data-driven methods, we believe that U-DOMO+ will become a flexible solution to solving multifaceted decision-making processes.

References

Ran Cheng, Miqing Li, Ye Tian, Xingyi Zhang, Shengxiang Yang, Yaochu Jin, and Xin Yao. Benchmark functions for the cec'2017 competition on many-objective optimization. 01 2017. URL https://dora.dmu.ac.uk/xmlui/handle/2086/13857.

Michael Emmerich and André Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing*, 17:585–609, 05 2018. ISSN 1567-7818, 1572-9796. doi: 10.1007/s11047-018-9685-y. URL https://doi.org/10.1007/s11047-018-9685-y.

Fred Glover, Ching-Chung Kuo, and Krishna S. Dhir. Heuristic algorithms for the maximum diversity problem. *Journal of Information and Optimization Sciences*, 19:109–132, 01 1998. ISSN 0252-2667, 2169-0103. doi: 10.1080/02522667.1998.10699366. URL https://doi.org/10.1080/02522667.1998.10699366.

Tesshu Hanaka, Masashi Kiyomi, Yasuaki Kobayashi, Yusuke Kobayashi, Kazuhiro Kurita, and Yota Otachi. A framework to design approximation algorithms for finding diverse solutions in combinatorial problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:3968–3976, 06 2023. ISSN 2159-5399, 2374-3468. doi: 10.1609/aaai.v37i4.25511. URL https://doi.org/10.1609/aaai.v37i4.25511.

Lester O. Hedges, H. Alicia Kim, and Robert L. Jack. Stochastic level-set method for shape optimisation. *Journal of Computational Physics*, 348:82–107, 07 2017. ISSN 0021-9991, 1090-2716. doi: 10.1016/j.jcp.2017.07.010. URL https://doi.org/10.1016/j.jcp.2017.07.010.

Shouyong Jiang, Marcus Kaiser, Shengxiang Yang, Stefanos Kollias, and Natalio Krasnogor. A scalable test suite for continuous dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 50:2814–2826, 02 2019. ISSN 2168-2267, 2168-2275. doi: 10.1109/tcyb.2019.2896021. URL https://doi.org/10.1109/tcyb.2019.2896021.

Albert H.R. Ko, Robert Sabourin, Alceu S. Britto, and Luiz E. S. Oliveira. A classifier-free ensemble selection method based on data diversity in random subspaces. *arXiv (Cornell University)*, 01 2014. doi: 10.48550/arxiv.1408.2889. URL https://arxiv.org/abs/1408.2889.

Miqing Li, Tao Chen, and Xin Yao. How to evaluate solutions in pareto-based search-based software engineering: A critical review and methodological guidance, 11 2020. ISSN 0098-5589, 1939-3520, 2326-3881. URL https://doi.org/10.1109/tse.2020.3036108.

Qiang Long, Changzhi Wu, Tingwen Huang, and Xiangyu Wang. A genetic algorithm for unconstrained multi-objective optimization. *Swarm and Evolutionary Computation*, 22:1–14, 01 2015a. ISSN 2210-6502, 2210-6510. doi: 10.1016/j.swevo.2015.01.002. URL https://doi.org/10.1016/j.swevo.2015.01.002.

Qiang Long, Changzhi Wu, Xiangyu Wang, Lin Jiang, and Jueyou Li. A multiobjective genetic algorithm based on a discrete selection procedure. *Mathematical Problems in Engineering*, 2015:1–17, 01 2015b. ISSN 1024-123X, 1026-7077, 1563-5147. doi: 10.1155/2015/349781. URL https://doi.org/10.1155/2015/349781.

Ahmed Maghawry, Rania Hodhod, Yasser Omar, and Mohamed Kholief. An approach for optimizing multi-objective problems using hybrid genetic algorithms. *Soft Computing*, 25:389–405, 07 2020. ISSN 1432-7643, 1433-7479. doi: 10.1007/s00500-020-05149-3. URL https://doi.org/10.1007/s00500-020-05149-3.

Dhagash Mehta and Crina Groşan. A collection of challenging optimization problems in science, engineering and economics. *2022 IEEE Congress on Evolutionary Computation (CEC)*, 25:2697–2704, 05 2015. doi: 10.1109/cec.2015.7257223. URL https://doi.org/10.1109/cec.2015.7257223.

Elliot Meyerson and Risto Miikkulainen. Discovering evolutionary stepping stones through behavior domination. *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 139–146, 06 2017. doi: 10.1145/3071178.3071315. URL https://doi.org/10.1145/3071178.3071315.

Shuhairie Mohammad, Ahmad Nor Kasruddin Nasir, Nor Maniha Abdul Ghani, Raja Kamil, Ahmad Azwan Abd Razak, Mohd Falfazli Mat Jusof, and Nurul Amira Mhd Rizal. Hybrid bacterial foraging sine cosine algorithm for solving global optimization problems. *IOP Conference Series Materials Science and Engineering*, 917:12081–12081, 09 2020. ISSN 1757-8981, 1757-899X. doi: 10.1088/1757-899x/917/1/012081. URL https://doi.org/10.1088/1757-899x/917/1/012081.

Adel Nikfarjam, Amirhossein Moosavi, Aneta Neumann, and Frank Neumann. *Computing High-Quality Solutions for the Patient Admission Scheduling Problem Using Evolutionary Diversity Optimisation*, pages 250–264. Springer Science+Business Media, 01 2022. doi: 10.1007/978-3-031-14714-2_18. URL https://doi.org/10.1007/978-3-031-14714-2_18.

Effanga Effanga Okon. On technique for generating pareto optimal solutions of multi-objective linear programming problems. *Science Journal of Applied Mathematics and Statistics*, 7:15–15, 01 2019. ISSN 2376-9491, 2376-9513. doi: 10.11648/j.sjams.20190702.12. URL https://doi.org/10.11648/j.sjams.20190702.12.

Alberto Pajares, Xavier Blasco, J. M. Herrero, and Gilberto Reynoso-Meza. A multiobjective genetic algorithm for the localization of optimal and nearly optimal solutions which are potentially useful: nevmoga. *Complexity*, 2018, 01 2018. ISSN 1076-2787, 1099-0526. doi: 10.1155/2018/1792420. URL https://doi.org/10.1155/2018/1792420.

Erickson Puchta, Priscilla Bassetto, Lucas Henrique Biuk, Marco A. Itaborahy Filho, Attílio Converti, M. Kaster, and Hugo Valadares Siqueira. Swarm-inspired algorithms to optimize a nonlinear gaussian adaptive pid controller. *Energies*, 14:3385–3385, 06 2021. ISSN 1996-1073. doi: 10.3390/en14123385. URL https://doi.org/10.3390/en14123385.

Alexander Thebelt, Calvin Tsay, Robert M. Lee, Nathan Sudermann-Merx, D. Walz, Thomas G. Tranter, and Ruth Misener. Multi-objective constrained optimization for energy applications via tree ensembles. *Applied Energy*, 306:118061–118061, 10 2021. ISSN 0306-2619, 1872-9118. doi: 10.1016/j.apenergy.2021.118061. URL https://doi.org/10.1016/j.apenergy.2021.118061.

Shuxin Wang, Yingcai Zheng, Li Cao, and Maoren Xiong. Comprehensive adaptive enterprise optimization algorithm and its engineering applications. *Biomimetics*, 10:302–302, 05 2025. ISSN 2313-7673. doi: 10.3390/biomimetics10050302. URL https://doi.org/10.3390/biomimetics10050302.

Xin Xia, Jie Ji, Chaoshun Li, Xiaoming Xue, Xiaolu Wang, and Chu Zhang. Multiobjective optimal control for hydraulic turbine governing system based on an improved mogwo algorithm. *Complexity*, 2019, 01 2019. ISSN 1076-2787, 1099-0526. doi: 10.1155/2019/3745924. URL https://doi.org/10.1155/2019/3745924.

Huarong Xu, Qianwei Deng, Zhiyu Zhang, and Sijie Lin. A hybrid differential evolution particle swarm optimization algorithm based on dynamic strategies. *Scientific Reports*, 15, 02 2025. ISSN 2045-2322. doi: 10.1038/s41598-024-82648-5. URL https://doi.org/10.1038/s41598-024-82648-5.

Jiajun Yu, Yizhen Zheng, Huan Yee Koh, Shirui Pan, Tianyue Wang, and Haishuai Wang. Collaborative expert llms guided multi-objective molecular optimization. 01 2025. doi: 10.48550/ARXIV.2503.03503. URL https://arxiv.org/abs/2503.03503.

Guangyi Zhang and Aristides Gionis. Maximizing diversity over clustered data. *arXiv (Cornell University)*, 01 2020. doi: 10.48550/arxiv.2001.03050. URL https://arxiv.org/abs/2001.03050.

Shuhan Zhang, Shengsheng Wang, Ruyi Dong, Kai Zhang, and Xiaohui Zhang. A multi-strategy improved outpost and differential evolution mutation marine predators algorithm for global optimization. *Arabian Journal for Science and Engineering*, 48:10493–10516, 02 2023. ISSN 2191-4281, 2193-567X. doi: 10.1007/s13369-023-07683-2. URL https://doi.org/10.1007/s13369-023-07683-2.