A Unified *Innovized* Progress Operator for Performance Enhancement in Evolutionary Multi- and Many-objective Optimization

Sukrit Mittal, Dhish Kumar Saxena, Kalyanmoy Deb,and Erik D. Goodman

COIN Report Number 2022006

***Abstract*—Recent studies have shown that for reference-vector-(RV-) based evolutionary multi- and many-objective optimization algorithms, denoted as RV-EMâOAs, machine learning models can be trained to capture efficient *search* directions, through mapping of *inter-generational* solutions *along* the various RVs. Such search directions could then be utilized to create pro-convergence offspring, facilitating convergence enhancement. To broaden the above scope, this paper first proposes an exclusive diversity-based *innovized progress* (IP3) operator. It relies on producing pro-diversity offspring, by learning efficient search directions, through a mapping of *intra-generational* solutions *across* the RVs. This operator is shown to significantly enhance the performance of RV-EMâOAs on some of the well-known diversity-hard problems. Finally, a *unified innovized progress* (UIP) operator has been proposed for simultaneous enhancement of convergence and diversity, by adaptively balancing the creation of pro-convergence and pro-diversity offspring. In principle, these operators are *generic* (avoiding *ad hoc* decisions on critical aspects) and *practicable* (avoiding any extra solution evaluations). Based on $24,056$ experimental runs on multi- and many-objective problems, the UIP operator, when integrated with RV-EMâOAs such as NSGA-III, θ-DEA, MOEA/DD, and LHFiD, has provided statistically better performance in about 32% of instances, and equivalent or better in about 92% of instances, compared to the respective base RV-EMâOAs.**

Index Terms*—Multiobjective Optimization, Learning-assisted Optimization, *Innovized Progress*, Machine Learning, *Innovization*, Online *Innovization

I. Introduction

OPTIMIZATION is an iterative process of arriving at one or more optimal solution(s), depending on the number of objectives (M) involved. In multi-objective ($M = 2$ and 3) or many-objective ($M \geq 4$) optimization, the aim is to find a set of well-distributed Pareto optimal (PO) solutions, generally referred to, in the objective (F)-space, as the Pareto Front (PF). Towards this, multi- and many-objective evolutionary algorithms are often used [1, 2], collectively referred to as EMâOAs in this paper. Given a problem, EMâOAs seek to evolve a finite set of initially random solutions over several generations, using *natural variation* and *selection*, towards a good PF-approximation, in terms of *convergence* and *diversity* [1]. Notably, a well-diversified set of solutions is not practically relevant if the solutions have not converged to the PF. Similarly, well-converged solutions that occupy only a small part of the PF are undesirable. Hence, maintaining the convergence-diversity balance is critical for any EMâOA [3].

One class of EMâOAs relies on the use of dominance principles to pursue convergence, within a reference vector (RV)-based framework that helps to pursue diversity [4]. Such EMâOAs are here called RV-EMâOAs. In the context of RV-EMâOAs, some recent studies have proposed additional machine learning (ML)-based operators for *convergence enhancement*, such as ANN-based basic *innovized progress (IP)* [5, 6, 7] and random forest-based more advanced *innovized progress 2 (IP2)* [8] operators. These operators rely on learning the efficient search directions in variable (X)-space, based on a mapping of *inter-generational* solutions in F-space, *along* the RVs; and utilizing that learning in the same generation, to produce pro-convergence *offspring*. While such offspring are shown to facilitate enhancement of convergence properties of RV-EMâOAs, the hallmark of these operators is that they do not necessitate any extra solution evaluations (SEs). However, the efficacy of these operators critically depends on: (a) the proportion ($\mathcal{P}$) of pro-convergence offspring produced by these operators vis-à-vis offspring produced by natural variation operators, and (b) the number of generations between two successive invocations of these operators (t_{freq}). The above studies have used $\mathcal{P} = 50\%$ and $t_{\text{freq}} = 5$, in consideration of the following:

- *Convergence-diversity balance:* notably, the offspring produced by the natural variation operators, relying on *guided randomness*, are convergence-diversity-neutral. Hence, it is important that in any generation where these operators are invoked, they do not over-skew the offspring in favour of convergence, at the cost of diversity.
- *Risk-rewards tradeoff:* notably, the above operators rely on *learning* efficient search directions (in X-space), based on mapping of inter-generational solutions in F-space. Clearly, this *learning* may not be meaningful for several reasons, such as *non-linearity* in the given problem, or the *choice* of ML method. Hence, to moderate the degree of reliance on these operators, they are invoked only intermittently, as opposed to every generation.

This work was supported by the Ministry of Human Resource Development, Government of India through the SPARC Scheme under Project P66.

Sukrit Mittal and Dhish Kumar Saxena are with the Department of Mechanical and Industrial Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India (e-mail: smittal1@me.iitr.ac.in; dhish.saxena@me.iitr.ac.in).

Kalyanmoy Deb and Erik D. Goodman are with the BEACON Center for the Study of Evolution in Action and the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: kdeb@egr.msu.edu; goodman@egr.msu.edu).

Taking what is learned from these operators forward, this paper aims to broaden their scope by simultaneously catering to the dual goals of RV-EMâOAs, namely, *convergence* and *diversity*, while infusing *generality* (avoiding *ad hoc* decisions on critical aspects) and retaining *practicability* (avoiding any extra SEs). In this context, this paper proposes:

- IP2$^+$ (Innovized Progress 2$^+$) operator: a variant of the existing IP2 operator [8], wherein the number of hard-fixed tunable parameters, including t_{freq}, is reduced, without compromising its performance.
- IP3 (Innovized Progress 3) operator: for diversity enhancement. It relies on learning efficient search directions (in X-space), based on mapping of *intra-generational* solutions (in F-space), *across* the RVs; and utilizing that learning in the same generation, to create pro-diversity offspring (in X-space).
- UIP (Unified Innovized Progress) operator: for convergence and diversity enhancement, which relies on invoking either or both of IP2$^+$ and IP3 in the same generation, to produce both pro-convergence and pro-diversity offspring.

To establish the search efficacy infused by the IP2$^+$ and IP3 operators into an RV-EMâOA, on multi-objective instances:

- the IP2$^+$ operator has been compared with the existing IP2 operator, through a comparison of the IP2$^+$ and IP2 variants of the same RV-EMâOAs.
- the IP3 variant of each RV-EMâOA has been compared with the base RV-EMâOA (without ML-based operator).

A direct comparison of IP2$^+$ and IP3 operators has been avoided above, since both cater to mutually exclusive goals in an RV-EMâOA. Further, to establish the search efficacy infused by the UIP operator into an RV-EMâOA:

- the UIP variant of each RV-EMâOA has been compared with its respective IP2$^+$ and IP3 variants, on multi-objective instances, to establish the UIP's superiority.
- the UIP variant of each RV-EMâOA has been compared with the base RV-EMâOA, on many-objective instances, to investigate its scalability with respect to M.

The remainder of this paper is structured as summarized below. An overview of existing ML-based enhancements in RV-EMâOAs is provided, including a state-of-the-art IP2 operator, in Section II. Sections III, IV and V discuss the proposed IP2$^+$, IP3 and UIP operators, respectively, and their operation with the various RV-EMâOAs. Section VI presents the experimental settings used, followed by the results on a wide range of multi- and many-objective test problems in Section VII. Finally, the paper concludes with Section VIII.

II. Existing State of the Art

In EMâOAs, the natural *variation* operators, namely, crossover and mutation, are not found effective in finding the optimal solutions for certain problems [9, 10]. To effect a better search, several enhancements to these EMâOAs have been proposed [11, 12]. These include several ML-based enhancements, such as: (i) surrogate modeling, (ii) model-based offspring creation, and (iii) online innovization.

In surrogate modeling, the overall idea is to learn the variable-objective relationships locally, using an ML method (called a surrogate model), and evolve the solutions based on approximate function and constraint values. This requires fewer *actual* SEs to converge [13, 14]. Besides facilitating the *approximated* SEs, surrogate models have also been used to perform other tasks in EMâOAs, including local search [15, 16] and efficient offspring-creation [17, 18].

Model-based offspring creation started with Estimation of Distribution Algorithms (EDAs) [19]. EDAs, such as BOA [20], EDA-VNS [21] and HMOBEDA [22], have shown a potentially distinctive advantage in exploiting the inter-variable dependencies in creating new offspring solutions. They extract the global search space statistical information from the current search and build a *probabilistic model* of *elite* solutions, using ML methods such as Bayesian networks or decision trees. This model is then used to sample new offspring. However, their advantage is tangible only where inter-variable dependencies exist in the given problem. Towards this, several algorithms that rely partially on natural variation operators and partially on model-based sampling have been proposed, including IM-MOEA [23] and GMOEA [24].

Some other studies have focused on an *online innovization* approach [25, 26, 27], that relies on extracting inter-variable relationships from current non-dominated solutions, and utilizing these in subsequent generations to *repair* the offspring created using variation operators. However, the structure of these relationships has to be specified a priori. Doing away with this need, some ML-based operators for convergence enhancement have been proposed that exploit the RV-based architecture of RV-EMâOAs. These include: IP [7] and IP2 [8] operators. Since IP2 is an improved version of the IP operator [8], its constitutive modules are worth noting. These include:

- *Training-dataset construction:* first the solutions in F-space, from t_{past} earlier generations are associated with the RVs, and then mapped to the best solutions found so far along the respective RVs. The X-vectors of these mapped pairs of solutions constitute the training-dataset.
- *ML Training:* the training-dataset is first normalized, and then a random forest (RF) model is trained.
- *Offspring's progression:* the trained ML model is used to *advance* a proportion $\mathcal{P}^{\text{IP2}}$ of the total offspring (N) in the current generation. These advanced offspring ($\lfloor \mathcal{P}^{\text{IP2}} N \rfloor$ in number) are further improved by extrapolation using a parameter η, to yield the jutted offspring.

Moreover, in an RV-EMâO-IP2 run (RV-EMâOA integrated with IP2), the IP2 operator is invoked only after every t_{freq} generations. Overall, the IP2 operator utilizes four parameters, namely, $t_{\text{past}} = 5$, $\mathcal{P}^{\text{IP2}} = 50\%$, $\eta = 1.1$ and $t_{\text{freq}} = 5$.

III. IP2$^+$ Operator for Convergence Enhancement

The proposed IP2$^+$ operator vis-à-vis the IP2 operator, or equivalently RV-EMâO-IP2$^+$ vis-à-vis RV-EMâO-IP2, is marked by three key variations:

- First invocation: unlike the case of IP2 which gets invoked for the first time as soon as t_{past} generations pass,

IP2$^+$ is invoked for the first time only when the entire population becomes non-dominated.
- Frequency of invocation governed by t_{freq}: unlike the case of IP2 where $t_{\text{freq}} = 5$ was hard fixed, IP2$^+$ relies on an adaptive $t_{\text{freq}} \geq 1$.
- Extent of offspring progression governed by parameter η: unlike the case of IP2 where $\eta = 1.1$ was hard fixed, IP2$^+$ infuses variability by relying on $\eta \in [1, 1.5]$.

The above changes, pertaining to the first invocation (t_{past}), t_{freq} and η, are motivated by the need to avoid hard fixing of tunable parameters. Further, basing the first-invocation of IP2$^+$ on the entire population's non-domination, implicitly ensures a minimum level of population-diversity. This is desirable in the context of the subsequently proposed UIP operator, that focuses on simultaneous convergence and diversity enhancement. In that, achieving a minimum level of diversity before focusing on convergence enhancement could be crucial.

IV. IP3 Operator for Diversity Enhancement

Similar to the IP2 operator, the IP3 operator consists of three modules: *Training-dataset construction*; *ML training*; and *Offspring creation*. The design and implementation of these modules are detailed in the following subsections.

A. Training-dataset Construction

This section focuses on identification of solution-mapping requirements towards diversity enhancement; training-dataset constitution and its algorithmic implementation.

1) Deciphering the solution-mapping requirements towards diversity enhancement: It must be acknowledged up front that the training-dataset is to be designed in a manner that the consequent ML model can cater to both aspects of diversity - solutions' *spread* and *uniform distribution* within the spread.

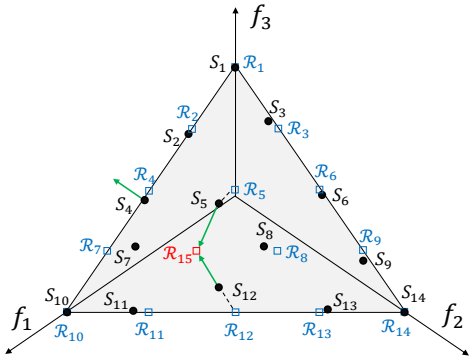


Fig. 1: Depicting the need for objective-wise mapping, towards training-dataset construction.

Figure 1 reveals that no solution is associated with $\mathcal{R}_{15}$, adversely impacting the uniformity. This could be resolved if a trained ML model could help advance one of the neighboring solutions (S_{12}, S_5, etc., in increasing order of distance) onto $\mathcal{R}_{15}$. A natural choice is to advance S_{12} onto $\mathcal{R}_{15}$, characterized by an improvement in f_1 and f_2 and a deterioration in f_3. Alternatively, if say S_{12} was not existent, the S_5 could be advanced onto $\mathcal{R}_{15}$, characterized by an improvement in f_3, and a deterioration in f_1 and f_2. The different characterizations above (for S_{12} and S_5) suggest that the *mapping requirements for diversity enhancement cannot be generalized based on simultaneous variation in all the objectives*.

The above challenge could be resolved by recognizing that that an M-dimensional space could be most efficiently spanned through M linearly independent *basis* vectors. In the current context, it promises that if M independent ML models could be trained - *each capable of advancing any solution towards an improvement in a distinct objective*, then both aspects of diversity could be addressed, as highlighted below:

- uniform distribution: towards it, first, the nearest neighboring solution to a vacant RV can be identified (S_{12} for $\mathcal{R}_{15}$). Then for such a solution, the objective undergoing maximum transition during its advancement can be determined, say f_m. If this transition pertains to an improvement in f_m ($m = 2$ for S_{12}), the m^{th} ML model can be used. However, if it pertains to a deterioration in f_m ($m = 3$ for S_{12}), the m^{th} model can still be used, but the direction of advancement needs to be reversed.
- spread expansion: towards it, first all the boundary RVs (with at least one associated solution) can be identified, as those having one of the weight components as zero. Given this: (i) a solution S_i will be a boundary solution, if for the underlying $\mathcal{R}_i$, $w_{ik} = 0$ for some $k \in [1 : M]$, and (ii) S_i can facilitate spread-expansion through its advancement in the X-space, such that in the F-space it marks an improvement in f_k - enabled by the k^{th} ML model. For example in Figure 1, S_4 is a boundary solution ($w_{42} = 0$) whose improvement in f_2 (enabled by the second ML model) will lead to spread-expansion.

2) Proposed Training-dataset: The above section has highlighted the potential utility of M ML models, each capable of facilitating improvement in a particular objective. This points to the need for M training-datasets. Before detailing their how such datasets could be constituted, the numerical and conceptual prerequisites are presented below:

1) Projection of F-space onto the unit simplex: this recommendation, for projecting the solutions (parent population) onto the unit simplex, has been made:
   - to ensure that the difference in the convergence levels of different solutions does not impact the endeavor for diversity improvement, in any manner.
   - to ensure a viable and meaningful assessment of the ML model to be for solution advancement. Consider that S_j is the nearest solution, to be advanced to a vacant RV, $\mathcal{R}_i$. Determination of the specific objective which undergoes maximum transition (improvement/deterioration) in such an advancement entails computation of $\Delta f_k = f_{jk} - w_{ik}\ \forall k \in \{1 \dots M\}$, where, f_{jk} denote the elements of the objective-vector of S_j, and w_{ik} denote the weight components of $\mathcal{R}_i$. For Δf_k computations to be meaningful, f_{jk} and and w_{ik} ought to be on the same scale. This necessitates projection of the solutions onto the same unit simplex, on which the RVs are sampled. Towards

it, first the normalized objective values, given by $\bar{F}(X_i) \equiv \{\bar{f}_1(X_i), \ldots, \bar{f}_M(X_i)\}$ can be determined, enabling computation of projected values, as $\hat{F}(X_i) \equiv \{\hat{f}_1(X_i), \ldots, \hat{f}_M(X_i)\}$, where:

$$\hat{f}_m(X_i) = \frac{\bar{f}_m(X_i)}{\Sigma_{j=1}^{M} \bar{f}_j(X_i)} \;\forall\; m \in \{1, 2, \ldots, M\} \quad (1)$$

Finally, if $|\Delta f_k|$ is maximum for $k = m$, then the m^{th} ML model can be used, as is (if $\Delta f_m > 0$), or with reversed direction of advancement (if $\Delta f_m < 0$).

2) Notion of neighbourhood of a solution: given a solution S_i associated with $\mathcal{R}_i$, its neighborhood is defined here, as $Nbd(S_i) \equiv \{S_j |\; 0.5r < dist(\mathcal{R}_i, S_j) < 1.5r\}$, where r - representing the average spacing between the RVs can be given by Equation 2.

$$r = \frac{\Sigma_{i=1}^{N} \{\min_{j=1}^{N} \text{dist}(\mathcal{R}_i, \mathcal{R}_j)\}}{N} \quad (2)$$

The rationale for this definition of $Nbd(S_i)$ is symbolically depicted in Figure 2. In that: (i) $\mathcal{R}_i$ and $\mathcal{R}_j$, separated by r, are two adjacent RVs, (ii) considering that solutions are associated with the closest RV: S_i is associated with $\mathcal{R}_i$, while S_{j1} and S_{j2} are associated with $\mathcal{R}_j$, and (iii) $Nbd(S_i) \equiv \{S_{j1}, S_{j2}\}$.

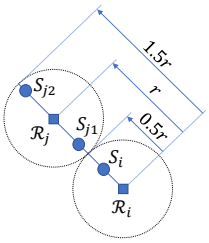


Fig. 2: Depicting the neighbourhood of a solution (S_i).

Proposed Training-dataset construction: With an aim to facilitate construction of M ML models (one-per-objective):

- it is natural that M training-datasets be constructed, one-per-objective, say, $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$.
- for $\mathcal{D}_k$, which is to serve as the training-dataset for the k^{th} ML model (ML_k): it is proposed that for every solution $S_i,\ i = 1 \ldots N$, its underlying X-vector be mapped onto the X-vector of another solution $S_j \in Nbd(S_i)$, such that S_j offers the *maximum improvement* in $\hat{f}_k$, compared to S_i. In effect, $\mathcal{D}_k$ captures the pertinent X-space transitions which facilitate maximum improvement in $\hat{f}_k$, across all the solutions, that would be learned by ML_k. Repeated for each objective, this process would lead to M ML models, capable of advancing any given solution towards improvement in a particular objective.

Notably, $\mathcal{D}_1$–$\mathcal{D}_M$ can constitute a matrix $\mathcal{S}$, ideally sized $N \times M$, such that its element $\mathcal{S}_{im} = \{X(S_i), X(S_j) - X(S_i)\}\ \forall\ i \in [1, N],\ m \in [1, M]$, where S_i is an input solution in the F-space; $S_j \in Nbd(S_i)$ is the target solution that offers *maximum improvement* in $\hat{f}_m$; and $X(S_i)$ and $X(S_j)$ represent the X-vectors of S_i and S_j, respectively. Notably, the $Nbd(S_i)$ may be such that solutions offering improvement in each objective may not exist. Hence, the matrix $\mathcal{S}$ may not be fully populated.

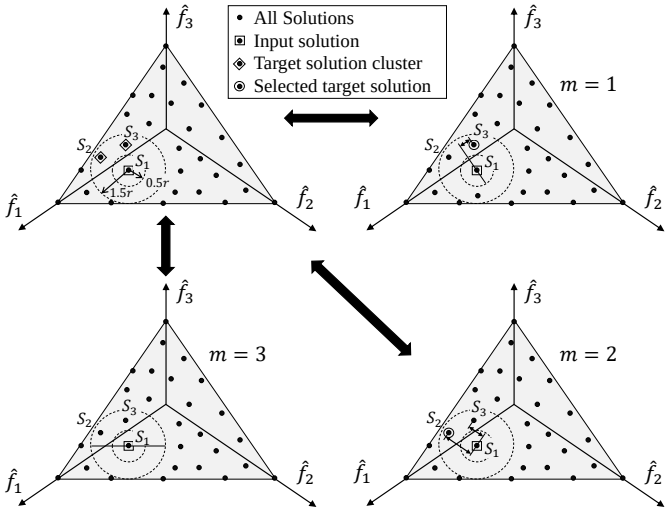


Fig. 3: A schematic representation of the training-dataset construction based on *objective-wise* mapping, within a predefined *neighbourhood*, in the *projected* F-space.

For visual depiction of the proposed training-dataset construction, a three-objective scenario has been presented in Figure 3. There, hypothetical solutions projected on the unit simplex are shown. One of the solutions, marked as S_1, is treated as the *input* solution; $Nbd(S_1)$ has been marked by two dotted circles; and the solutions S_2 and S_3 belonging to $Nbd(S_1)$ are said to constitute the *target solutions* cluster $\mathcal{C}$. For each objective $m \in \{1, 2, 3\}$, the selection of *target* solutions from $\mathcal{C}$ is discussed below.

1) $m = 1$: only S_3 offers a better value in $\hat{f}_1$ than S_1, and hence is selected as the *target* solution.
2) $m = 2$: both S_2 and S_3 offer better values in $\hat{f}_2$ than S_1. Among these, S_2 is selected as the *target* solution since it offers larger improvement in $\hat{f}_2$.
3) $m = 3$: there is no solution offering a better value in $\hat{f}_3$ than S_1, implying no *target* solution. Given this, S_1 does not contribute to $\mathcal{D}_3$.

3) Algorithmic implementation: In the above context, the procedure of constructing $\mathcal{D}_1$–$\mathcal{D}_M$ is summarized in Algorithm 1 (AL.1). There, the first step is the computation of projected objective values $\hat{F}$, using Equation 1. Subsequently, each solution $S_i \in P_t$ is considered as a potential *input* solution (line 3, AL.1), and subjected to the following steps.

1) *Identification of target solution cluster $\mathcal{C}$ (lines 4–7, AL.1)*: the *target* cluster $\mathcal{C}$, corresponding to an *input* solution S_i, constitutes all such solutions $S_j \in P_t$ that belong to the neighbourhood of S_i, implying $S_j \in Nbd(S_i)$.
2) *Identification of the target solutions from $\mathcal{C}$, towards each of $\mathcal{D}_1$–$\mathcal{D}_M$ (lines 8–11, AL.1):* for each objective $m \in [1, M]$, the solution $S_j \in \mathcal{C}$ offering the minimum value of $\hat{f}_m(S_j)$, is identified. If the latter is better than $\hat{f}_m(S_i)$, then the underlying X vectors are included in the corresponding training-dataset $\mathcal{D}_m$ (lines 11–12, AL.1).

Algorithm 1: `Dataset_Construction`(P_t, $\mathcal{R}$, r)

Input: Parent population P_t, RVs $\mathcal{R}$, neighbourhood radius r
Output: M training-datasets $\mathcal{D}_1$–$\mathcal{D}_M$

Compute $\hat{F}$ values using Equation 1
$\{\mathcal{D}_1, \mathcal{D}_1, \ldots, \mathcal{D}_M\} \leftarrow \{\emptyset, \emptyset, \ldots, \emptyset\}$
for $S_i \in P_t$ **do**
  $\mathcal{C} \leftarrow \emptyset$
  for $S_j \in P_t \ \forall\ j \neq i$ **do**
    if $S_j \in Nbd(S_i)$ **then**
      $\mathcal{C} \leftarrow \mathcal{C} \cup S_j$
  for $m = 1$ *to* M **do**
    $S_j \leftarrow \text{argmin}_{S_j \in \mathcal{C}} \hat{f}_m(S_j)$
    if $\hat{f}_m(S_j) < \hat{f}_m(S_i)$ **then**
      $\mathcal{D}_m \leftarrow \mathcal{D}_m \cup [X(S_i), (X(S_j) - X(S_i))]$

Algorithm 2: `ML_Training` ($\mathcal{D}, [x^l, x^u]$)

Input: Combined training dataset $\mathcal{D}$, lower & upper bounds of variables specified in the problem, x^l and x^u
Output: M trained ML models ML_1–ML_M

$\{x^{l,t}, x^{u,t}\} \leftarrow$ Minimum and Maximum of each variable in $\mathcal{D}$ at generation t
$x^{\min}, x^{\max} \leftarrow \emptyset, \emptyset$
for $k = 1$ *to* n_{var} **do**
  $x_k^{\min} = 0.5 \times (x_k^{l,t} + x_k^l)$
  $x_k^{\max} = 0.5 \times (x_k^{u,t} + x_k^u)$
Normalize $\mathcal{D}$ using $\mathbf{x}^{\min}$ and $\mathbf{x}^{\max}$ as bounds
for $m = 1$ *to* M **do**
  Train ML_m using $\mathcal{D}_m$

B. ML Training

The goal here is to train M ML models (one-per-dataset), such that for each objective m, an aggregated search direction in X-space that promises improvement in $\hat{f}_m$, could be *learned* on the basis of the corresponding N promising search directions embedded in $\mathcal{D}_m$. To that end, k-Nearest Neighbours (kNN) regression[1] has been used as the ML method. When a test instance is provided for prediction, the kNN identifies its k nearest inputs in the original training-dataset, and returns the average of their respective targets as the prediction [28]. In the context of $\mathcal{D}_m$, given a solution's X-vector as test input: (a) the kNN model (trained on $\mathcal{D}_m$) identifies its k nearest *input* solutions in $\mathcal{D}_m$; and (b) returns the average of their respective difference vectors, which could be treated as a potential search direction in X-space. Notably,

- during the training, the *input* solutions in $\mathcal{D}_m$ are strategically stored using a k-d tree method [29], so that while making the prediction for a test input, the corresponding k nearest neighbours can be identified efficiently.
- it is known that a very low or high value of k (in kNN) leads to overfitting or underfitting, respectively. An appropriate choice for k depends on the dataset and the application it represents. In IP3's context, $k = n_{\text{var}}$ (number of variables) has been used, based on the rationale that: (i) the training-dataset, say $\mathcal{D}_m$, involves $n_{\text{var}} \times 1$ dimensional vectors, say, X_I and X_O for the input and output, respectively, and (ii) given an $n_{\text{var}} \times 1$ dimensional test-vector (say, X_T), it is appropriate to account for as many nearest-neighbors (k) of X_T among X_I, as there are variables, so that even if each neighboring X_I accounts for variation in only one of the distinct elements in an $n_{\text{var}} \times 1$ dimensional X_T, the neighboring X_Is collectively account for sufficient variation on the input side, so that the average of the corresponding X_Os does not amount to overfitting. Moreover, a sensitivity analysis for k has been presented in Section S6 of the supplementary document (S.D.).

Similarly to IP2, as presented in Algorithm 2 (AL.2), the ML training module is executed as a two-step process: (a) training-dataset normalization using the *dynamic normalization* method [7]; and (b) the ML training itself. Since the kNN training relies on identifying the k nearest neighbours, dataset normalization (in X-space) before training is crucial.

[1]The implementation of the kNN method has been taken from Scikit-learn package (in python language).

C. Offspring Creation

In any generation where IP3 is invoked, this module entails the advancement of a proportion $\mathcal{P}^{\text{IP3}}$ of the parent population P_t (sized N), leading to $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring. In that process, half of the offspring, Q_t^B (sized $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$), are created by subjecting suitable P_t members to *boundary progression* for better spread; and the other half, Q_t^G (sized $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$), are created by subjecting suitable P_t members to *gap progression* for better uniformity, as detailed below.

1) Boundary Progression: the procedure for creating the offspring (Q_t^B) towards better spread, using *boundary progression*, is summarized in Algorithm 3 (AL.3). As per the rationale set up in Section IV-A1, it entails identification of the boundary solutions in P_t, and their advancement using an appropriate ML model to create new offspring.

The boundary solutions can be characterized as those associated with RVs with at least one '0' component (some $w_i = 0$). For algorithmic implementation, let such RVs having at least one member of P_t associated with themselves be denoted by $\mathcal{R}^B$ (line 1, AL.3). Subsequently, the creation of each offspring in Q_t^B, involves the following steps.

Algorithm 3: `Bound_Prog` (P_t, $\mathcal{R}$, ML, [$x^{\min}$, $x^{\max}$], [x^l, x^u], $\mathcal{P}^{\text{IP3}}$)

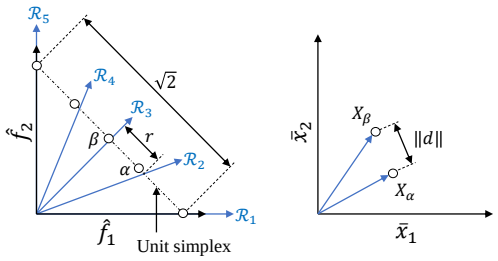
Input: Current population P_t, RVs $\mathcal{R}$, ML models ML_1–ML_m, bounds from Algorithm 2 [$x^{\min}$, $x^{\max}$], variable bounds in problem definition [x^l, x^u], proportion $\mathcal{P}^{\text{IP3}}$
Output: New solutions created Q_t^B

$\mathcal{R}^B \leftarrow$ All associated RVs in $\mathcal{R}$ with at least one '0' component
$Q_t^B \leftarrow \emptyset$ % sized $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor \times n_{\text{var}}$
for $i = 1$ *to* $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ **do**
  $\vec{B} \leftarrow$ A randomly selected RV from $\mathcal{R}^B$
  $S_{\text{start}} \leftarrow$ Nearest solution associated with $\vec{B}$
  $m \leftarrow$ A randomly selected objective such that $\vec{B}_m = 0$
  $\bar{X}(S_{\text{start}}) \leftarrow$ Normalized $X(S_{\text{start}})$ using $x^{\min}$ and $x^{\max}$
  $\bar{d}_X \leftarrow ML_m(\bar{X}(S_{\text{start}}))$
  $d_X \leftarrow$ Denormalized $\bar{d}_X$ using $x^{\min}$ and $x^{\max}$
  $X(S_{\text{new}}) \leftarrow X(S_{\text{start}}) + \lambda_B \times \hat{d}_X$ % $\hat{d}_X = d_X / \|d_X\|$
  Boundary repair on $X(S_{\text{new}})$
  $Q_t^B \leftarrow Q_t^B \cup S_{\text{new}}$

1) *Identification of the solution to be advanced (lines 4–5, AL.3):* an RV $\vec{B}$ is randomly selected from $\mathcal{R}^B$. Among the solutions associated with $\vec{B}$, the one closest to it by perpendicular distance, say S_{start}, is chosen for advancement.
2) *Selection of the ML model (line 6, AL.3):* the solution S_{start}, identified above, can be characterized by the fact that it offers one of the best values in at least one particular objective. For instance, if $\vec{B}_m = 0$, it implies that S_{start} offers one of the best values in $\hat{f}_m$, and that further improvement in $\hat{f}_m$ would imply boundary expansion (beyond the current projected F-space). In case more than one component of $\vec{B}$ is 0, one of the objectives (m, such that $\vec{B}_m = 0$) is randomly selected. Consequently, ML_m becomes the desired ML model.
3) *Obtaining the search direction (lines 7–9, AL.3):* given the X-vector of a solution, say S_{start}, the trained ML_m is capable of providing a potential search direction d_X in X-space. Considering that ML_m was trained on the normalized dataset: (i) $X(S_{\text{start}})$ needs to be normalized using the bounds computed in Algorithm 2, before applying ML_m; and (ii) the normalized search direction obtained by applying ML_m, denoted by $\bar{d_X}$, needs to be denormalized using the same bounds, leading to d_X.
4) *Advancement and repair (lines 10–11, AL.3):* the advancement of S_{start} leading to S_{new}, can be given by $X(S_{\text{new}}) = X(S_{\text{start}}) + \lambda_B \times \hat{d}_X$, where $\hat{d}_X$ is a unit vector along the search direction d_X (computed above), and λ_B is the step length. Towards the determination of λ_B, the desired scaling in F-space for boundary progression needs to be imposed on the relationship of scales in F-space and X-space that is inherent in the ML training-dataset, as discussed below:
   a) the scaling requirements in the F-space: consider doubling-up the current spread in F-space. The required scaling can be given by the product of: (i) the average distance (r) between any two adjacent RVs on the unit simplex, and (ii) the number of adjacent-RV transitions required to span the boundary of the unit simplex, given by $\lfloor\sqrt{2}/r\rceil$. The justification for the later is that spanning the boundary on the unit simplex ($\mathcal{R}_1$ to $\mathcal{R}_5$, Figure 4a) corresponds to $\sqrt{2}$.
   b) the inherent relationship of scales in F- and X-space in the ML training-dataset: Figure 4 suggests that two solutions α and β, associated with adjacent RVs, interspaced by r in the F-space, are separated in the X-space by $\|d\|$, where $d \equiv X_\beta - X_\alpha$. Hence, the desired scaling factor of $\lfloor\sqrt{2}/r\rceil \times r$ in the F-space calls for a step length in the X-space, given by $\lambda_B = \lfloor\sqrt{2}/r\rceil \times \|d_A\|$, where $\|d_A\|$ is the average of $\|d\|$ over the solutions associated with all the pairwise adjacent RVs.

   While the founding principles of the scaling factor in F-space, and step length in X-space are presented above, they need to be adapted, so they can cater to the varying requirements along the RV-EMâOA generations. That is:
   - during the early generations of an RV-EMâOA, the current spread in F-space may be a minor fraction of the ideal PF, necessitating a large scaling factor,
   - during the later generations, where the spread in F-space may be comparable to the ideal PF spread, a small scaling factor may suffice owing to the limited scope for spread expansion.

   Given the above, it is important to: (i) impose an upper and lower bound on the scaling factor, and (ii) infuse randomness while assigning an intermediate value to it (since the scope for spread expansion cannot be determined a priori). Towards it, this paper proposes that the upper bound be based on doubling the current spread, while the lower bound and intermediate values can be obtained through a randomized scalar. Considering the above modifications to the scaling factor (in F-space), the corresponding step length (in X-space) can be given by $\lambda_B = rand(0,1) \times \lfloor\sqrt{2}/r\rceil \times \|d_A\|$. Notably, the obtained $X(S_{\text{new}})$ may have some out-of-bound variables. These variables are repaired using the method proposed in [30], which is the same as used in the IP2 operator [8].



(a) Projected F-space (b) Normalized X-space

Fig. 4: A symbolic depiction of distance between two solutions associated with adjacent RVs.

While the above steps are repeated $\lfloor\mathcal{P}^{\text{IP3}}N/2\rfloor$ times for creation of $\lfloor\mathcal{P}^{\text{IP3}}N/2\rfloor$ offspring, the random selection of a boundary solution and its progression with a random step length ensures that, across the generations, the spread expansion is fairly emphasized on all boundaries. Some key features of boundary progression are highlighted in Section S2 of S.D.

2) Gap Progression: The procedure for creating the offspring (Q_t^G) for better uniformity, using *gap progression*, is summarized in Algorithm 4 (AL.4). As per the rationale set earlier, it entails identification of the unassociated RVs (denoted as $\mathcal{R}^G$, line 1, AL.4), and the advancement of their respective nearest solutions in P_t to create new offspring, using an appropriate ML model, as discussed below.

1) *Identification of the solution to be advanced (lines 4–5, AL.4):* an RV $\vec{G}$ is randomly selected from $\mathcal{R}^G$. The solution nearest to $\vec{G}$ (by perpendicular distance) that is associated with some other RV, is identified as the solution to be advanced, and denoted as S_{start}.
2) *Selection of the ML model (lines 6–7, AL.4):* here, the objective in which S_{start} would undergo maximum transition in its advancement to $\vec{G}$ is identified, say f_m. Following this, ML_m is selected for subsequent use.
3) *Obtaining the search direction (lines 8–12, AL.4):* as highlighted earlier in Section IV-B, ML_m applied on

Algorithm 4: `Gap_Prog` (P_t, $\mathcal{R}$, ML, $[x^{\min}, x^{\max}]$, $[x^l, x^u]$, $\mathcal{P}^{\text{IP3}}$)

Input: Current population P_t, RVs $\mathcal{R}$, ML models ML_1–ML_M, bounds from Algorithm 2 $[x^{\min}, x^{\max}]$, variable bounds in problem definition $[x^l, x^u]$, proportion $\mathcal{P}^{\text{IP3}}$
Output: New solutions created Q_t^G

1 $\mathcal{R}^G \leftarrow$ All empty RVs in $\mathcal{R}$
2 $Q_t^G \leftarrow \emptyset$ % sized $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor \times n_{\text{var}}$
3 **for** $i = 1$ *to* $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ **do**
4 $\quad \vec{G} \leftarrow$ A randomly selected RV from $\mathcal{R}^G$
5 $\quad S_{\text{start}} \leftarrow$ Nearest solution to $\vec{G}$
6 $\quad \vec{\delta} \leftarrow \hat{F}(S_{\text{start}}) - \vec{G}$
7 $\quad m \leftarrow \text{argmax}_{m \in [1,M]} |\vec{\delta}|$
8 $\quad \bar{X}(S_{\text{start}}) \leftarrow$ Normalized $X(S_{\text{start}})$ using $x^{\min}$ and $x^{\max}$
9 $\quad \bar{d_X} \leftarrow ML_m(\bar{X}(S_{\text{start}}))$
10 $\quad d_X \leftarrow$ Denormalized $\bar{d_X}$ using $x^{\min}$ and $x^{\max}$
11 $\quad$ **if** $\vec{\delta}_m < 0$ **then**
12 $\quad\quad d_X \leftarrow -1 \times d_X$
13 $\quad X(S_{\text{new}}) \leftarrow X(S_{\text{start}}) + \lambda_G \times \hat{d}_X$ % $\hat{d}_X = d_X / \|d_X\|$
14 $\quad$ Boundary repair on $X(S_{\text{new}})$
15 $\quad Q_t^G \leftarrow Q_t^G \cup S_{\text{new}}$

to $X(S_{\text{start}})$ provides a potential search direction in X-space, say, d_X. Since ML_m was trained on the normalized dataset: (i) $X(S_{\text{start}})$ needs to be normalized using the bounds computed in Algorithm 2, before applying ML_m; and (ii) the normalized search direction obtained by applying ML_m, denoted by $\bar{d_X}$, needs to be denormalized using the same bounds, leading to d_X. In the case of $\Delta f_m < 0$, the d_X is reversed by multiplying each of its components with (-1), as depicted in line 12 (AL.4).

4) *Advancement and repair (lines 13–14, AL.4):* the advancement of S_{start}, leading to S_{new}, can be given by $X(S_{\text{new}}) = X(S_{\text{start}}) + \lambda_G \times \hat{d}_X$, where $\hat{d}_X$ is a unit vector along d_X (computed above), and λ_G is the step length. In the context of *boundary progression*, the founding principles enabling computation of the scaling factor and the step length were laid down. These principles apply, as such, in the context of *gap progression* too, just that the specific formulations differ owing to the contextual difference, as evident below.

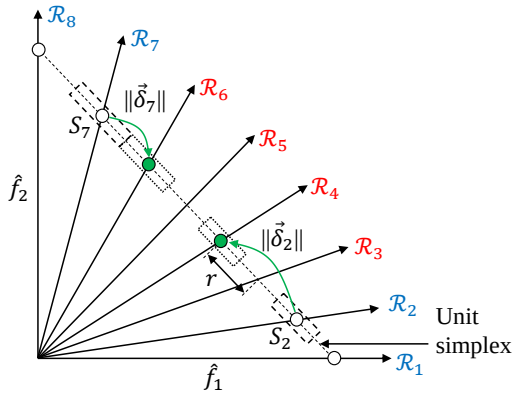


Fig. 5: A symbolic depiction of the gap progression on the unit simplex, during an RV-EMâO-IP3 run.

a) the scaling requirements in the F-space: this relates to the product of (i) the average distance r between two adjacent RVs on the unit simplex, and (ii) the number of adjacent-RV transitions required in F-space to arrive at a vacant RV ($\vec{G}$) from a starting solution (S_{start}). Notably, the RV underlying S_{start} and $\vec{G}$ may or may not be adjacent, as symbolically depicted in Figure 5. In that, if $\vec{\delta}$ is defined as $\vec{\delta} \equiv \hat{F}(S_{\text{start}}) - \vec{G}$, then for advancement of: (i) $S_{\text{start}} \equiv S_2$ to non-adjacent $\vec{G} \equiv \mathcal{R}_4$, $\|\vec{\delta}_2\|/r = 2$, and (ii) $S_{\text{start}} \equiv S_7$ to adjacent $\vec{G} \equiv \mathcal{R}_6$, $\|\vec{\delta}_7\|/r = 1$. Hence, the scaling factor required on the unit simplex can be given by $\|\vec{\delta}\|/r \times r$.

b) the inherent relationship of scales in F-space and X-space in the ML training-dataset: in analogy to the case of *boundary progression*, the step length in the current context of *gap progression* can be given by $\lambda_G = \|\vec{\delta}\|/r \times \|d_A\|$.

Notably, λ_G needs to be adapted to ensure that even when S_{start} may not lie exactly on the RV it is associated with (non-integer: $\|\vec{\delta}\|/r$), λ_G must still facilitate its advancement onto an unassociated RV. For instance, in Figure 5, both S_2 and S_7 may lie anywhere within the engulfing rectangle shown around $\mathcal{R}_2$ and $\mathcal{R}_7$, respectively. However, despite this uncertainty, λ_G should ensure S_2's and S_7's advancement within the engulfing rectangles shown around $\mathcal{R}_4$ and $\mathcal{R}_6$, respectively. Considering that each of the engulfing rectangles are bound by $\pm 0.5r$ around the underlying RV, the goal could be achieved by adapting λ_G as: $\lambda_G = rand(\lfloor \|\vec{\delta}\|/r \rceil - 0.5, \lfloor \|\vec{\delta}\|/r \rceil + 0.5) \times \|d_A\|$. Notably, the obtained $X(S_{\text{new}})$ may have some out-of-bound variables, which can be repaired using the same method as used in *boundary progression*.

While the above steps are repeated $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ times for creating $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ offspring solutions, the random selection of the unassociated RVs ensures that each identified gap in the current population is fairly emphasized.

D. IP3 Operator's Integration with RV-EMâOAs

This subsection highlights the integration of the IP3 operator with an RV-EMâOA, leading to RV-EMâO-IP3. Towards it, first the IP3 operator is defined as a self-sufficient function, as presented in Algorithm 5 (AL.5).

Algorithm 5: `IP3`($P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}}$)

Input: Parent population P_t, RV set $\mathcal{R}$, neighbourhood radius r, original variable bounds $[x^l, x^u]$, proportion $\mathcal{P}^{\text{IP3}}$
Output: Offspring solutions Q_t^{IP3}

1 $\mathcal{D} \leftarrow$ `Dataset_Construction`$(P_t, \mathcal{R}, r)$
2 $ML, [x^{\min}, x^{\max}] \leftarrow$ `ML_Training`$(\mathcal{D}, [x^l, x^u])$
3 $Q_t^B \leftarrow$ `Bound_Prog`$(P_t, \mathcal{R}, ML, [x^{\min}, x^{\max}], [x^l, x^u], \mathcal{P}^{\text{IP3}})$
4 $Q_t^G \leftarrow$ `Gap_Prog`$(P_t, \mathcal{R}, ML, [x^{\min}, x^{\max}], [x^l, x^u], \mathcal{P}^{\text{IP3}})$
5 $Q_t^{\text{IP3}} \leftarrow Q_t^B \cup Q_t^G$

AL.5 includes: (a) construction of M training-datasets $\mathcal{D}_1$–$\mathcal{D}_M$ using AL.1; (b) training of M ML models on $\mathcal{D}_1$–$\mathcal{D}_M$ using AL.2; (c) creation of $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ offspring, Q_t^B, using AL.3; (d) creation of $\lfloor \mathcal{P}^{\text{IP3}} N/2 \rfloor$ offspring, Q_t^G, using AL.4; and (e) merging of Q_t^B and Q_t^G, leading to $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring, denoted as Q_t^{IP3}.

Further, the integration of the IP3 operator with NSGA-III (an RV-EMâOA), leading to NSGA-III-IP3, is summarized below in Algorithm 6 (AL.6). It involves a frequency parameter, namely $t^{\text{IP3}}_{\text{freq}}$, which specifies the number of generations between two successive invocations of the IP3 operator. In AL.6, first the prerequisite condition for invocation of the IP3 operator—i.e., mild population stabilization—is checked (line 1). To do that, a stabilization tracking algorithm [31] has been used[2]. If stability is detected, the $startIP3$ flag is marked as $True$ (lines 2–3, AL.6). It is important to let the population mildly stabilize (ensuring some degree of convergence) before applying the IP3 operator, else, pre-mature focus on diversity enhancement may lead to a delayed convergence, at the cost of additional computational expense. Subsequently:

- if $t^{\text{IP3}}_{\text{freq}}$ generations have passed after IP3's last invocation (at $t = t^{\text{IP3}}_{\text{pg}}$), then IP3 is invoked, leading to creation of: $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring (Q^{IP3}_t) using the `IP3` function; and rest $\lceil (1 - \mathcal{P}^{\text{IP3}}) N \rceil$ offspring (Q^{V}_t) using the variation operators, leading to Q_t, sized N (lines 4–8, AL.6).
- otherwise, if IP3 is not invoked, all N offspring (Q_t) are created using the variation operators (lines 9–10, AL.6).
- the offspring Q_t are evaluated and NSGA-III's survival selection procedure is executed (lines 11–12, AL.6).
- the count of offspring N^{surv}_t that survived to the next generation is estimated. In a generation where IP3 is invoked, if this count has improved compared to the previous generation, implying good performance of the IP3 operator, then $t^{\text{IP3}}_{\text{freq}}$ is reduced by 1, yielding a more frequent progression. Otherwise, if the count has reduced, $t^{\text{IP3}}_{\text{freq}}$ is increased by 1 (lines 13–16, AL.6).

Algorithm 6: Generation t of NSGA-III-IP3

Input: RV set $\mathcal{R}$, original variable bounds $[x^l, x^u]$, Parent population P_t, frequency $t^{\text{IP3}}_{\text{freq}}$, last invocation $t^{\text{IP3}}_{\text{pg}}$, neighbourhood radius r, number of survived offspring in $(t-1)^{\text{th}}$ generation N^{surv}_{t-1}, proportion $\mathcal{P}^{\text{IP3}}$
Output: P_{t+1}, $t^{\text{IP3}}_{\text{pg}}$, $t^{\text{IP3}}_{\text{freq}}$, N^{surv}_t

1 $check \leftarrow$ `Check_Mild_Stabilization()`
2 **if** $check$ **then**
3 $\quad startIP3 = True$
4 **if** $startIP3$ *and* $t - t^{\text{IP3}}_{\text{pg}} = t^{\text{IP3}}_{\text{freq}}$ **then**
5 $\quad Q^{\text{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$ `% AL.5`
6 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`(P_t)
7 $\quad Q_t \leftarrow Q^{\text{IP3}}_t \cup Q^{\text{V}}_t$
8 $\quad t^{\text{IP3}}_{\text{pg}} \leftarrow t$
9 **else**
10 $\quad Q_t \leftarrow$ `Variation`(P_t)
11 Evaluate Q_t
12 $P_{t+1} \leftarrow$ `Survival_selection`$(P_t \cup Q_t)$
13 $N^{\text{surv}}_t \leftarrow sizeof(Q_t \cap P_{t+1})$
14 **if** $t = t^{\text{IP3}}_{\text{pg}}$ **then**
15 $\quad$ **if** $N^{\text{surv}}_t > N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} - 1$
16 $\quad$ **if** $N^{\text{surv}}_t < N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} + 1$

Further, to facilitate the implementation of the IP3 operator with other RV-EMâOAs, its integration with θ-DEA, MOEA/DD and LHFiD, is detailed in Section S3 of S.D.

[2]The same algorithm can be used for: (a) triggering the IP3 operator with a mild setting, and (b) terminating an RV-EMâOA run with a strict setting.

Notably, $t^{\text{IP3}}_{\text{freq}}$ and $\mathcal{P}^{\text{IP3}}$ are critical parameters. As evident above, $t^{\text{IP3}}_{\text{freq}}$ is adaptive and can assume any value (≥ 1) depending on the survival of Q^{IP3}. Further, this paper recommends and uses $\mathcal{P}^{\text{IP3}} = 50\%$, whose rationale is as follows: (a) in generations where IP3 is invoked, Q^{V} contributes to 50% of N, and (b) in generations where IP3 is not invoked (owing to $t^{\text{IP3}}_{\text{freq}} \geq 1$), Q^{V} contributes to 100% of N. These facts ensure that over all the generations of an RV-EMâOA, the role of natural variation operators contributing to convergence-diversity-neutral offspring (Q^{V}) is *not* superseded by the role of pro-diversity IP3 operator. This feature, symbolically depicted through a dominant share of Q^{V} over Q^{IP3} in Figure 6, is critical towards addressal of convergence-diversity balance and risk-reward tradeoff considerations, highlighted upfront. The above argument can be extended to RV-EMâOA-IP2$^+$ as well (Figure 6), given that $\mathcal{P}^{\text{IP2}^+} = 50\%$ is used.

Source of offspring solutions that are subjected to selection	Linkage of offspring solutions with the dual goals in EMâOAs	
	Convergence	Diversity
RV-EMâO	Q^{V}	
RV-EMâO-IP2$^+$	Q^{IP2^+}	Q^{V}
RV-EMâO-IP3	Q^{V}	Q^{IP3}
RV-EMâO-UIP	Q^{IP2^+}	Q^{V} Q^{IP3}

Fig. 6: Symbolic depiction of the degree of operators' contribution to convergence and diversity, over an entire run of RV-EMâO-IP2$^+$/IP3/UIP. The offspring created through natural variation operators Q^{V} are represented by a different color since they are convergence-diversity-neutral.

V. UIP Operator for Convergence and Diversity Enhancement

This section marks the culminating contribution of this paper, through proposition of the UIP operator for simultaneous convergence and diversity enhancement along an RV-EMâOA run. Towards it, the UIP operator relies on invoking either or both of IP2$^+$ and IP3 in the same generation, to produce both pro-convergence and pro-diversity offspring.

The key challenges in the above endeavor relate to:

1) avoiding any new parameter to govern independent or simultaneous invocations of the IP2$^+$ and IP3 operators: towards it, the proposed UIP operator scopes for invocation of both these constituent operators, such that, for *each* operator:
   - the criterion for its *first* invocation, remains the same as is, for its standalone invocation.
   - its successive invocation is guided by an independent parameter (as before), implying, $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ for the case of IP2$^+$ and IP3, respectively.
2) ensuring that the overarching considerations on convergence-diversity balance and ML driven risk-reward tradeoff are not violated: towards this, it may be noted that any generation t of an RV-EMâOA-UIP (UIP operator integrated with an RV-EMâOA), may be marked by one of the following scenarios:

- *neither IP2$^+$ nor IP3 operator is invoked*: in such a scenario, an RV-EMâOA-UIP shall operate like an RV-EMâOA. Hence, as depicted in Figure 6 for the case of an RV-EMâOA - the natural variation operators shall contribute to convergence-diversity-neutral offspring (Q^{V}), and the risk-reward associated with these ML based operators shall not be relevant.
- *either of IP2$^+$ or IP3 operator is invoked:* in such a scenario, the considerations of convergence-diversity balance and risk-reward tradeoff shall remain addressed, as explained earlier (Section IV-D, Figure 6).
- *both the IP2$^+$ and IP3 operators are simultaneously invoked:* in such a scenario the convergence-diversity balance is implicit, considering that $\mathcal{P}^{\mathrm{IP2}^+} = \mathcal{P}^{\mathrm{IP3}} = 50\%$ lead to an equal share of pro-convergence Q^{IP2^+} and pro-diversity Q^{IP3}. However, creating 100% of the offspring using ML-based operators ($Q^{\mathrm{IP2}^+} \cup Q^{\mathrm{IP3}}$) could be a risky proposition. To negate such risks, the UIP operator relies on imposing $t^{\mathrm{IP2}^+}_{\mathrm{freq}} \geq 2$ and $t^{\mathrm{IP3}}_{\mathrm{freq}} \geq 2$. These settings imply, that if both IP2$^+$ and IP3 operators are invoked in a particular generation (0% contribution of Q^{V}), then neither of them can be invoked in the next generation (100% contribution of Q^{V}). This constraint ensures that the overall contribution of Q^{V} remains 50% over two such consecutive generations, where the former generation is characterized by simultaneous invocation of IP2$^+$ and IP3 operators. This aligned with the fact that the nature of both $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ and $t^{\mathrm{IP3}}_{\mathrm{freq}}$ is adaptive, and hence, their invocations may not coincide in some generations, implies that the overall contribution of Q^{V} across all the generations of an RV-EMâOA-UIP, remains at least 50%. This feature is symbolically depicted in Figure 6, where the overall share of Q^{V} can be seen to be dominant compared to Q^{IP2^+} and/or Q^{IP3}. In that, Q^{V} is separated from others through a fuzzy boundary, considering that its exact share cannot be determined a priori, since both $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ and $t^{\mathrm{IP3}}_{\mathrm{freq}}$ are adapted on-the-fly.

Given this background, a representative generation t of NSGA-III-UIP has been summarized in Algorithm 7 (AL.7). Here, first the target-archive T_t as required by the IP2$^+$ function (Algorithm S1 in S.D.) is updated (line 1, AL.7). Then the prerequisite conditions for invocations of IP2$^+$ and IP3 are checked, and if fulfilled, appropriate flags ($startIP2^+$, $startIP3$) which influence whether or not IP2$^+$ and IP3 are to be invoked, are triggered as *True* (lines 2–7, AL.7). In the subsequent generations:

- if $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ generations have passed after IP2$^+$'s last invocation (at $t = t^{\mathrm{IP2}^+}_{\mathrm{pg}}$), then $\lfloor \mathcal{P}^{\mathrm{IP2}^+} N \rfloor$ offspring are created using IP2$^+$, denoted as $Q^{\mathrm{IP2}^+}_t$ (lines 8–10, AL.7).
- similarly, if $t^{\mathrm{IP3}}_{\mathrm{freq}}$ generations have passed after IP3's last invocation (at $t = t^{\mathrm{IP3}}_{\mathrm{pg}}$), then $\lfloor \mathcal{P}^{\mathrm{IP3}} N \rfloor$ offspring are created using IP3, denoted as Q^{IP3}_t (lines 11–13, AL.7).
- if the total count of offspring created in the above two steps ($|Q^{\mathrm{IP2}^+}| + |Q^{\mathrm{IP3}}|$) is smaller than N, then the rest of the offspring (Q^{V}_t) are created using the natural variation operators (lines 14–15, AL.7).

Algorithm 7: Generation t of NSGA-III-UIP

Input: RV set $\mathcal{R}$, variable bounds $[x^l, x^u]$, parent population P_t, offspring survived $N^{\mathrm{surv(V)}}_{t-1}$
IP2$^+$-specific: target archive T_{t-1}, input archive A_t, frequency $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$, last invocation $t^{\mathrm{IP2}^+}_{\mathrm{pg}}$, proportion $\mathcal{P}^{\mathrm{IP2}^+}$
IP3-specific: neighbourhood radius r, frequency $t^{\mathrm{IP3}}_{\mathrm{freq}}$, last invocation $t^{\mathrm{IP3}}_{\mathrm{pg}}$, proportion $\mathcal{P}^{\mathrm{IP3}}$
Output: P_{t+1}, T_t, A_{t+1}, $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$, $t^{\mathrm{IP3}}_{\mathrm{freq}}$, $t^{\mathrm{IP2}^+}_{\mathrm{pg}}$, $t^{\mathrm{IP3}}_{\mathrm{pg}}$

1 $T_t \leftarrow$ `Update_Target_Archive`$(P_t, T_{t-1}, \mathcal{R})$
2 $check1 \leftarrow$ `Check_Non_Dominated`(P_t)
3 $check2 \leftarrow$ `Check_Mild_Stabilization`$()$
4 **if** $check1$ **then**
5 $startIP2^+ = True$
6 **if** $check2$ **then**
7 $startIP3 = True$
8 **if** $startIP2^+$ *and* $t - t^{\mathrm{IP2}^+}_{\mathrm{pg}} = t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ **then**
9 $Q^{\mathrm{IP2}^+}_t \leftarrow$ `IP2+`$(A_t, T_t, \mathcal{R}, [x^l, x^u], P_t, \mathcal{P}^{\mathrm{IP2}^+})$ `% AL.S1`
10 $t^{\mathrm{IP2}^+}_{\mathrm{pg}} \leftarrow t$
11 **if** $startIP3$ *and* $t - t^{\mathrm{IP3}}_{\mathrm{pg}} = t^{\mathrm{IP3}}_{\mathrm{freq}}$ **then**
12 $Q^{\mathrm{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\mathrm{IP3}})$ `% AL.5`
13 $t^{\mathrm{IP3}}_{\mathrm{pg}} \leftarrow t$
14 **if** $|Q^{\mathrm{IP2}^+}| + |Q^{\mathrm{IP3}}| < N$ **then**
15 $Q^{\mathrm{V}}_t \leftarrow$ `Variation`(P_t)
16 $Q_t \equiv Q^{\mathrm{IP2}^+}_t \cup Q^{\mathrm{IP3}}_t \cup Q^{\mathrm{V}}_t$
17 `Evaluate`(Q_t)
18 $A_{t+1} \leftarrow (A_t \cup Q_t \cup P_{t+1-t_{\mathrm{past}}}) \backslash [P_{t-t_{\mathrm{past}}} \cup Q_{t-t_{\mathrm{past}}}]$
19 $P_{t+1} \leftarrow$ `Survival_Selection`$(P_t \cup Q_t)$
20 **if** $t = t^{\mathrm{IP2}^+}_{\mathrm{pg}}$ **then**
21 $t^{\mathrm{IP2}^+}_{\mathrm{freq}} \leftarrow$ `Adapt`$(Q^{\mathrm{IP2}^+}_t, P_{t+1}, Q^{\mathrm{V}}_{t-1}, N^{\mathrm{surv(V)}}_{t-1}, t^{\mathrm{IP2}^+}_{\mathrm{freq}})$
22 **if** $t = t^{\mathrm{IP3}}_{\mathrm{pg}}$ **then**
23 $t^{\mathrm{IP3}}_{\mathrm{freq}} \leftarrow$ `Adapt`$(Q^{\mathrm{IP3}}_t, P_{t+1}, Q^{\mathrm{V}}_{t-1}, N^{\mathrm{surv(V)}}_{t-1}, t^{\mathrm{IP3}}_{\mathrm{freq}})$

- all offspring, namely, $Q^{\mathrm{IP2}^+}_t$, Q^{IP3}_t and Q^{V}_t are merged into Q_t (sized N) and evaluated. (lines 16–17, AL.7)
- A_{t+1} is updated (required by `IP2+` function), followed by NSGA-III's survival selection (lines 18–19, AL.7).
- finally, $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ and $t^{\mathrm{IP3}}_{\mathrm{freq}}$ are adapted, if the respective operators were invoked in the current generation t (lines 20–23, AL.7). Details of the `Adapt` function (based on offspring survival) are provided in Section S3-B of S.D.

Further, to facilitate the implementation of the UIP operator with other RV-EMâOAs, its integration with θ-DEA, MOEA/DD and LHFiD, is detailed in Section S3 of S.D.

Moreover, the computational complexities of the IP3 and UIP operators have been discussed in Section S4 of S.D.

VI. Experimental Settings

This section sets the foundation for experimental investigation, by highlighting the: (a) test-suite, (b) performance indicators and related statistical analysis, and (c) parameters pertaining to the RV-EMâOAs, IP2$^+$, IP3 and UIP operators.

A. Test-suite

For multi-objective instances, problems are used with the following specifications:

- DTLZ [32]: with distance variables $k = 20$ and $M = 3$.
- MaF [33]: with $k = 20$ and $M = 3$.

- ZDT [34]: their respective $g(X)$-functions are modified to have PO solutions at $x_k = 0.5$, for $k = 2, \ldots, 30$, and are referred to as $\tilde{\text{Z}}$DT here onward [7].
- CIBN [35]: with $n_{\text{var}} = 10$.
- DASCMOP [36]: with $n_{\text{var}} = 30$ and setting 5.
- MW [37]: with $n_{\text{var}} = 15$.

For many-objective, the 5-, 8- and 10-objective versions of the DTLZ and MaF problems are used, with $k = 20$ [8].

B. Performance Indicator and Statistical Analysis

For performance comparison, a hypervolume (HV) measure that accounts for both convergence and diversity has been used, with reference point $R_{1\times M} = [1 + \frac{1}{p}, \ldots, 1 + \frac{1}{p}]$ (p is the number of gaps set for the Das-Dennis method [38] while generating the RVs). Further,

- for problems where the scales of different objectives are different, the solutions are normalized in the F-space using the theoretical PF extremes.
- when comparing *only two* algorithms, the Wilcoxon ranksum test [39] is performed on the HV values reported over multiple/independently seeded runs. Here, the threshold p-value of 0.05 (95% confidence level) is used.
- when comparing *more than two* algorithms at a time, the Kruskal-Wallis test [40] with threshold p-value of 0.05 has been used, to infer whether or not their overall differences are statistically insignificant. If not, the Wilcoxon ranksum test is used for pairwise comparisons with the UIP variant as reference. Furthermore, the threshold p-value is adjusted using the standard Bonferroni correction [41], to retain the same overall confidence.

C. Parameter Settings

In this subsection, the parameters and settings used for: (a) RV-EMâOAs; (b) IP2 operator; (c) IP3 operator, and (d) UIP operator, have been discussed.

1) RV-EMâOA Settings: Notably, all RV-EMâOAs require a set of RVs, for which the DD method is used. The input p (gaps) for the DD method and the resulting population sizes N are highlighted in Table I. For $M = 5$, 8 and 10, two gap values are required to create boundary and interior points [42].

TABLE I: Parameter settings for the Das and Dennis method.

M	2	3	5	8	10
N	100	105	196	240	440
p (gaps)	99	13	5, 4	3, 3	3, 3

For all the experiments across the RV-EMâOAs, the SBX crossover ($p_c = 0.9$ and $\eta_c = 20$) and polynomial mutation ($p_m = 1/n_{\text{var}}$ and $\eta_m = 20$) are used. Apart from these, the RV-EMâOA-specific parameters are highlighted below.

- θ-DEA: the penalty parameter: $\theta = 5$ [43].
- MOEA/DD: neighbourhood size $T = 20$; and mating restriction probability $\delta = 0.9$ [44].
- LHFiD: objective preferences $p_m = 1/w_m \; \forall \; m \in [1, M]$, where $w_m \equiv m^{\text{th}}$ component of the underlying RV; and mild stabilization parameters $\psi_{\text{mild}} = \{2, 20\}$ [4].

For each test instance, the performance of each RV-EMâOA has been assessed over its runs with 31 random seeds. To avoid an arbitrary fixation of the termination generations, t_{TM}, a stabilization tracking algorithm [31] has been included in RV-EMâO-UIP. This stabilization tracking algorithm requires a parameter set, kept as $\psi_{\text{TM}} \equiv \{3, 50\}$, which suggests the t_{TM} for RV-EMâO-UIP on-the-fly. The mean t_{TM} determined for RV-EMâO-UIP over 31 runs has been used as the t_{TM} for all variants of that RV-EMâOA (including RV-EMâO-UIP).

2) IP2$^+$ Operator Settings: The IP2$^+$ operator involves four parameters (same as IP2): $\mathcal{P}^{\text{IP2}^+}$, t_{past}, $t_{\text{freq}}^{\text{IP2}^+}$ and η. Here: (a) $\mathcal{P}^{\text{IP2}^+} = 50\%$ is used, considering the convergence-diversity balance and risk-rewards tradeoff as discussed in Sections I and IV-D, (b) $t_{\text{past}} = 5$ is used since the performance is not very sensitive to t_{past} [8], (c) $t_{\text{freq}}^{\text{IP2}^+}$ is made adaptive, similarly to $t_{\text{freq}}^{\text{IP3}}$ in Algorithm 6, while its initial value is set as 1, and (d) η assumes a random value in the range $[1, 1.5]$.

3) IP3 Operator Settings: The IP3 operator involves three parameters: $\mathcal{P}^{\text{IP3}}$, $t_{\text{freq}}^{\text{IP3}}$ and ψ_{mild}. $\mathcal{P}^{\text{IP3}} = 50\%$ is used based on the discussion at the end of Section IV-D. Further, $t_{\text{freq}}^{\text{IP3}}$ is adaptive (AL.6), and its initial value is set as 1, the same as its minimum value. The last parameter, ψ_{mild} governs the degree of stabilization required to trigger IP3's first invocation. While it is intuitive that ψ_{mild} should correspond to a lower degree of stabilization than the ψ_{TM} that is used to terminate the RV-EMâOA runs, its exact setting is borrowed from the suggestion made in [4, 31]. According to that, $\psi_{\text{mild}} = \{2, 20\}$.

4) UIP Operator Settings: The UIP operator does not involve any additional parameters, other than the ones that are a part of either IP2$^+$ or IP3 operators, for which the details have been provided above. The only change is that the initial values of $t_{\text{freq}}^{\text{IP2}^+}$ and $t_{\text{freq}}^{\text{IP3}}$ have been set as 2 instead of 1, which aligns with their respective minimum values. Additionally, there is a constraint, given by $\mathcal{P}^{\text{IP2}^+} + \mathcal{P}^{\text{IP3}} \leq 100\%$. Violating this constraint would lead to creation of more than N offspring in a generation where both IP2$^+$ and IP3 are invoked, leading to additional SEs over the base RV-EMâOA.

VII. Experimental Results

This section aims to investigate the search efficacy infused by the proposed operators into RV-EMâOAs. All investigations are executed using the pymoo library [45]. Towards that end:

- IP2$^+$: since it is only a variant of the existing IP2 operator, its efficacy vis-à-vis IP2 has been discussed in Section S5 of S.D. There, it has been observed that IP2$^+$ performed statistically better or equivalently in the majority of the test instances, while reducing the number of hard-fixed but tunable parameters from four to two.
- IP3: its efficacy has been realized through a comparison of the IP3 variants of NSGA-III, θ-DEA, MOEA/DD and LHFiD, with their respective base variants, on multi-objective instances.
- UIP: first, its efficacy has been investigated vis-à-vis the IP2$^+$ and IP3 operators, through a comparison of: (a) UIP and IP2$^+$ variants, and (b) UIP and IP3 variants, of all considered RV-EMâOEAs on multi-objective instances. Second, to investigate its scalability vis-à-vis M, its efficacy has been investigated on many-objective instances.

TABLE II: HV based comparison of IP2$^+$, IP3 and UIP operators, performed independently for NSGA-III, θ-DEA and MOEA/DD. Each column shows the median HV (from 31 runs) at the end of t_{TM} generations, determined on-the-fly for the respective UIP variants. The symbols "+", "=" or "–" against each variant highlight where these are statistically better than, comparable to, or worse than the corresponding UIP variant, respectively. Similar comparison for LHFiD is shown in Table S6.

Problem	M	t_{TM}	NSGA-III	NSGA-III -IP2$^+$	NSGA-III -IP3	NSGA-III -UIP	t_{TM}	θ-DEA	θ-DEA-IP2$^+$	θ-DEA-IP3	θ-DEA-UIP	t_{TM}	MOEA/DD	MOEA/DD -IP2$^+$	MOEA/DD -IP3	MOEA/DD -UIP
CIBN1	2	1365	0.327867–	0.327755–	0.482560–	0.504742	1367	0.335619–	0.335619–	0.335619–	0.508695	1325	0.334598–	0.340659–	0.488823=	0.488058
CIBN2	2	789	0.658712–	0.669290–	0.669122=	0.669583	811	0.667394–	0.668774–	0.669350=	0.670158	766	0.668834–	0.669077–	0.669608=	0.669695
CIBN3	2	960	0.213584–	0.215653–	0.219614–	0.226977	1013	0.213996–	0.215127–	0.219928–	0.225995	914	0.214547–	0.215170–	0.221714–	0.226747
CIBN4	3	439	0.912600–	0.911834=	0.920534=	0.920335	505	0.895468=	0.897117=	0.910920=	0.898464	480	0.896755=	0.895503=	0.911635=	0.896848
CIBN5	3	294	0.629637+	0.628627=	0.629868+	0.628892	299	0.619556=	0.619315=	0.618989=	0.619234	305	0.622079=	0.622057=	0.621415=	0.621281
DASCMOP1	2	2101	0.089766–	0.088922–	0.320434–	0.332737	1889	0.087565–	0.087565–	0.087565–	0.332968	1944	0.132878–	0.132680–	0.332522–	0.333200
DASCMOP2	2	1944	0.414610–	0.411890–	0.645024–	0.666735	1851	0.415182–	0.415182–	0.415182–	0.664793	1963	0.472963–	0.472220–	0.667765–	0.669519
DASCMOP3	2	1658	0.391774–	0.381741–	0.396188=	0.399082	1836	0.398531–	0.395109–	0.400589–	0.489664	1866	0.408387–	0.409257–	0.483243=	0.483395
DASCMOP4	2	1993	0.336866=	0.336881=	0.336810=	0.336960	2006	0.335362=	0.335592=	0.335716=	0.335532	2183	0.335677=	0.335763=	0.335776=	0.335616
DASCMOP5	2	2113	0.672667=	0.672650=	0.672740=	0.672663	2056	0.671229=	0.671435=	0.671475=	0.671456	2293	0.671290=	0.671291=	0.671540=	0.671308
DASCMOP6	2	2432	0.549906–	0.574574=	0.574880=	0.574797	2536	0.574568–	0.574701–	0.574908=	0.574866	2024	0.575065=	0.575090=	0.575103=	0.575132
DASCMOP7	3	1731	1.027245=	1.026331=	1.026493=	1.025053	1767	1.014610=	1.012289=	1.015271=	1.011353	1888	1.016219=	1.016695=	1.016602=	1.017115
DASCMOP8	3	1675	0.630175=	0.656181=	0.659697=	0.628234	1751	0.648435=	0.642712=	0.645123=	0.632479	1850	0.651319=	0.652573=	0.652680=	0.651529
DASCMOP9	3	1537	0.346493–	0.352620–	0.647141=	0.647739	1524	0.451661–	0.411027–	0.435853–	0.643508	1511	0.638729–	0.640038–	0.647199=	0.647314
DTLZ1	3	1497	1.221639=	1.222229=	1.222352=	1.222575	1118	1.213503+	1.215133+	1.216521+	0.873703	1136	1.216124=	1.216929+	1.216587+	1.202253
DTLZ2	3	978	0.667327=	0.667309=	0.667309=	0.667323	963	0.651611=	0.650687=	0.651717=	0.651758	912	0.653244=	0.652445=	0.653327=	0.652884
DTLZ3	3	1750	0.652251–	0.656486–	0.660367=	0.662523	1237	0.611978=	0.613765+	0.624126+	0.085917	981	0.551322=	0.538874+	0.550773+	0.000000
DTLZ4	3	1449	0.667309–	0.667331=	0.667307–	0.667346	1935	0.652918=	0.654291=	0.653936=	0.654825	1226	0.654143–	0.653867–	0.655467=	0.655937
MaF1	3	608	0.235973+	0.234828=	0.236232+	0.235082	618	0.228233=	0.228967=	0.228714=	0.229025	625	0.229522=	0.229064=	0.229199=	0.227796
MaF2	3	486	0.396887=	0.396552=	0.396665=	0.396411	515	0.390904=	0.391264=	0.391494=	0.391698	513	0.393007=	0.392538–	0.393789=	0.393700
MaF3	3	2135	1.193651=	1.193381=	1.194075=	1.194107	2123	1.193223=	1.194325=	1.195539+	1.194434	2133	1.191562–	1.193389–	1.194346=	1.194840
MaF4	3	1214	0.612050–	0.621803=	0.625814=	0.629769	1300	0.610459–	0.614759=	0.618814=	0.620879	1295	0.613593=	0.616997=	0.622302=	0.617812
MaF5	3	1345	1.227613=	1.227604=	1.227614=	1.227604	1831	1.228640=	1.228139–	1.228279–	1.228654	1503	1.228484=	1.228388=	1.228515=	1.228521
MaF7	3	1201	0.375791=	0.376011=	0.375918=	0.376159	1166	0.371659=	0.371891=	0.372390=	0.372502	1118	0.371731=	0.371436=	0.371923=	0.371782
MaF8	3	1244	0.463948–	0.464591–	0.467241=	0.466653	1532	0.000331=	0.000284=	0.000243=	0.000170	1608	0.000109=	0.000173=	0.000106=	0.000117
MaF9	3	1160	0.626751=	0.626751=	0.626755=	0.626729	1155	0.610583–	0.613877=	0.614781=	0.614858	1227	0.610000–	0.612475=	0.613873=	0.613124
MaF10	3	996	0.528291+	0.520969=	0.526276+	0.519436	1019	0.477226–	0.480306–	0.497108+	0.492448	1025	0.508591+	0.507794+	0.506016+	0.499053
MaF11	3	993	0.980497=	0.980142=	0.980725=	0.979771	1071	1.144277=	1.137190=	1.141846=	1.138350	998	1.142115=	1.143221=	1.141620=	1.142002
MaF12	3	725	0.599802–	0.614806+	0.594912–	0.612526	743	0.534650–	0.596054+	0.535897–	0.590104	761	0.535199–	0.589841=	0.535312–	0.592456
MaF13	3	1065	0.365532–	0.371873–	0.371356–	0.377158	1021	0.366427=	0.365610=	0.367100=	0.369516	1174	0.501206=	0.497252–	0.515203=	0.524071
MW1	2	1042	0.415304=	0.415267=	0.415293=	0.415353	1022	0.414481=	0.414559=	0.414484=	0.414439	1074	0.413709–	0.413856–	0.414167=	0.414543
MW2	2	848	0.483094=	0.483050=	0.482937=	0.491251	833	0.482597–	0.482288=	0.482653–	0.489976	891	0.482312–	0.489935=	0.482202–	0.490426
MW3	2	868	0.469880=	0.469743=	0.469740=	0.469568	1050	0.469747=	0.469662=	0.469746=	0.469411	971	0.469695=	0.469676=	0.469719=	0.469723
MW4	3	743	1.041376=	1.041454+	1.041260=	1.041269	756	1.024349=	1.026268=	1.028647+	1.026601	755	1.024167–	1.023859–	1.028760=	1.028355
MW5	2	1783	0.083010–	0.141081–	0.196301=	0.198466	1632	0.100401–	0.091867–	0.198217=	0.200033	1453	0.195836=	0.198941=	0.199983=	0.200113
MW6	2	1244	0.298309–	0.298281–	0.298408–	0.317611	1212	0.297263=	0.297277=	0.297059=	0.310665	1272	0.296733=	0.315600=	0.296643=	0.309847
MW7	2	906	0.366397=	0.366423=	0.366483=	0.366207	905	0.365294=	0.365294=	0.365301=	0.365080	890	0.365752=	0.365995=	0.366083=	0.366039
MW8	3	732	0.626663=	0.626527=	0.626711=	0.627444	732	0.609678=	0.606148=	0.609363=	0.612839	668	0.615115=	0.623660=	0.619145=	0.624875
MW9	2	1063	0.293673–	0.292268–	0.296315=	0.295451	986	0.293277–	0.292758–	0.295043=	0.295267	1128	0.293964=	0.294199=	0.295363=	0.293913
MW10	2	1071	0.247084=	0.246970=	0.247373=	0.268710	1032	0.246414=	0.246275=	0.246637=	0.263219	1054	0.261012=	0.289332=	0.261707=	0.289881
MW11	2	976	0.268232+	0.264411=	0.264602=	0.257800	987	0.265164=	0.264203=	0.264145=	0.208145	1087	0.265863=	0.258493=	0.260028=	0.254258
MW12	2	1055	0.570528=	0.570625–	0.570794=	0.570699	988	0.570107–	0.570203–	0.570372=	0.570262	1085	0.570166–	0.570272–	0.570424=	0.570440
MW13	2	1001	0.328966–	0.329740–	0.328578–	0.346489	990	0.322834=	0.322893–	0.322720–	0.336527	945	0.322475=	0.336078=	0.321894=	0.334442
MW14	3	848	0.152752=	0.153890=	0.151446=	0.154984	894	0.156222=	0.160729=	0.157497=	0.155941	960	0.164248=	0.166413=	0.160076=	0.168807
ŽDT1	2	1198	0.681860=	0.681860=	0.681859=	0.681859	1161	0.68128=	0.681305=	0.681256=	0.681299	1155	0.681215=	0.681179=	0.681171=	0.681193
ŽDT2	2	1267	0.348794=	0.348794=	0.348793=	0.348794	1237	0.347854=	0.347807=	0.347817=	0.347861	1247	0.347746=	0.347795=	0.347733=	0.347800
ŽDT3	2	1007	1.068445=	1.068408=	1.068450=	1.068492	977	1.067375=	1.067368=	1.067415=	1.067561	1005	1.067135=	1.066854=	1.067396=	1.067364
ŽDT4	2	1767	0.681859=	0.681860=	0.681860=	0.681860	1744	0.681257=	0.681191=	0.681227=	0.681255	1675	0.681196=	0.681201=	0.681177=	0.681192
ŽDT6	2	1836	0.312752–	0.324640–	0.333501–	0.336457	1822	0.313645–	0.320266–	0.331430=	0.334052	1771	0.313384–	0.323259–	0.332291=	0.332646
Total (+/=/–) $\longrightarrow$				04/25/**20**	02/31/**16**	03/36/**10**	of 49 probs.		01/31/**17**	03/31/**15**	05/34/**10**	of 49 probs.		01/32/**16**	03/31/**15**	03/41/**05**	of 49 probs.

A. Performance of IP3 operator on multi-objective instances

In this subsection, for each of the algorithms: NSGA-III, θ-DEA, MOEA/DD and LHFiD, its base version is compared with its respective IP3 variants. The median HV values are shown in Table II (along with Table S6 in S.D.), where one can see:

- NSGA-III-IP3 performs statistically better in 18 (out of 49) instances, and statistically better or equivalently in 47 (out of 49) instances, compared to NSGA-III.
- θ-DEA-IP3 performs statistically better in 14 (out of 49) instances, and statistically better or equivalently in 47 (out of 49) instances, compared to θ-DEA.
- MOEA/DD-IP3 performs statistically better in 16 (out of 49) instances, and statistically better or equivalently in 48 (out of 49) instances, compared to MOEA/DD.
- LHFiD-IP3 performs statistically better in 11 (out of 49) instances, and statistically better or equivalently in 42 (out of 49) instances, compared to LHFiD.

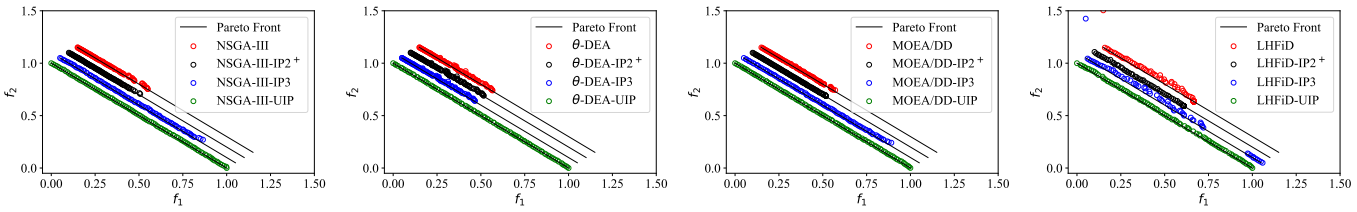
Overall, the IP3 variants are statistically better in about 30% of the instances, and statistically better or equivalent in about 94% of the instances. This performance enhancement could be directly attributed to the search efficacy infused by the pro-diversity IP3 operator, when integrated with an RV-EMâOA.

B. Performance of UIP operator on multi-objective instances

This subsection presents the assessment of: (a) UIP vis-à-vis IP2$^+$; and (b) UIP vis-à-vis IP3, to collectively establish the efficacy of the UIP operator, over the pro-convergence IP2$^+$ and pro-diversity IP3 operators.

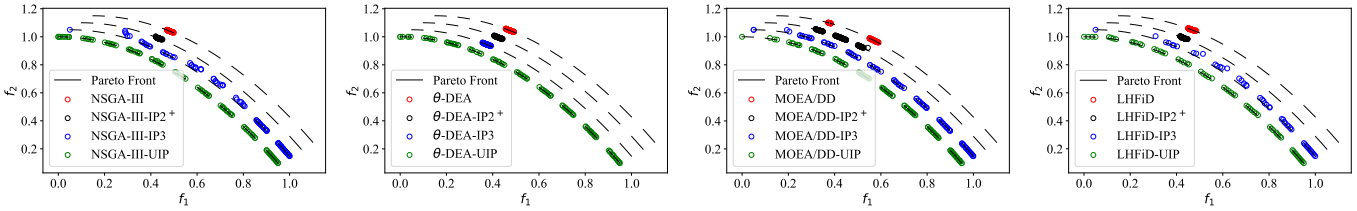
1) Performance of UIP vis-à-vis IP2$^+$ Operator: Here, the performance comparison of UIP and IP2$^+$ operators has been realized through a direct comparison of the UIP and IP2$^+$ variants of each of: NSGA-III, θ-DEA, MOEA/DD and LHFiD. The median HV values are shown in Table II (along with Table S6 in S.D.), where one can see:

- NSGA-III-UIP performs statistically better in 16 (out of 49) instances, and statistically better or equivalently in 47 (out of 49) instances, compared to NSGA-III-IP2$^+$.
- θ-DEA-UIP performs statistically better in 15 (out of 49) instances, and statistically better or equivalently in 46 (out of 49) instances, compared to θ-DEA-IP2$^+$.



(a) NSGA-III (b) θ-DEA (c) MOEA/DD (d) LHFiD

Fig. 7: Results with integration of proposed operators with different RV-EMâOAs on CIBN1. For visual clarity, the reference PFs have been slightly displaced for different variants of same RV-EMâOA.



(a) NSGA-III (b) θ-DEA (c) MOEA/DD (d) LHFiD

Fig. 8: Results with integration of proposed operators with different RV-EMâOAs on DASCMOP1. For visual clarity, the reference PFs have been slightly displaced for different variants of the same RV-EMâOA.

- MOEA/DD-UIP performs statistically better in 15 (out of 49) instances, and statistically better or equivalently in 46 (out of 49) instances, compared to MOEA/DD-IP2$^+$.
- LHFiD-UIP performs statistically better in 9 (out of 49) instances, and statistically better or equivalently in 47 (out of 49) instances, compared to LHFiD-IP2$^+$.

Overall, the UIP variants are statistically better in about 28% of the instances, and statistically better or equivalent in about 95% of the instances, compared to the IP2$^+$ variants. This supports the simultaneous use of pro-convergence and pro-diversity offspring (as in UIP), over pro-convergence offspring only (as in IP2$^+$).

2) Performance of UIP vis-à-vis IP3 Operator: Here, the performance comparison of UIP and IP3 operators has been realized through a direct comparison of the UIP and IP3 variants of each of: NSGA-III, θ-DEA, MOEA/DD and LHFiD. The median HV values are shown in Table II (along with Table S6 in S.D.), where one can see:

- NSGA-III-UIP performs statistically better in 10 (out of 49) instances, and statistically better or equivalently in 46 (out of 49) instances, compared to NSGA-III-IP3.
- θ-DEA-UIP performs statistically better in 10 (out of 49) instances, and statistically better or equivalently in 44 (out of 49) instances, compared to θ-DEA-IP3.
- MOEA/DD-UIP performs statistically better in 5 (out of 49) instances, and statistically better or equivalently in 46 (out of 49) instances, compared to MOEA/DD-IP3.
- LHFiD-UIP performs statistically better in 8 (out of 49) instances, and statistically better or equivalent in 49 (out of 49) instances, than LHFiD-IP3.

Overall, the UIP variants are statistically better in about 17% of the instances, and statistically better or equivalently in about 94% of the instances, compared to the IP3 variants. This supports the simultaneous use of pro-convergence and pro-diversity offspring (as in UIP), over pro-diversity offspring only (as in IP3).

For further insights into the performance of IP3 and UIP operators, some sample test instances have been chosen for discussion, including: CIBN1 and DASCMOP1 (Figures 7 and 8). These problems are characterized as diversity-hard, since the base RV-EMâOAs fail to achieve a reasonable PF approximation. From Figures 7 and 8, it may be noted that:

- the IP2$^+$ variants offer little to no improvement over the respective base variants, since a pro-convergence operator may not help in diversity enhancement.
- the IP3 variants offer a better PF-approximation than the respective base variants, supporting the diversity enhancement offered by the IP3 operator.
- the UIP variants offer the best PF-approximation, compared to other variants.

Moreover, some insights into the adaptation of $t_{\text{freq}}^{\text{IP2}^+}$ and $t_{\text{freq}}^{\text{IP3}}$ are provided in Section S8 of S.D.

C. Performance of UIP operator on many-objective instances

The above subsections clearly reveal the distinct advantage offered by integrating the UIP operator with an RV-EMâOA on multi-objective problems. This subsection aims to extend the above discussion to more difficult many-objective problems. Towards this end, the so far best-performing UIP operator is integrated with each of: NSGA-III, θ-DEA, MOEA/DD and LHFiD, and the performance is compared with the respective base versions. The median HV values, as shown in Table III, reveal the following:

- NSGA-III-UIP performs statistically better in 14 (out of 48) instances, and statistically better or equivalently in 43 (out of 48) instances, compared to NSGA-III.

TABLE III: HV based performance comparison of NSGA-III-UIP, θ-DEA-UIP, MOEA/DD-UIP and LHFiD-UIP with their respective base variants, i.e., NSGA-III, θ-DEA, MOEA/DD and LHFiD, respectively. For each test instance of $M = 5$, 8 and 10, median hypervolume is shown at the end of t_{TM} generations determined on-the-fly for the respective UIP variants. The symbols "+", "=" or "–" against each base variant highlight where these are statistically better than, comparable to, or worse than the corresponding UIP variant, respectively.

Problem	M	t_{TM}	NSGA-III	NSGA-III-UIP	t_{TM}	θ-DEA	θ-DEA-UIP	t_{TM}	MOEA/DD	MOEA/DD-UIP	t_{TM}	LHFiD	LHFiD-UIP
DTLZ1	5	1367	2.486820=	2.487010	748	2.157168=	2.155684	883	2.670244=	2.456018	859	4.207928=	4.052889
	8	1604	9.988500=	9.988620	875	9.981730=	9.979700	853	6.960960+	5.976893	869	9.988074=	9.982169
	10	2007	17.757700=	17.757700	833	0.000000=	0.000000	808	16.139124+	15.307207	1223	17.757704=	17.757702
DTLZ2	5	860	2.172040=	2.172090	941	2.121200=	2.117510	894	2.121840=	2.120590	578	3.891166=	3.890823
	8	777	9.816830–	9.818580	905	9.763760+	9.757400	839	9.758620=	9.755060	644	9.847959=	9.847937
	10	793	17.665800–	17.667500	877	17.591500=	17.592100	834	17.574200=	17.566900	676	17.694772=	17.694515
DTLZ3	5	1071	1.938460+	0.000000	765	2.082338=	2.012005	904	0.260120=	0.066750	776	0.783335+	0.000000
	8	1537	9.741440=	9.759030	847	8.546232+	7.622917	803	8.319002=	8.116630	863	9.245672=	0.014920
	10	1718	17.650300–	17.664100	819	0.000000=	0.000000	814	9.385090+	6.920623	881	17.633675=	17.105152
DTLZ4	5	912	2.173240=	2.173190	969	2.129440=	2.128680	960	2.133190=	2.130790	868	3.895547=	3.895753
	8	825	9.826530+	9.826270	857	9.731910+	9.707990	784	9.713380=	9.714600	841	9.849535=	9.849960
	10	820	17.677500=	17.677200	763	17.219400=	17.165100	754	17.296200=	17.248200	857	17.696040=	17.696277
MaF1	5	843	0.059222=	0.059753	844	0.056371=	0.056379	856	0.055339=	0.056772	828	0.037931=	0.037928
	8	989	0.015552=	0.015751	928	0.013510=	0.012873	839	0.013111=	0.013142	886	0.000128=	0.000129
	10	830	0.002220=	0.002241	773	0.001562–	0.001647	757	0.001787–	0.001837	862	0.000724=	0.000724
MaF2	5	359	0.997132=	0.998834	366	0.988566–	0.995460	376	0.981635=	0.984138	468	0.867556+	0.843094
	8	422	4.352760–	4.377980	454	4.122040–	4.177970	455	4.039790–	4.096930	580	1.290279+	1.175563
	10	487	7.846630=	7.856730	521	7.475370–	7.613500	535	7.321450–	7.494960	594	8.161162+	7.354750
MaF3	5	2238	2.488190–	2.488200	1209	1.977794=	1.866041	1842	2.483580=	2.378010	1729	2.160181=	2.160109
	8	3333	9.988720–	9.988720	612	0.000000=	0.000007	1504	0.000000=	0.000045	1453	3.432204+	3.432000
	10	3765	17.757700–	17.757700	461	0.000000=	0.000000	568	0.000000=	0.000237	1401	17.757727+	17.757130
MaF4	5	604	0.117321=	0.083836	863	0.192197=	0.220878	820	0.154150=	0.168207	1101	0.112496=	0.108572
	8	874	0.118441=	0.153381	963	0.032738=	0.045256	833	8.957987=	8.748836	1172	0.000273–	0.000533
	10	1069	0.064231–	0.077138	1222	0.026055–	0.034063	975	0.015330=	0.019916	1146	0.000000=	0.000000
MaF5	5	1109	2.487950=	2.487940	1157	2.487720=	2.487770	1173	2.487690–	2.487740	1039	2.160346–	2.161071
	8	1160	9.988720–	9.988720	1254	9.988720=	9.988720	1167	9.988720–	9.988720	970	3.431163–	3.432212
	10	1196	17.757700–	17.757700	1292	17.757700=	17.757700	1090	17.757700=	17.757700	1020	15.021723+	6.174841
MaF7	5	1087	0.970398=	0.969063	1090	0.956168=	0.955427	1126	0.968569+	0.964265	1244	0.657965–	0.716870
	8	912	4.402240=	4.390560	1022	4.222140–	4.262750	959	4.291200=	4.295080	1562	0.453748–	0.562075
	10	849	7.602060=	7.595800	766	6.468070=	6.712380	514	4.935440–	5.351250	1715	0.000000=	0.000000
MaF8	5	1546	0.475295=	0.476568	1767	0.000233–	0.000340	1853	0.000094–	0.000208	1675	0.000392+	0.000390
	8	1315	0.709058+	0.686091	1429	0.000517–	0.000570	1508	0.000390=	0.000382	1647	0.000003=	0.000003
	10	1214	0.577736=	0.572205	1317	0.000075=	0.000081	1438	0.000063=	0.000061	1595	0.000117=	0.000117
MaF9	5	985	0.027410=	0.027633	2037	0.028336=	0.028005	1607	0.026104=	0.027067	5632	0.022343–	0.024079
	8	716	0.000518–	0.002425	849	0.000833=	0.001078	745	0.000000=	0.000004	1132	0.000054=	0.000054
	10	774	0.000106=	0.000136	786	0.000014=	0.000042	912	0.000022–	0.000071	1151	0.000504=	0.000504
MaF10	5	1077	0.939567–	0.947997	1049	0.830748–	0.877855	975	0.828694–	0.879025	1180	0.967104–	0.991590
	8	853	3.240900–	3.456380	888	2.709280–	3.293280	931	2.599670–	3.326120	1284	1.498149–	1.578252
	10	855	6.178800+	5.927050	962	5.286690–	5.609290	938	4.797120–	5.624200	1475	0.689354+	0.254470
MaF11	5	968	2.437800=	2.437950	1048	2.376760=	2.370580	1020	2.361810=	2.364970	963	2.143115–	2.147966
	8	1369	9.807760–	9.847950	1105	9.448400=	9.465670	1073	9.368320=	9.367990	1027	3.395103=	3.396399
	10	1606	17.526500–	17.613100	1193	16.919800–	17.053500	1149	16.695000–	16.906100	1055	15.582983+	14.025161
MaF12	5	573	1.775930=	1.776530	558	1.671030=	1.647670	505	1.637250=	1.622540	663	1.615504–	1.668499
	8	448	7.256190=	7.254460	431	6.450900=	6.438770	462	5.856190=	5.954880	718	2.724894–	2.840152
	10	483	13.127900=	13.199400	490	11.461000=	11.612800	496	10.449200=	10.762600	750	0.000000=	0.000000
MaF13	5	820	0.225160+	0.206183	857	0.248616=	0.513728	944	0.409027–	0.515826	1815	0.204676–	0.521277
	8	872	0.234528=	0.224742	966	0.269744=	0.297863	1132	0.621752=	1.007820	2091	0.057134–	0.057623
	10	867	0.172316=	0.165359	1211	0.160839–	0.213371	1164	1.268000–	1.918400	2414	0.256875–	0.474652
Total (+/=/–) $\longrightarrow$			05/28/**15**			03/32/**13**			04/30/**14**			10/24/**14**	of 48 probs.

- θ-DEA-UIP performs statistically better in 13 (out of 48) instances, and statistically better or equivalently in 45 (out of 48) instances, compared to θ-DEA.
- MOEA/DD-UIP performs statistically better in 14 (out of 48) instances, and statistically better or equivalent in 44 (out of 48) instances, than MOEA/DD.
- LHFiD-UIP performs statistically better in 14 (out of 48) instances, and statistically better or equivalent in 38 (out of 48) instances, than LHFiD.

Overall, the UIP variants are statistically better in about 28% of the instances, and statistically better or equivalent in about 89% of the instances. This performance enhancement could be directly attributed to the search efficacy infused by the UIP operator, when integrated with an RV-EMâOA.

Towards a more comprehensive investigation of the UIP operator, its effect on an RV-EMâOA's runtime has been investigated in Section S9 of S.D. It is found that when an average SE time is 0.01 sec (or 1 sec), the overall run-time of NSGA-III-UIP is 9.5% (or 0.1%) larger than base NSGA-III to complete identical number of generations on certain problems, making the UIP approach practically viable.

VIII. Conclusions

This paper has broadened the scope of the existing pro-convergence *innovized progress* operators by simultaneously catering to the dual goals of convergence and diversity, while infusing generality and retaining practicability. Towards this end, three operators, namely, IP2$^+$, IP3 and UIP, have been

proposed. IP2$^+$ is a more generic variant of the existing pro-convergence IP2 operator, which requires fewer hard-fixed but tunable parameters while offering competitive performance. The IP3 operator is a pro-diversity operator that emphasizes equally on both aspects: spread and uniformity of the distribution. It consists of three modules, namely, training-dataset construction, ML training, and offspring creation. The final proposed UIP operator invokes either or both of IP2$^+$ and IP3 operators, for simultaneous enhancement of convergence and diversity. Some notable aspects of these operators include: (a) adaptive emphasis on convergence and/or diversity through adaptive invocations, as required by the problem being solved; and (b) practicability, since these operators successfully avoid any additional solution evaluations over and above those of the base RV-EMâOA.

The efficacy of the proposed operators has been investigated through their integration with different RV-EMâOAs, including NSGA-III, θ-DEA, MOEA/DD, and LHFiD. The above investigation over a wide range of 49 multi-objective instances established the superiority of UIP over other operators (IP2$^+$ and IP3), supporting the simultaneous enhancement of both convergence and diversity rather than catering to either of them individually. The performance of the UIP operator (best among the three) is then investigated on 48 many-objective instances, where the UIP variants of the four RV-EMâOAs are found to outperform their respective base variants.

While the scope of ML-assisted interventions has been restricted to offspring creation in this paper, their utility could also be explored in other phases of an RV-EMâOA run, including, *initialization* of solutions, *selection* operators, algorithm *termination* and *decision making*. Towards a wider gamut of ML-assisted evolutionary optimization, similar interventions in the context of EMâOAs that do not rely on using RVs may also be explored. The authors hope that the proposed operators and the results presented shall prompt more ML-based interventions in EMâOAs.

REFERENCES

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[2] C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer, 2002.

[3] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*. USA: L. Erlbaum Associates Inc., 1985, pp. 93–100.

[4] D. K. Saxena, S. Mittal, S. Kapoor, and K. Deb, "A localized high-fidelity-dominance based many-objective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

[5] S. Mittal, D. K. Saxena, and K. Deb, "Learning-based multi-objective optimization through ANN–assisted online innovization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, ser. GECCO 20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 171–172. [Online]. Available: https://doi.org/10.1145/3377929.3389925

[6] K. Deb, S. Mittal, D. K. Saxena, and E. D. Goodman, "Embedding a repair operator in evolutionary single and multi-objective algorithms - an exploitation-exploration perspective," in *Evolutionary Multi-Criterion Optimization*, H. Ishibuchi, Q. Zhang, R. Cheng, K. Li, H. Li, H. Wang, and A. Zhou, Eds. Cham: Springer International Publishing, 2021, pp. 89–101.

[7] S. Mittal, D. K. Saxena, K. Deb, and E. D. Goodman, "A learning-based *innovized* progress operator for faster convergence in evolutionary multi-objective optimization," *ACM Trans. Evol. Learn. Optim.*, vol. 2, no. 1, nov 2021. [Online]. Available: https://doi.org/10.1145/3474059

[8] ——, "Enhanced *Innovized*‘ progress operator for evolutionary multi- and many-objective optimization," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[9] M. Pelikan, D. Goldberg, and F. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5–20, 2002.

[10] L. Li, H. Chen, and C. L. et al., "A robust hybrid approach based on estimation of distribution algorithm and support vector machine for hunting candidate disease genes," *The Scientific World Journal*, vol. 2013, no. 393570, p. 7, 2013.

[11] K. Deb and C. Myburgh, "A population-based fast algorithm for a billion-dimensional resource allocation problem with integer variables," *European Journal of Operational Research*, vol. 261, no. 2, pp. 460–474, 2017.

[12] C. A. C. Coello and G. B. Lamont, *Applications of Multi-Objective Evolutionary Algorithms*. World Scientific, 2004.

[13] K. S. Bhattacharjee, A. Isaacs, and T. Ray, "Multi-objective optimization using an evolutionary algorithm embedded with multiple spatially distributed surrogates," in *Multi-Objective Optimization*. World Scientific, 2017, pp. 135–155.

[14] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 129–142, 2018.

[15] Z. Zhou, Z. Wang, T. Pang, J. Wei, and Z. Chen, "A competition-cooperation evolutionary algorithm with bidirectional multi-population local search and local hypervolume-based strategy for multi-objective optimization," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, 2021, pp. 153–160.

[16] H. G. KoÃ§er and S. A. Uymaz, "A novel local search method for LSGO with golden ratio and dynamic search step," *Soft Computing*, vol. 25, pp. 2115–2130, 2021.

[17] R. Mallipeddi and M. Lee, "Surrogate model assisted ensemble differential evolution algorithm," in *2012 IEEE Congress on Evolutionary Computation*, 2012, pp. 1–8.

[18] F. Li, L. Gao, W. Shen, X. Cai, and S. Huang, "A surrogate-assisted offspring generation method for expensive multi-objective optimization problems," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8.

[19] H. Miuhlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I. binary parameters," in *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*. London, UK, 1996, pp. 178–187.

[20] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "Boa: The bayesian optimization algorithm," in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1*, ser. GECCO'99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 525–532.

[21] Y. Du, J. qing Li, C. Luo, and L. lei Meng, "A hybrid estimation of distribution algorithm for distributed flexible job shop scheduling with crane transportations," *Swarm and Evolutionary Computation*, vol. 62, p. 100861, 2021.

[22] M. S. R. Martins, M. E. Yafrani, M. Delgado, R. Lüders, R. Santana, H. V. Siqueira, H. G. Akcay, and B. Ahiod, "Analysis of bayesian network learning techniques for a hybrid multi-objective bayesian estimation of distribution algorithm: a case study on MNK landscape," *Journal of Heuristics*, vol. 27, pp. 549–573, 2021.

[23] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, "A multiobjective evolutionary algorithm using gaussian process-based inverse modeling," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, 2015.

[24] C. He, S. Huang, R. Cheng, K. C. Tan, and Y. Jin, "Evolutionary multiobjective optimization driven by generative adversarial networks (GANs)," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3129–3142, 2021.

[25] A. Gaur and K. Deb, "Effect of size and order of variables in rules for multi-objective repair-based innovization procedure," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 2177–2184.

[26] A. Ghosh, E. Goodman, K. Deb, R. Averill, and A. Diaz, "A large-scale bi-objective optimization of solid rocket motors using innovization," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8.

[27] A. Ghosh, K. Deb, R. Averill, and E. Goodman, "Combining user knowledge and online innovization for faster solution to multi-objective design optimization problems," in *Evolutionary Multi-Criterion Optimization*,

H. Ishibuchi, Q. Zhang, R. Cheng, K. Li, H. Li, H. Wang, and A. Zhou, Eds. Cham: Springer International Publishing, 2021, pp. 102–114.
[28] T. Cover, "Estimation by the nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 14, no. 1, pp. 50–55, 1968.
[29] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, sep 1975. [Online]. Available: https://doi.org/10.1145/361002.361007
[30] N. Padhye, K. Deb, and P. Mittal, "Boundary handling approaches in particle swarm optimization," in *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012). Advances in Intelligent Systems and Computing*, J. Bansal, P. Singh, K. Deep, M. Pant, and A. Nagar, Eds., vol. 201. Springer, India, 2013, pp. 287–298.
[31] D. K. Saxena and S. Kapoor, "On timing the nadir-point estimation and/or termination of reference-based multi- and many-objective evolutionary algorithms," in *Evolutionary Multi-Criterion Optimization*, K. Deb, E. Goodman, C. A. Coello Coello, K. Klamroth, K. Miettinen, S. Mostaghim, and P. Reed, Eds. Cham: Springer International Publishing, 2019, pp. 191–202.
[32] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*. London: Springer London, 2005, pp. 105–145. [Online]. Available: https://doi.org/10.1007/1-84628-137-7_6
[33] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao, "A benchmark test suite for evolutionary many-objective optimization," *Complex & Intelligent Systems*, vol. 3, pp. 67–81, 2017.
[34] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000. [Online]. Available: https://doi.org/10.1162/106365600568202
[35] J. Lin, H. Liu, and C. Peng, "The effect of feasible region on imbalanced problem in constrained multi-objective optimization," in *2017 13th International Conference on Computational Intelligence and Security (CIS)*, 2017, pp. 82–86.
[36] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Difficulty Adjustable and Scalable Constrained Multiobjective Test Problem Toolkit," *Evolutionary Computation*, vol. 28, no. 3, pp. 339–378, 09 2020. [Online]. Available: https://doi.org/10.1162/evco_a_00259
[37] Z. Ma and Y. Wang, "Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 972–986, 2019.
[38] I. Das and J. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal of Optimization*, vol. 8, no. 3, pp. 631–657, 1998.
[39] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945. [Online]. Available: http://www.jstor.org/stable/3001968
[40] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, 1952. [Online]. Available: http://www.jstor.org/stable/2280779
[41] H. Abdi, "Bonferroni and sidak corrections for multiple comparisons," *Encyclopedia of measurement and statistics*, pp. 103–107, 2007. [Online]. Available: https://ci.nii.ac.jp/naid/20001381227/en/
[42] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.
[43] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 16–37, 2016.
[44] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.
[45] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

A Supplementary Document for "A Unified *Innovized* Progress Operator for Performance Enhancement in Evolutionary Multi- and Many-objective Optimization"

Sukrit Mittal, Dhish Kumar Saxena, Kalyanmoy Deb,and Erik D. Goodman

***Abstract*—This document presents additional analysis and results in support of the main paper. They could not be included in the main paper due to space restrictions.**

S1. Introduction to the Supplementary File

This supplementary document is structured as follows. Section S2 discusses the features of the IP3 operator's *boundary progression* procedure. Section S3 discussed the pseudo-codes used in this paper, followed by a discussion on the computational complexity of the proposed operators in Section S4. Sections S5 and S6 present the performance comparison of IP2$^+$ versus IP2 operator and IP3 operator's performance sensitivity towards a variation in parameter k (of kNN method), respectively. Section S7 presents some additional results on multi-objective instances that could not be incorporated in the main paper, followed by some insights into the UIP operator in Section S8. Finally, Section S9 presents a discussion on the overall runtime with the UIP operator.

S2. Features of IP3's Boundary Progression

In extension to Section IV-C1 of the main paper, the key features of the *boundary progression* procedure are symbolically depicted in Figure S1. In that, the projected F-space at an intermediate generation of an RV-EMâO-IP3 run is shown. This space is discretized through 5 equi-spaced RVs ($\mathcal{R}_1$–$\mathcal{R}_5$), whose associated parent solutions are marked. Since $\mathcal{R}_1$ and $\mathcal{R}_5$ are the only boundary RVs, only S_1 and S_5 can offer boundary progression. Notably, the maximum progression corresponding to $rand(0,1) = 1$, implying a scaling factor of $\lfloor\sqrt{2}/r\rceil \times r$, is depicted by the green arrow. Further, the moderate progression corresponding to an intermediate value of $rand(0,1)$, implying a scaling factor of $< \lfloor\sqrt{2}/r\rceil \times r$, is depicted by the red arrows.

This work was supported by the Ministry of Human Resource Development, Government of India, through the SPARC Scheme under Project P66. *(Corresponding author: Dhish Kumar Saxena.)*

Sukrit Mittal and Dhish Kumar Saxena are with the Department of Mechanical and Industrial Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India (e-mail: smittal1@me.iitr.ac.in; dhish.saxena@me.iitr.ac.in).

Kalyanmoy Deb and Erik D. Goodman are with the BEACON Center for the Study of Evolution in Action and the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: kdeb@egr.msu.edu; goodman@egr.msu.edu).

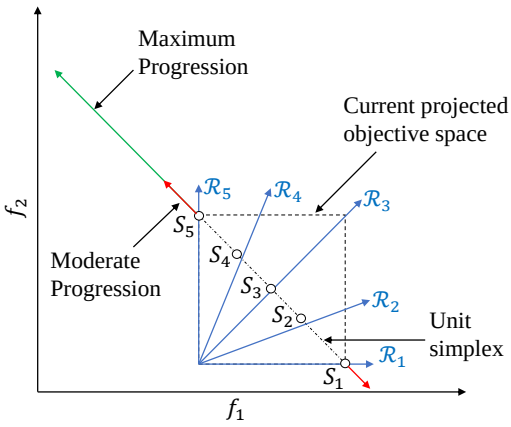


Fig. S1: A symbolic depiction of the boundary progression, during an RV-EMâO-IP3 run.

S3. Other Pseudo-codes for IP2$^+$, IP3 and UIP Operators

This section presents all related pseudo-codes that could not be provided in the main paper due to space restrictions.

A. IP2$^+$ Operator's Representation as a Function

The IP2$^+$ operator as a *function* is presented in Algorithm S1. In that, it uses some external functions such as `Archive_Mapping`, `Training` and `Progression`, details of which can be found in [1]. The underlying steps include: (a) construction of the training-dataset D_t (using `Archive_Mapping`); (b) training of the ML model on D_t (using `Training`); (c) creation of $\lfloor \mathcal{P}^{\text{IP2}^+} N \rfloor$ offspring solutions using the natural variation operators, denoted as Q_t^{V}; and (d) advancement of all the offspring solutions in Q_t^{V} (using `Progress`), leading to advanced offspring solutions $Q_t^{\text{IP2}^+}$ (sized $\lfloor \mathcal{P}^{\text{IP2}^+} N \rfloor$).

B. Frequency Adaptation in UIP Operator

As mentioned in Section V of the main paper, the UIP operator relies on the invocations of both IP2$^+$ and IP3 operators for convergence and diversity enhancement, respectively. Hence, it is imperative to adapt their invocation frequencies, $t_{\text{freq}}^{\text{IP2}^+}$ and $t_{\text{freq}}^{\text{IP3}}$, independently. However, the procedure for

Algorithm S1: $\texttt{IP2}^{+}(A_t, T_t, \mathcal{R}, [x^l, x^u], P_t, \mathcal{P}^{\text{IP2}^+})$

Input: Input archive A_t, target archive T_t, RV set $\mathcal{R}$, original variable bounds $[x^l, x^u]$, parent population P_t, proportion $\mathcal{P}^{\text{IP2}^+}$

Output: Offspring solutions $Q_t^{\text{IP2}^+}$

1 $D_t \leftarrow \texttt{Archive\_Mapping}(A_t, T_t, \mathcal{R})$
2 $ML, [x^{\min}, x^{\max}] \leftarrow \texttt{Training}(D_t, [x^l, x^u])$
3 $Q_t^{\text{V}} \leftarrow \texttt{Variation}(P_t)$
4 $Q_t^{\text{IP2}^+} \leftarrow \texttt{Progress}(Q_t^{\text{V}}, ML, [x^{\min}, x^{\max}], [x^l, x^u], \mathcal{P}^{\text{IP2}^+})$

adapting both $t_{\text{freq}}^{\text{IP2}^+}$ and $t_{\text{freq}}^{\text{IP3}}$ is exactly the same. Hence, a generic adaptation procedure has been presented for 'IPj' in Algorithm S2 (AL.S2), where j $= 2^+$ and 3 correspond to IP2$^+$ and IP3 functions, respectively. In that procedure,

- first, the offspring solutions in Q_t^{IPj} that are selected through RV-EMâOA's survival selection are counted (line 1, AL.S2). This count is denoted as $N_t^{\text{surv(IPj)}}$.
- then, the survival rate of the offspring solutions: (i) created through IPj in the current generation, S_{IPj}; and (ii) created through natural variation operators in the previous generation, S_{V}, are computed (lines 2–3, AL.S2).
- if $S_{\text{IPj}} > S_{\text{V}}$, implying a better performance of IPj operator than variation operators, $t_{\text{freq}}^{\text{IPj}}$ is reduced by 1, leading to a more frequent invocation of the IPj function.
- if $S_{\text{IPj}} < S_{\text{V}}$, implying a worse performance of IPj operator than variation operators, $t_{\text{freq}}^{\text{IPj}}$ is increased by 1, leading to a lesser frequent invocation of the IPj function.
- finally, it is ensured that $t_{\text{freq}}^{\text{IPj}}$ satisfies its minimum value of 2 (lines 8–9, AL.S2). The rationale behind this minimum value has been provided in Section V of the main paper, in the context of the risk-rewards tradeoff.

Algorithm S2: $\texttt{Adapt}(Q_t^{\text{IPj}}, P_{t+1}, Q_{t-1}^{\text{V}}, N_{t-1}^{\text{surv(V)}}, t_{\text{freq}}^{\text{IPj}})$

Input: Offspring created by IPj in current generation Q_t^{IPj}, survived population P_{t+1}, offspring created by variation operators in previous generation Q_{t-1}^{V}, number of offspring in Q_{t-1}^{V} that survived in $(t-1)^{\text{th}}$ generation $N_{t-1}^{\text{surv(V)}}$, and frequency $t_{\text{freq}}^{\text{IPj}}$

Output: $t_{\text{freq}}^{\text{IPj}}$

1 $N_t^{\text{surv(IPj)}} \leftarrow sizeof(Q_t^{\text{IPj}} \cap P_{t+1})$
2 $S_{\text{IPj}} \leftarrow N_t^{\text{surv(IPj)}} / |Q_t^{\text{IPj}}|$
3 $S_{\text{V}} \leftarrow N_{t-1}^{\text{surv(V)}} / |Q_{t-1}^{\text{V}}|$
4 **if** $S_{\text{IPj}} > S_{\text{V}}$ **then**
5 $\quad t_{\text{freq}}^{\text{IPj}} \leftarrow t_{\text{freq}}^{\text{IPj}} - 1$
6 **if** $S_{\text{IPj}} < S_{\text{V}}$ **then**
7 $\quad t_{\text{freq}}^{\text{IPj}} \leftarrow t_{\text{freq}}^{\text{IPj}} + 1$
8 **if** $t_{\text{freq}}^{\text{IPj}} < 2$ **then**
9 $\quad t_{\text{freq}}^{\text{IPj}} = 2$

C. Integration of IP3 and UIP Operators with θ-DEA

In this subsection, the integration of IP3 and UIP operators with θ-DEA [2], leading to θ-DEA-IP3 and θ-DEA-UIP, respectively, has been discussed.

Algorithm S3: Generation t of θ-DEA-IP3

Input: RV set $\mathcal{R}$, original variable bounds $[x^l, x^u]$, Parent population P_t, frequency of progression $t_{\text{freq}}^{\text{IP3}}$, last invocation $t_{\text{pg}}^{\text{IP3}}$, neighbourhood radius r, number of survived offspring in $(t-1)^{\text{th}}$ generation N_{t-1}^{surv}, proportion $\mathcal{P}^{\text{IP3}}$

Output: P_{t+1}, $t_{\text{pg}}^{\text{IP3}}$, $t_{\text{freq}}^{\text{IP3}}$, N_t^{surv}

1 $check \leftarrow \texttt{Check\_Mild\_Stabilization}()$
2 **if** *check* **then**
3 $\quad startIP3 = True$
4 **if** *startIP3 and* $t - t_{\text{pg}}^{\text{IP3}} = t_{\text{freq}}^{\text{IP3}}$ **then**
5 $\quad Q_t^{\text{IP3}} \leftarrow \texttt{IP3}(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$
6 $\quad Q_t^{\text{V}} \leftarrow \texttt{Variation}(P_t)$
7 $\quad Q_t \leftarrow Q_t^{\text{IP3}} \cup Q_t^{\text{V}}$
8 $\quad t_{\text{pg}}^{\text{IP3}} \leftarrow t$
9 **else**
10 $\quad Q_t \leftarrow \texttt{Variation}(P_t)$
11 Evaluate Q_t
12 $R_t \leftarrow P_t \cup Q_t$
13 $S_t \leftarrow \texttt{Get\_Pareto\_Nondomination\_Levels}(R_t)$
14 $\texttt{Update\_Ideal\_Point}(S_t)$
15 $\texttt{Normalize}\ (S_t, \mathbf{z}^*, \mathbf{z}^{\text{nad}})$
16 $\mathcal{C} \leftarrow \texttt{Clustering}(S_t, \mathcal{R})$
17 $\{F_1', F_2', \ldots\} \leftarrow \theta\text{-Nondominated\_Sort}(S_t, \mathcal{C})$
18 $P_{t+1} \leftarrow \emptyset$
19 $i \leftarrow 1$
20 **while** $|P_{t+1}| + |F_i'| < N$ **do**
21 $\quad P_{t+1} \leftarrow P_{t+1} \cup F_i'$
22 $\quad i \leftarrow i + 1$
23 $\texttt{Random\_Sort}\ (F_i')$
24 $P_{t+1} \leftarrow P_{t+1} \cup F_i'[N - |P_{t+1}|]$
25 $N_t^{\text{surv}} \leftarrow sizeof(Q_t \cap P_{t+1})$
26 **if** $t = t_{\text{pg}}^{\text{IP3}}$ **then**
27 $\quad$ **if** $N_t^{\text{surv}} > N_{t-1}^{\text{surv}}$ **then** $t_{\text{freq}}^{\text{IP3}} \leftarrow t_{\text{freq}}^{\text{IP3}} - 1$
28 $\quad$ **if** $N_t^{\text{surv}} < N_{t-1}^{\text{surv}}$ **then** $t_{\text{freq}}^{\text{IP3}} \leftarrow t_{\text{freq}}^{\text{IP3}} + 1$

1) θ-DEA-IP3: The algorithmic description of any generation t of θ-DEA-IP3 is summarized in Algorithm S3 (AL.S3). First, the prerequisite condition for invocation of IP3 operator, i.e., mild population stabilization, is checked (line 1, AL.S3). Towards it, a stabilization tracking algorithm has been used. If stability is detected, the $startIP3$ flag is marked as $True$ (lines 2–3, AL.S3). In subsequent generations:

- if $t_{\text{freq}}^{\text{IP3}}$ generations have passed after the last invocation of IP3 (at $t = t_{\text{pg}}^{\text{IP3}}$), then IP3 is invoked (line 4, AL.S3).
- if IP3 is invoked, $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring are created using the IP3 function, and rest $\lceil (1 - \mathcal{P}^{\text{IP3}}) N \rceil$ using the variation operators, leading to Q_t (lines 5–7, AL.S3). Otherwise, all offspring are created using the variation operators (lines 9–10, AL.S3).
- all N offspring are evaluated (line 11, AL.S3).
- the steps in lines 12–24 (AL.S3) relate to the steps of survival selection, as in base θ-DEA proposed originally [2].
- the count of offspring N_t^{surv} that survived to the next generation is estimated. In a generation where IP3 is invoked, if this count has improved compared to the previous generation, implying a good performance of the IP3 operator, then $t_{\text{freq}}^{\text{IP3}}$ is reduced by 1, yielding a more frequent progression. Otherwise, if the count has reduced, $t_{\text{freq}}^{\text{IP3}}$ is increased by 1 (lines 25–28, AL.S3).

2) θ-DEA-UIP: The algorithmic description of any generation t of θ-DEA-UIP is summarized in Algorithm S4 (AL.S4).

In that, first the target-archive T_t as required by the IP2$^+$ function is updated (line 1, AL.S4). Then the prerequisite conditions for invocations of IP2$^+$ and IP3 are checked, and if fulfilled, appropriate flags ($startIP2^+$, $startIP3$), which influence whether or not IP2$^+$ and IP3 are to be invoked, are triggered as True (lines 2–7, AL.S4). In the subsequent generations:

- if $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ generations have passed after the last invocation of IP2$^+$ (at $t = t^{\mathrm{IP2}^+}_{\mathrm{pg}}$), then IP2$^+$ is invoked and $\lfloor \mathcal{P}^{\mathrm{IP2}^+} N \rfloor$ offspring solutions are created, denoted as $Q^{\mathrm{IP2}^+}_t$ (lines 8–10, AL.S4).
- similarly, if $t^{\mathrm{IP3}}_{\mathrm{freq}}$ generations have passed after the last invocation of IP3 (at $t = t^{\mathrm{IP3}}_{\mathrm{pg}}$), then IP3 is invoked and $\lfloor \mathcal{P}^{\mathrm{IP3}} N \rfloor$ offspring solutions are created, denoted as Q^{IP3}_t (lines 11–13, AL.S4).
- if the total count of offspring created in the above two steps ($|Q^{\mathrm{IP2}^+}| + |Q^{\mathrm{IP3}}|$) is smaller than N, then the rest of the offspring are created using the natural variation operators (lines 14–15, AL.S4).
- the offspring solutions $Q^{\mathrm{IP2}^+}_t$, Q^{IP3}_t and Q^{V}_t are merged into Q_t, sized N. Then the offspring solutions Q_t are evaluated (lines 16–17, AL.S4).
- Q_t is used to update an input-archive A_{t+1}, as required by the IP2$^+$ function (line 18, AL.S4).
- the steps in lines 19–31 (AL.S4) relate to the steps of the survival selection procedure of θ-DEA [2].
- finally, $t^{\mathrm{IP2}}_{\mathrm{freq}}$ and $t^{\mathrm{IP3}}_{\mathrm{freq}}$ are adapted, if the respective operators were invoked in the current generation t (lines 32–35, AL.S4).

Algorithm S4: Generation t of θ-DEA-UIP

Input: RV set $\mathcal{R}$, variable bounds $[x^l, x^u]$, parent population P_t, offspring survived $N^{\mathrm{surv(V)}}_{t-1}$
IP2$^+$-specific: target archive T_{t-1}, input archive A_t, frequency $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$, last invocation $t^{\mathrm{IP2}^+}_{\mathrm{pg}}$, proportion $\mathcal{P}^{\mathrm{IP2}^+}$
IP3-specific: neighbourhood radius r, frequency $t^{\mathrm{IP3}}_{\mathrm{freq}}$, last invocation $t^{\mathrm{IP3}}_{\mathrm{pg}}$, proportion $\mathcal{P}^{\mathrm{IP3}}$
Output: P_{t+1}, T_t, A_{t+1}, $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$, $t^{\mathrm{IP3}}_{\mathrm{freq}}$, $t^{\mathrm{IP2}^+}_{\mathrm{pg}}$, $t^{\mathrm{IP3}}_{\mathrm{pg}}$

1 $T_t \leftarrow$ `Update_Target_Archive`$(P_t, T_{t-1}, \mathcal{R})$
2 $check1 \leftarrow$ `Check_Non_Dominated`(P_t)
3 $check2 \leftarrow$ `Check_Mild_Stabilization`()
4 **if** $check1$ **then**
5 $\quad startIP2^+ = True$
6 **if** $check2$ **then**
7 $\quad startIP3 = True$
8 **if** $startIP2^+$ *and* $t - t^{\mathrm{IP2}^+}_{\mathrm{pg}} = t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ **then**
9 $\quad Q^{\mathrm{IP2}^+}_t \leftarrow$ `IP2`$^+(A_t, T_t, \mathcal{R}, [x^l, x^u], P_t, \mathcal{P}^{\mathrm{IP2}^+})$
10 $\quad t^{\mathrm{IP2}^+}_{\mathrm{pg}} \leftarrow t$
11 **if** $startIP3$ *and* $t - t^{\mathrm{IP3}}_{\mathrm{pg}} = t^{\mathrm{IP3}}_{\mathrm{freq}}$ **then**
12 $\quad Q^{\mathrm{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\mathrm{IP3}})$
13 $\quad t^{\mathrm{IP3}}_{\mathrm{pg}} \leftarrow t$
14 **if** $|Q^{\mathrm{IP2}^+}| + |Q^{\mathrm{IP3}}| < N$ **then**
15 $\quad Q^{\mathrm{V}}_t \leftarrow$ `Variation`(P_t)
16 $Q_t \equiv Q^{\mathrm{IP2}^+}_t \cup Q^{\mathrm{IP3}}_t \cup Q^{\mathrm{V}}_t$
17 `Evaluate`(Q_t)
18 $A_{t+1} \leftarrow (A_t \cup Q_t \cup P_{t+1-t_{\mathrm{past}}}) \backslash [P_{t-t_{\mathrm{past}}} \cup Q_{t-t_{\mathrm{past}}}]$
19 $R_t \leftarrow P_t \cup Q_t$
20 $S_t \leftarrow$ `Get_Pareto_Nondomination_Levels`(R_t)
21 `Update_Ideal_Point`(S_t)
22 `Normalize` $(S_t, \mathbf{z}^*, \mathbf{z}^{\mathrm{nad}})$
23 $\mathcal{C} \leftarrow$ `Clustering`$(S_t, \mathcal{R})$
24 $\{F'_1, F'_2, \ldots\} \leftarrow \theta$-`Nondominated_Sort`$(S_t, \mathcal{C})$
25 $P_{t+1} \leftarrow \emptyset$
26 $i \leftarrow 1$
27 **while** $|P_{t+1}| + |F'_i| < N$ **do**
28 $\quad P_{t+1} \leftarrow P_{t+1} \cup F'_i$
29 $\quad i \leftarrow i + 1$
30 `Random_Sort` (F'_i)
31 $P_{t+1} \leftarrow P_{t+1} \cup F'_i[N - |P_{t+1}|]$
32 **if** $t = t^{\mathrm{IP2}^+}_{\mathrm{pg}}$ **then**
33 $\quad t^{\mathrm{IP2}^+}_{\mathrm{freq}} \leftarrow$ `Adapt`$(Q^{\mathrm{IP2}^+}_t, P_{t+1}, Q^{\mathrm{V}}_{t-1}, N^{\mathrm{surv(V)}}_{t-1}, t^{\mathrm{IP2}^+}_{\mathrm{freq}})$
34 **if** $t = t^{\mathrm{IP3}}_{\mathrm{pg}}$ **then**
35 $\quad t^{\mathrm{IP3}}_{\mathrm{freq}} \leftarrow$ `Adapt`$(Q^{\mathrm{IP3}}_t, P_{t+1}, Q^{\mathrm{V}}_{t-1}, N^{\mathrm{surv(V)}}_{t-1}, t^{\mathrm{IP3}}_{\mathrm{freq}})$

D. Integration of IP3 and UIP Operators with MOEA/DD

In this subsection, the integration of IP3 and UIP operators with MOEA/DD [3], leading to MOEA/DD-IP3 and MOEA/DD-UIP, respectively, has been discussed.

1) MOEA/DD-IP3: The algorithmic description of any generation t of MOEA/DD-IP3 is summarized in Algorithm S5 (AL.S5). First, the prerequisite condition for invocation of IP3 operator, i.e., mild population stabilization, is checked (line 1, AL.S5). Towards it, a stabilization tracking algorithm has been used. If stability is detected, the $startIP3$ flag is marked as $True$ (lines 2–3, AL.S5). In subsequent generations:

- if $t^{\mathrm{IP3}}_{\mathrm{freq}}$ generations have passed after the last invocation of IP3 (at $t = t^{\mathrm{IP3}}_{\mathrm{pg}}$), then IP3 is invoked (line 4, AL.S5).
- if IP3 is invoked, $\lfloor \mathcal{P}^{\mathrm{IP3}} N \rfloor$ offspring are created using the `IP3` function, and rest $\lceil (1 - \mathcal{P}^{\mathrm{IP3}}) N \rceil$ using the variation operators, leading to Q_t (lines 5–8, AL.S5). Otherwise, all offspring are created using the variation operators (lines 10–12, AL.S5).
- all N offspring are evaluated (line 13, AL.S5).
- the steps in lines 14–16 (AL.S5) relate to the steps of survival selection, as in base MOEA/DD proposed originally [3].
- the count of offspring N^{surv}_t that survived to the next generation is estimated. In a generation where IP3 is invoked, if this count has improved compared to the previous generation, implying a good performance of the IP3 operator, then $t^{\mathrm{IP3}}_{\mathrm{freq}}$ is reduced by 1, yielding a more frequent progression. Otherwise, if the count has reduced, $t^{\mathrm{IP3}}_{\mathrm{freq}}$ is increased by 1 (lines 17–20, AL.S5).

2) MOEA/DD-UIP: The algorithmic description of any generation t of MOEA/DD-UIP is summarized in Algorithm S6 (AL.S6). In that, first the target-archive T_t as required by the IP2$^+$ function is updated (line 1, AL.S6). Then the prerequisite conditions for invocations of IP2$^+$ and IP3 are checked, and if fulfilled, appropriate flags ($startIP2^+$, $startIP3$), which influence whether or not IP2$^+$ and IP3 are to be invoked, are triggered as True (lines 2–7, AL.S6). In the subsequent generations:

- if $t^{\mathrm{IP2}^+}_{\mathrm{freq}}$ generations have passed after the last invocation of IP2$^+$ (at $t = t^{\mathrm{IP2}^+}_{\mathrm{pg}}$), then IP2$^+$ is invoked and $\lfloor \mathcal{P}^{\mathrm{IP2}^+} N \rfloor$ offspring solutions are created, denoted as $Q^{\mathrm{IP2}^+}_t$ (lines 8–10, AL.S6).

Algorithm S5: Generation t of MOEA/DD-IP3

Input: RV set $\mathcal{R}$, original variable bounds $[x^l, x^u]$, Parent population P_t, frequency of progression $t^{\text{IP3}}_{\text{freq}}$, last invocation $t^{\text{IP3}}_{\text{pg}}$, neighbourhood radius r, number of survived offspring in $(t-1)^{\text{th}}$ generation N^{surv}_{t-1}, proportion $\mathcal{P}^{\text{IP3}}$

Output: P_{t+1}, $t^{\text{IP3}}_{\text{pg}}$, $t^{\text{IP3}}_{\text{freq}}$, N^{surv}_t

1 $check \leftarrow$ `Check_Mild_Stabilization()`
2 **if** $check$ **then**
3 $\quad startIP3 = True$
4 **if** $startIP3$ *and* $t - t^{\text{IP3}}_{\text{pg}} = t^{\text{IP3}}_{\text{freq}}$ **then**
5 $\quad Q^{\text{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$
6 $\quad \bar{P}_t \leftarrow$ `Mating_Selection`(P_t)
7 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`$(\bar{P}_t)$
8 $\quad Q_t \leftarrow Q^{\text{IP3}}_t \cup Q^{\text{V}}_t$
9 $\quad t^{\text{IP3}}_{\text{pg}} \leftarrow t$
10 **else**
11 $\quad \bar{P}_t \leftarrow$ `Mating_Selection`(P_t)
12 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`$(\bar{P}_t)$
13 `Evaluate`(Q_t)
14 $P_{t+1} \leftarrow P_t$
15 **for** $q \in Q_t$ **do**
16 $\quad P_{t+1} \leftarrow$ `Update_Population`(P_{t+1}, q)
17 $N^{\text{surv}}_t \leftarrow sizeof(Q_t \cap P_{t+1})$
18 **if** $t = t^{\text{IP3}}_{\text{pg}}$ **then**
19 $\quad$ **if** $N^{\text{surv}}_t > N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} - 1$
20 $\quad$ **if** $N^{\text{surv}}_t < N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} + 1$

- similarly, if $t^{\text{IP3}}_{\text{freq}}$ generations have passed after the last invocation of IP3 (at $t = t^{\text{IP3}}_{\text{pg}}$), then IP3 is invoked and $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring solutions are created, denoted as Q^{IP3}_t (lines 11–13, AL.S6).
- if the total count of offspring created in the above two steps ($|Q^{\text{IP2}^+}| + |Q^{\text{IP3}}|$) is smaller than N, then the rest of the offspring are created using the natural variation operators (lines 14–16, AL.S6).
- the offspring solutions $Q^{\text{IP2}^+}_t$, Q^{IP3}_t and Q^{V}_t are merged into Q_t, sized N. Then the offspring solutions Q_t are evaluated (lines 17–18, AL.S6).
- Q_t is used to update an input-archive A_{t+1}, as required by the `IP2`$^+$ function (line 19, AL.S6).
- the steps in lines 20–22 (AL.S6) relate to the steps of the survival selection procedure of MOEA/DD [3].
- finally, $t^{\text{IP2}}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ are adapted, if the respective operators were invoked in the current generation t (lines 23–26, AL.S6).

E. Integration of IP3 and UIP operators with LHFiD

In this subsection, the integration of IP3 and UIP operators with LHFiD [4], leading to LHFiD-IP3 and LHFiD-UIP, respectively, has been discussed. Notably, LHFiD, as originally proposed in [4], does not include constraint handling. Owing to the use of constrained problems in this paper:

- its random mating selection procedure is replaced with the constraint-based mating selection procedure [5].
- its survival selection procedure is modified to select feasible solutions first. If the number of feasible solutions is less than N, then the infeasible solutions offering the least constraint violation values are selected to fill the population [5].

Algorithm S6: Generation t of MOEA/DD-UIP

Input: RV set $\mathcal{R}$, variable bounds $[x^l, x^u]$, parent population P_t, offspring survived $N^{\text{surv(V)}}_{t-1}$
IP2$^+$**-specific:** target archive T_{t-1}, input archive A_t, frequency $t^{\text{IP2}^+}_{\text{freq}}$, last invocation $t^{\text{IP2}^+}_{\text{pg}}$, proportion $\mathcal{P}^{\text{IP2}^+}$
IP3-specific: neighbourhood radius r, frequency $t^{\text{IP3}}_{\text{freq}}$, last invocation $t^{\text{IP3}}_{\text{pg}}$, proportion $\mathcal{P}^{\text{IP3}}$

Output: P_{t+1}, T_t, A_{t+1}, $t^{\text{IP2}^+}_{\text{freq}}$, $t^{\text{IP3}}_{\text{freq}}$, $t^{\text{IP2}^+}_{\text{pg}}$, $t^{\text{IP3}}_{\text{pg}}$

1 $T_t \leftarrow$ `Update_Target_Archive`$(P_t, T_{t-1}, \mathcal{R})$
2 $check1 \leftarrow$ `Check_Non_Dominated`(P_t)
3 $check2 \leftarrow$ `Check_Mild_Stabilization()`
4 **if** $check1$ **then**
5 $\quad startIP2^+ = True$
6 **if** $check2$ **then**
7 $\quad startIP3 = True$
8 **if** $startIP2^+$ *and* $t - t^{\text{IP2}^+}_{\text{pg}} = t^{\text{IP2}^+}_{\text{freq}}$ **then**
9 $\quad Q^{\text{IP2}^+}_t \leftarrow$ `IP2`$^+(A_t, T_t, \mathcal{R}, [x^l, x^u], P_t, \mathcal{P}^{\text{IP2}^+})$
10 $\quad t^{\text{IP2}^+}_{\text{pg}} \leftarrow t$
11 **if** $startIP3$ *and* $t - t^{\text{IP3}}_{\text{pg}} = t^{\text{IP3}}_{\text{freq}}$ **then**
12 $\quad Q^{\text{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$
13 $\quad t^{\text{IP3}}_{\text{pg}} \leftarrow t$
14 **if** $|Q^{\text{IP2}^+}| + |Q^{\text{IP3}}| < N$ **then**
15 $\quad \bar{P}_t \leftarrow$ `Mating_Selection`(P_t)
16 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`$(\bar{P}_t)$
17 $Q_t \equiv Q^{\text{IP2}^+}_t \cup Q^{\text{IP3}}_t \cup Q^{\text{V}}_t$
18 `Evaluate`(Q_t)
19 $A_{t+1} \leftarrow (A_t \cup Q_t \cup P_{t+1-t_{\text{past}}}) \backslash [P_{t-t_{\text{past}}} \cup Q_{t-t_{\text{past}}}]$
20 $P_{t+1} \leftarrow P_t$
21 **for** $q \in Q_t$ **do**
22 $\quad P_{t+1} \leftarrow$ `Update_Population`(P_{t+1}, q)
23 **if** $t = t^{\text{IP2}^+}_{\text{pg}}$ **then**
24 $\quad t^{\text{IP2}^+}_{\text{freq}} \leftarrow$ `Adapt`$(Q^{\text{IP2}^+}_t, P_{t+1}, Q^{\text{V}}_{t-1}, N^{\text{surv(V)}}_{t-1}, t^{\text{IP2}^+}_{\text{freq}})$
25 **if** $t = t^{\text{IP3}}_{\text{pg}}$ **then**
26 $\quad t^{\text{IP3}}_{\text{freq}} \leftarrow$ `Adapt`$(Q^{\text{IP3}}_t, P_{t+1}, Q^{\text{V}}_{t-1}, N^{\text{surv(V)}}_{t-1}, t^{\text{IP3}}_{\text{freq}})$

Further, there is a minor modification with the first invocation criterion of the IP2$^+$ operator when integrated with LHFiD. Instead of ensuring that the entire population is non-dominated, the non-domination is checked *locally* within each cluster (one cluster per RV). Since LHFiD (originally) does not have a non-domination check over the entire population, there is no inherent selection pressure for the entire population to be non-dominated. Hence, without the above modification, it is plausible that the IP2$^+$ may never be invoked over the whole of LHFiD-UIP run.

1) LHFiD-IP3: The algorithmic description of any generation t of LHFiD-IP3 is summarized in Algorithm S7 (AL.S7). First, the prerequisite condition for invocation of IP3 operator, i.e., mild population stabilization, is checked (line 1, AL.S7). Towards it, a stabilization tracking algorithm has been used. If stability is detected, the $startIP3$ flag is marked as $True$ (lines 2–3, AL.S7). In subsequent generations:

- if $t^{\text{IP3}}_{\text{freq}}$ generations have passed after the last invocation of IP3 (at $t = t^{\text{IP3}}_{\text{pg}}$), then IP3 is invoked (line 4, AL.S7).
- if IP3 is invoked, $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring are created using the `IP3` function, and rest $\lceil (1 - \mathcal{P}^{\text{IP3}}) N \rceil$ using the variation operators, leading to Q_t (lines 5–7, AL.S7). Otherwise, all offspring are created using the variation

Algorithm S7: Generation t of LHFiD-IP3

Input: RV set $\mathcal{R}$, original variable bounds $[x^l, x^u]$, Parent population P_t, frequency of progression $t^{\text{IP3}}_{\text{freq}}$, last invocation $t^{\text{IP3}}_{\text{pg}}$, neighbourhood radius r, number of survived offspring in $(t-1)^{\text{th}}$ generation N^{surv}_{t-1}, proportion $\mathcal{P}^{\text{IP3}}$

Output: P_{t+1}, $t^{\text{IP3}}_{\text{pg}}$, $t^{\text{IP3}}_{\text{freq}}$, N^{surv}_t

1 $check \leftarrow$ `Check_Mild_Stabilization()`
2 **if** $check$ **then**
3 $\quad startIP3 = True$
4 **if** $startIP3$ *and* $t - t^{\text{IP3}}_{\text{pg}} = t^{\text{IP3}}_{\text{freq}}$ **then**
5 $\quad Q^{\text{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$
6 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`(P_t)
7 $\quad Q_t \leftarrow Q^{\text{IP3}}_t \cup Q^{\text{V}}_t$
8 $\quad t^{\text{IP3}}_{\text{pg}} \leftarrow t$
9 **else**
10 $\quad Q_t \leftarrow$ `Variation`(P_t)
11 Evaluate Q_t
12 $U_t \leftarrow P_t \cup Q_t$
13 $Z^I \leftarrow \min(Z^I_k, \min_{x \in U_t} f_k(x)) \; \forall \; k \in [1, M]$
14 $F \leftarrow$ All objective vectors in U_t
15 **if** $Z^N \neq \emptyset$ **then**
16 $\quad \tilde{F} \leftarrow$ Normalized F using Z^I & Z^N
17 **else**
18 $\quad \tilde{F} \leftarrow F - Z^I$
19 $P_{t+1} \leftarrow$ `Survival_selection`$(U_t, \tilde{F}, N, \mathcal{R})$
20 $C^N \leftarrow$ `Stabilization_Tracking`$(P_t, P_{t+1}, \mathcal{R}, \psi_{\text{mild}})$
21 **if** C^N ***is True*** **then**
22 $\quad Z^N \leftarrow$ `Update_Nadir`(P_{t+1}, Z^I)
23 $N^{\text{surv}}_t \leftarrow sizeof(Q_t \cap P_{t+1})$
24 **if** $t = t^{\text{IP3}}_{\text{pg}}$ **then**
25 $\quad$ **if** $N^{\text{surv}}_t > N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} - 1$
26 $\quad$ **if** $N^{\text{surv}}_t < N^{\text{surv}}_{t-1}$ **then** $t^{\text{IP3}}_{\text{freq}} \leftarrow t^{\text{IP3}}_{\text{freq}} + 1$

Algorithm S8: Generation t of LHFiD-UIP

Input: RV set $\mathcal{R}$, variable bounds $[x^l, x^u]$, parent population P_t, offspring survived $N^{\text{surv(V)}}_{t-1}$
IP2$^+$-specific: target archive T_{t-1}, input archive A_t, frequency $t^{\text{IP2}^+}_{\text{freq}}$, last invocation $t^{\text{IP2}^+}_{\text{pg}}$, proportion $\mathcal{P}^{\text{IP2}^+}$
IP3-specific: neighbourhood radius r, frequency $t^{\text{IP3}}_{\text{freq}}$, last invocation $t^{\text{IP3}}_{\text{pg}}$, proportion $\mathcal{P}^{\text{IP3}}$

Output: P_{t+1}, T_t, A_{t+1}, $t^{\text{IP2}^+}_{\text{freq}}$, $t^{\text{IP3}}_{\text{freq}}$, $t^{\text{IP2}^+}_{\text{pg}}$, $t^{\text{IP3}}_{\text{pg}}$

1 $T_t \leftarrow$ `Update_Target_Archive`$(P_t, T_{t-1}, \mathcal{R})$
2 $check1 \leftarrow$ `Check_Non_Dominated`(P_t)
3 $check2 \leftarrow$ `Check_Mild_Stabilization()`
4 **if** $check1$ **then**
5 $\quad startIP2^+ = True$
6 **if** $check2$ **then**
7 $\quad startIP3 = True$
8 **if** $startIP2^+$ *and* $t - t^{\text{IP2}^+}_{\text{pg}} = t^{\text{IP2}^+}_{\text{freq}}$ **then**
9 $\quad Q^{\text{IP2}^+}_t \leftarrow$ `IP2`$^+(A_t, T_t, \mathcal{R}, [x^l, x^u], P_t, \mathcal{P}^{\text{IP2}^+})$
10 $\quad t^{\text{IP2}^+}_{\text{pg}} \leftarrow t$
11 **if** $startIP3$ *and* $t - t^{\text{IP3}}_{\text{pg}} = t^{\text{IP3}}_{\text{freq}}$ **then**
12 $\quad Q^{\text{IP3}}_t \leftarrow$ `IP3`$(P_t, \mathcal{R}, r, [x^l, x^u], \mathcal{P}^{\text{IP3}})$
13 $\quad t^{\text{IP3}}_{\text{pg}} \leftarrow t$
14 **if** $|Q^{\text{IP2}^+}| + |Q^{\text{IP3}}| < N$ **then**
15 $\quad Q^{\text{V}}_t \leftarrow$ `Variation`(P_t)
16 $Q_t \equiv Q^{\text{IP2}^+}_t \cup Q^{\text{IP3}}_t \cup Q^{\text{V}}_t$
17 Evaluate Q_t
18 $A_{t+1} \leftarrow (A_t \cup Q_t \cup P_{t+1-t_{\text{past}}}) \backslash [P_{t-t_{\text{past}}} \cup Q_{t-t_{\text{past}}}]$
19 $U_t \leftarrow P_t \cup Q_t$
20 $Z^I \leftarrow \min(Z^I_k, \min_{x \in U_t} f_k(x)) \; \forall \; k \in [1, M]$
21 $F \leftarrow$ All objective vectors in U_t
22 **if** $Z^N \neq \emptyset$ **then**
23 $\quad \tilde{F} \leftarrow$ Normalized F using Z^I & Z^N
24 **else**
25 $\quad \tilde{F} \leftarrow F - Z^I$
26 $P_{t+1} \leftarrow$ `Survival_selection`$(U_t, \tilde{F}, N, \mathcal{R})$
27 $C^N \leftarrow$ `Stabilization_Tracking`$(P_t, P_{t+1}, \mathcal{R}, \psi_{\text{mild}})$
28 **if** C^N ***is True*** **then**
29 $\quad Z^N \leftarrow$ `Update_Nadir`(P_{t+1}, Z^I)
30 **if** $t = t^{\text{IP2}^+}_{\text{pg}}$ **then**
31 $\quad t^{\text{IP2}^+}_{\text{freq}} \leftarrow$ `Adapt`$(Q^{\text{IP2}^+}_t, P_{t+1}, Q^{\text{V}}_{t-1}, N^{\text{surv(V)}}_{t-1}, t^{\text{IP2}^+}_{\text{freq}})$
32 **if** $t = t^{\text{IP3}}_{\text{pg}}$ **then**
33 $\quad t^{\text{IP3}}_{\text{freq}} \leftarrow$ `Adapt`$(Q^{\text{IP3}}_t, P_{t+1}, Q^{\text{V}}_{t-1}, N^{\text{surv(V)}}_{t-1}, t^{\text{IP3}}_{\text{freq}})$

operators (lines 9–10, AL.S7).

- all N offspring are evaluated (line 11, AL.S7).
- the steps in lines 12–19 (AL.S7) relate to the steps of survival selection, as in proposed originally LHFiD [4].
- if mild stabilization of the population is detected, then nadir point is updated (lines 20–22, AL.S7).
- the count of offspring N^{surv}_t that survived to the next generation is estimated. In a generation where IP3 is invoked, if this count has improved compared to the previous generation, implying a good performance of the IP3 operator, then $t^{\text{IP3}}_{\text{freq}}$ is reduced by 1, yielding a more frequent progression. Otherwise, if the count has reduced, $t^{\text{IP3}}_{\text{freq}}$ is increased by 1 (lines 23–26, AL.S7).

2) LHFiD-UIP: The algorithmic description of any generation t of LHFiD-UIP is summarized in Algorithm S8 (AL.S8). In that, first the target-archive T_t as required by the IP2$^+$ function is updated (line 1, AL.S8). Then the prerequisite conditions for invocations of IP2$^+$ and IP3 are checked, and if fulfilled, appropriate flags ($startIP2^+$, $startIP3$), which influence whether or not IP2$^+$ and IP3 are to be invoked, are triggered as True (lines 2–7, AL.S8). In the subsequent generations:

- if $t^{\text{IP2}^+}_{\text{freq}}$ generations have passed after the last invocation of IP2$^+$ (at $t = t^{\text{IP2}^+}_{\text{pg}}$), then IP2$^+$ is invoked and $\lfloor \mathcal{P}^{\text{IP2}^+} N \rfloor$ offspring solutions are created, denoted as $Q^{\text{IP2}^+}_t$ (lines 8–10, AL.S8).
- similarly, if $t^{\text{IP3}}_{\text{freq}}$ generations have passed after the last invocation of IP3 (at $t = t^{\text{IP3}}_{\text{pg}}$), then IP3 is invoked and $\lfloor \mathcal{P}^{\text{IP3}} N \rfloor$ offspring solutions are created, denoted as Q^{IP3}_t (lines 11–13, AL.S8).
- if the total count of offspring created in the above two steps ($|Q^{\text{IP2}^+}| + |Q^{\text{IP3}}|$) is smaller than N, then the rest of the offspring are created using the natural variation operators (lines 14–15, AL.S8).
- the offspring solutions $Q^{\text{IP2}^+}_t$, Q^{IP3}_t and Q^{V}_t are merged into Q_t, sized N. Then the offspring solutions Q_t are evaluated (lines 16–17, AL.S8).
- Q_t is used to update an input-archive A_{t+1}, as required by the IP2$^+$ function (line 18, AL.S8).
- the steps in lines 19–26 (AL.S8) relate to the steps of survival selection, as in base LHFiD proposed originally.
- if mild stabilization of the population is detected, then

nadir point is updated (lines 27–29, AL.S8).
- finally, $t_{\text{freq}}^{\text{IP2}^+}$ and $t_{\text{freq}}^{\text{IP3}}$ are adapted, if the respective operators were invoked in the current generation t (lines 30–33, AL.S8).

S4. Computational Complexity

This section discusses the time and space complexities of the proposed IP2$^+$, IP3 and the UIP operators. In that, first IP2$^+$ and IP3 operators are discussed in the upcoming subsections, followed by the UIP operator.

A. IP2$^+$ Operator

Since the underlying mechanism and algorithm for training-dataset construction, ML training, and offspring progression are precisely the same for IP2 and IP2$^+$ operators, their computational complexities are also the same. The module-wise time and space complexities of the IP2$^+$ operator are provided in Table S1, as earlier derived in [1].

TABLE S1: Time- and space-complexities of different modules of the IP2$^+$ operator

Module	Time-complexity	Space-complexity
Dataset construction	$\mathcal{O}(MN^2t_{\text{past}})$	$\mathcal{O}(Nt_{\text{past}})$
ML training	$\mathcal{O}(N^3t_{\text{past}}^3n_{\text{var}}\log(Nt_{\text{past}}))$	$\mathcal{O}(N^2t_{\text{past}}^2n_{\text{var}})$
Offspring progression	$\mathcal{O}(N^3t_{\text{past}}^2n_{\text{var}})$	–

B. IP3 Operator

As detailed in Section IV of the main paper, the proposed IP3 operator comprises three modules. In the following subsubsections, each constituent module's time and space complexities have been discussed, followed by their overall summary.

1) Training-dataset Construction: The process of constructing M training-datsets $\mathcal{D}_1$–$\mathcal{D}_M$ has been summarized in Algorithm 1. In that, for each solution of P_t, first the *target* solutions cluster is identified which requires N computations and then, the *target* solutions are picked for each dataset, which requires $M \times M$ computations. Collectively, this procedure is repeated for each solution in P_t (sized N). Given that the above procedure is repeated for N solutions, the resulting time complexity is $\max\{\mathcal{O}(N^2), \mathcal{O}(NM^2))\}$. Generally, the population sizes N used in RV-EMâOAs (as also used in this paper) satisfy $N \geq M^2$. Hence, the resulting time complexity of this module can be approximated as $\mathcal{O}(N^2)$.

Further, only the training-datasets $\mathcal{D}_1$–$\mathcal{D}_M$ are additionally created, over base RV-EMâOA. Each dataset in $\mathcal{D}$ can have a maximum of N mapped pairs of solutions, considering which the resulting space complexity of this module is $\mathcal{O}(MN)$.

2) ML Training: The training-datasets constructed in the previous module are used to train M kNN regressor models in this module. The worst case time complexity of training a kNN model is $\mathcal{O}(N_{\text{dim}}N_{\text{sam}}\log(N_{\text{sam}}))$, where N_{dim} denotes the dimensionality of the training-dataset and N_{sam} denotes the number of samples. Similarly, the worst case space complexity of the kNN is $\mathcal{O}(N_{\text{dim}}N_{\text{sam}})$. In this case, $N_{\text{dim}} = n_{\text{var}}$, $\max(N_{\text{sam}}) = N$, and M trainings are executed. Upon substituting their values, the resulting time and space complexities of the ML training module are $\mathcal{O}(MNn_{\text{var}}\log(N))$ and $\mathcal{O}(MNn_{\text{var}})$, respectively.

3) Offspring Creation: Evidently, this module constitutes two submodules, each following the same procedure from the computational complexity perspective. Hence, a common discussion is presented here. Each submodule is executed through four steps: (a) identification of the solution to be advanced; (b) selection of the ML model; (c) identification of the search direction; and (d) advancement and repair. The worst-case computational complexity of step-(a) is $\mathcal{O}(MN)$, owing to the identification of the nearest solution to a given RV. Step-(b) requires $M + M$ computations, leading to a complexity of $\mathcal{O}(M)$. Step-(c) involves making a prediction using one of the learnt kNN models. The prediction time complexity of a kNN model is $\mathcal{O}(kN_{\text{sam}})$, where k is the number of neighbours. Since $k = n_{\text{var}}$ has been used, the resulting time complexity of making a prediction is $\mathcal{O}(Nn_{\text{var}})$. Finally, step-(d) involves the advancement of a given solution with $\mathcal{O}(M)$ complexity. Amongst the four steps, the worst time complexity can be given as $\max\{\mathcal{O}(MN), \mathcal{O}(Nn_{\text{var}})\}$. Generally, $n_{\text{var}} > M$ is majority of the MOPs. Hence, the worst-case time complexity can be approximated as $\mathcal{O}(Nn_{\text{var}})$. Since, these steps are repeated for $\lfloor \mathcal{P}^{\text{IP3}}N \rfloor$ solutions, the resulting time complexity becomes $\mathcal{O}(N^2n_{\text{var}})$. Moreover, since the offspring solutions created through advancement are included in the N offspring solutions created by any RV-EMâOA, there is no related space complexity of the offspring creation module.

The worst-case time complexity and space complexity of each constituent module of the IP3 operator is summarized in Table S2. Evidently, the training of M kNN models has the highest time complexity, and the M trained kNN models have the highest space complexity.

TABLE S2: Time- and space-complexities of different modules of the IP3 operator

Module	Time-complexity	Space-complexity
Dataset construction	$\mathcal{O}(N^2)$	$\mathcal{O}(MN)$
ML training	$\mathcal{O}(MNn_{\text{var}}\log(N))$	$\mathcal{O}(MNn_{\text{var}})$
Offspring creation	$\mathcal{O}(N^2n_{\text{var}})$	–

C. UIP Operator

As detailed in Section V of the main paper, the proposed UIP operator comprises two modules. These modules are based on the IP2$^+$ and IP3 operators, whose computational complexity analysis has been presented earlier in this section. From that analysis, the worst-case time and space complexities of IP2$^+$ based offspring advancement and IP3 based offspring creation are summarized in Table S3.

TABLE S3: Time- and space-complexities of different modules of the UIP operator

Module	Time-complexity	Space-complexity
IP2$^+$	$\mathcal{O}(N^3t_{\text{past}}^3n_{\text{var}}\log(Nt_{\text{past}}))$	$\mathcal{O}(N^2t_{\text{past}}^2n_{\text{var}})$
IP3	$\mathcal{O}(MNn_{\text{var}}\log(N))$	$\mathcal{O}(MNn_{\text{var}})$

It may be noted that these worst-case complexities, for both IP2$^+$ and IP3, correspond to their underlying ML methods,

i.e., RF and kNN, respectively. Using a different ML method may affect the corresponding complexities in Table S3. However, since the focus of this paper is to provide a proof-of-concept that ML methods could be utilized for such enhancements relating to convergence and diversity in RV-EMâOAs, the choice of ML method has not been incorporated within the scope of this paper.

S5. Performance Comparison: IP2$^+$ versus IP2

As can be observed in Sections III and VI-C2 of the main paper, there are some differences between IP2$^+$ and IP2 operators, as summarized below.

- In an RV-EMâO-IP2$^+$ run, IP2$^+$ is first invoked only when the population becomes entirely non-dominated. In contrast, in an RV-EMâO-IP2 run, IP2 is first invoked as soon as t_{past} generations have passed.
- In IP2$^+$, $t_{\text{freq}}^{\text{IP2}^+}$ is adapted on-the-fly, whereas in IP2, $t_{\text{freq}}^{\text{IP2}}$ is fixed at 5.
- In IP2$^+$, η_j assumes a random value between 1 and 1.5, whereas in IP2, η_j is fixed at 1.1.

While the above changes help reduce the number of user-defined parameters to be fixed a priori, it is imperative to investigate if IP2$^+$ can maintain the same performance as IP2. In this background, the median hypervolume obtained by NSGA-III-IP2 and NSGA-III-IP2$^+$ at the end of t_{TM} generations (determined on-the-fly for NSGA-III-IP2$^+$) are shown in Table S4. The best-obtained hypervolume and its statistically equivalent results are marked in bold. From Table S4, it is evident that NSGA-III-IP2$^+$ performs statistically better than or equivalent to NSGA-III-IP2 in 20 (out of 21) instances. Hence, it is fair to infer that the changes proposed in IP2, leading to IP2$^+$, did not deteriorate its performance while making it more generic through a reduction in user-defined parameters.

TABLE S4: Hypervolume based comparison of NSGA-III-IP2$^+$ and NSGA-III-IP2. Here, t_{TM} determined on-the-fly for NSGA-III-IP2$^+$, and the same has been used for NSGA-III-IP2. The best performing algorithm and the statistically equivalent algorithms are marked in bold.

	Problem	t_{TM}	NSGA-III-IP2	NSGA-III-IP2$^+$	p-value
$M=2$	ŽDT1	1197	0.681859	**0.681860**	1.46E-02
	ŽDT2	1280	**0.348794**	**0.348794**	6.17E-01
	ŽDT3	1005	**1.068475**	**1.068537**	3.79E-01
	ŽDT4	1768	**0.681859**	**0.681860**	8.82E-01
	ŽDT6	1808	**0.322988**	**0.319672**	5.12E-02
$M=3$	DTLZ1	1408	**1.222497**	**1.221979**	9.81E-02
	DTLZ2	970	**0.667316**	**0.667333**	7.73E-02
	DTLZ3	1658	**0.658056**	**0.660617**	6.07E-01
	DTLZ4	1509	**0.667341**	**0.667316**	9.61E-01
	MaF1	603	**0.234705**	**0.234557**	7.62E-01
	MaF2	500	**0.397191**	0.396523	3.18E-03
	MaF3	2078	1.192407	**1.193830**	4.34E-02
	MaF4	1315	0.625329	**0.627573**	4.95E-02
	MaF5	1344	**1.227595**	**1.227601**	9.94E-01
	MaF7	1215	**0.375744**	**0.376036**	3.56E-01
	MaF8	1510	**0.464411**	**0.463852**	4.26E-01
	MaF9	1326	**0.626816**	**0.626816**	9.94E-01
	MaF10	982	**0.512130**	**0.516973**	7.49E-02
	MaF11	965	0.975358	**0.979677**	4.36E-03
	MaF12	737	**0.614130**	**0.614154**	6.07E-01
	MaF13	934	**0.370384**	**0.372636**	1.37E-01
	Total $\longrightarrow$		17	**20**	of 21 probs.

Notably, the above investigation has been restricted to only some of the test suites (ZDT [6], DTLZ [7] and MaF [8]), which are the same as used in the original study in which IP2 was proposed [1].

S6. IP3 Operator's Performance Sensitivity towards Variation in k (kNN)

As discussed earlier in Section IV-B, setting the number of nearest neighbours k (in kNN) is essential since keeping it very low or very high may lead to underfitting or overfitting, respectively. Although the choice of $k = n_{\text{var}}$ has been reasoned, it is imperative to analyze the IP3 operator's performance sensitivity towards the variation in k. In this background, the performance of NSGA-III-IP3 is presented here with three different settings of k, on some test problems[1]. These include: (a) $k = 0.5n_{\text{var}}$, leading to a value lower than the proposed setting; (b) $k = n_{\text{var}}$, the proposed setting; and (c) $k = 1.5n_{\text{var}}$, leading to a value higher than the proposed setting. The median hypervolume obtained by NSGA-III-IP3 with all three settings of k, at t_{TM} generations determined on-the-fly for $k = n_{\text{var}}$, are shown in Table S5. The best-obtained hypervolume and its statistically equivalent results are marked in bold. From Table S5, following may be noted.

TABLE S5: Hypervolume based comparison of NSGA-III-IP3, across different settings of k (used in the ML method). Here, t_{TM} determined on-the-fly for NSGA-III-IP3 with $k = n_{\text{var}}$, has been used for other values of k. The best performing algorithm and the statistically equivalent algorithms are marked in bold.

	Problem	t_{TM}	$k = 0.5n_{\text{var}}$	$k = n_{\text{var}}$	$k = 1.5n_{\text{var}}$
$M=2$	CIBN1	1404	0.461365	**0.483381**	**0.480270**
	CIBN2	753	0.668817	**0.669094**	**0.670272**
	CIBN3	889	0.218083	**0.219149**	**0.219981**
	DASCMOP1	1948	0.092495	**0.316990**	**0.310258**
	DASCMOP2	1793	0.423645	**0.637702**	**0.643858**
	DASCMOP3	1639	**0.391166**	**0.394160**	**0.421778**
	DASCMOP4	1978	**0.336946**	**0.336812**	**0.336798**
	DASCMOP5	2101	**0.672619**	**0.672677**	**0.672666**
	DASCMOP6	2377	**0.574846**	**0.574818**	**0.571492**
	MW1	1047	**0.415397**	**0.415318**	**0.415296**
	MW2	836	**0.483818**	**0.482899**	**0.482363**
	MW3	875	**0.470024**	**0.469803**	**0.454545**
	MW5	1821	**0.196080**	**0.197173**	**0.190051**
	MW6	1229	**0.298365**	**0.298348**	**0.287468**
	MW7	892	**0.366522**	**0.366413**	**0.366388**
	MW9	1071	**0.295749**	**0.296410**	**0.295482**
	MW10	1063	**0.247155**	**0.247365**	**0.199358**
	MW11	961	**0.266061**	**0.259526**	**0.243113**
	MW12	1068	**0.570748**	**0.570814**	**0.570793**
	MW13	972	**0.328788**	**0.328191**	**0.326313**
$M=3$	CIBN4	438	**0.921200**	**0.917063**	**0.926683**
	CIBN5	287	**0.629971**	**0.629746**	**0.629372**
	DASCMOP7	1693	**1.022840**	**1.025306**	**1.016004**
	DASCMOP8	1650	**0.658195**	**0.658097**	**0.649069**
	DASCMOP9	1539	**0.647740**	**0.647012**	**0.646312**
	MW4	743	**1.041465**	**1.041362**	**1.041295**
	MW8	718	**0.626492**	**0.626362**	**0.619107**
	MW14	914	0.151894	**0.158839**	0.153753
	Total $\longrightarrow$		22	**28**	27

[1]Only diversity-hard problems (CIBN [9], DASCMOP [10] and MW [11]) have been considered here since they would inherently lead to more frequent invocations of IP3 operator, across the generations of NSGA-III-IP3 run, and would consequently offer a better sensitivity analysis.

TABLE S6: HV based comparison of IP2$^+$, IP3 and UIP operators, performed for LHFiD. Each column shows the median HV (from 31 runs) at the end of t_{TM} generations, determined on-the-fly for LHFiD-UIP. The symbols "–", "=" or "+" against each algorithm highlight where these are statistically worse than, comparable to, or better than LHFiD-UIP, respectively.

Problem	M	t_{TM}	LHFiD	LHFiD-IP2$^+$	LHFiD-IP3	LHFiD-UIP
CIBN1	2	518	0.381507–	0.387450–	0.463808–	0.506510
CIBN2	2	406	0.670678–	0.671273–	0.675471–	0.677235
CIBN3	2	426	0.213834–	0.215407–	0.211442–	0.222645
CIBN4	3	393	0.992818–	0.974488–	1.028136=	1.028556
CIBN5	3	359	0.622345=	0.623388=	0.619048–	0.622445
DASCMOP1	2	1929	0.082801–	0.087463–	0.317174–	0.329137
DASCMOP2	2	1876	0.410152–	0.407133–	0.644410–	0.663136
DASCMOP3	2	2973	0.402701=	0.392024=	0.392024=	0.392024
DASCMOP4	2	1741	0.267030=	0.277443=	0.270313=	0.250359
DASCMOP5	2	1863	0.646158=	0.631167=	0.635844=	0.629977
DASCMOP6	2	1887	0.549128=	0.536883=	0.534751=	0.532582
DASCMOP7	3	2257	1.028625=	1.028427=	1.028650=	1.028417
DASCMOP8	3	2115	0.655370=	0.656146=	0.656517=	0.655050
DASCMOP9	3	2201	0.641902–	0.635743–	0.643118=	0.642864
DTLZ1	3	633	0.012268=	0.005491=	0.000241=	0.002838
DTLZ2	3	499	0.659976=	0.660828=	0.660489=	0.660592
DTLZ3	3	635	0.000000=	0.000000=	0.000000=	0.000000
DTLZ4	3	908	0.661133=	0.661332=	0.660907=	0.661058
MaF1	3	546	0.208156=	0.208482=	0.208400=	0.209422
MaF2	3	488	0.394240=	0.393779=	0.393962=	0.394093
MaF3	3	1527	1.191826=	1.193666=	1.193629=	1.194385
MaF4	3	736	0.568126=	0.573084=	0.578781=	0.013983
MaF5	3	1119	1.222987–	1.228259=	1.228095=	1.228257
MaF7	3	1049	0.363208–	0.371419=	0.371198=	0.371374
MaF8	3	1552	0.458162=	0.457597=	0.457319=	0.458045
MaF9	3	777	1.160416=	1.160343=	1.160292=	1.160459
MaF10	3	944	0.568694=	0.573637=	0.573139=	0.571757
MaF11	3	800	1.146045–	1.159227=	1.157304=	1.159070
MaF12	3	626	0.533782–	0.594185=	0.542982=	0.543327
MaF13	3	946	0.401748=	0.388114=	0.444120=	0.456261
MW1	2	1268	0.415326+	0.414756=	0.415184=	0.415005
MW2	2	813	0.327672=	0.354998=	0.355033=	0.330055
MW3	2	1619	0.417336=	0.000000=	0.000000=	0.000000
MW4	3	879	1.039593=	1.039559=	1.039641=	1.039663
MW5	2	1125	0.082853=	0.010235=	0.010235=	0.010235
MW6	2	814	0.119352=	0.135437=	0.141371=	0.119553
MW7	2	814	0.363979=	0.364048=	0.363887=	0.364093
MW8	3	1223	0.391247=	0.345615=	0.351341=	0.472479
MW9	2	764	0.295782=	0.295103=	0.294730=	0.295258
MW10	2	1680	0.000000=	0.000000=	0.000000=	0.000000
MW11	2	681	0.268410=	0.268487=	0.268572=	0.268541
MW12	2	732	0.570348=	0.569705–	0.570263=	0.570297
MW13	2	723	0.299718=	0.300917=	0.300099=	0.301507
MW14	3	679	0.213535+	0.209248=	0.209682=	0.209878
ZDT1	2	683	0.681874+	0.681874+	0.681869=	0.681871
ZDT2	2	774	0.348792–	0.348793=	0.348792–	0.348793
ZDT3	2	664	0.987924–	1.021885=	1.023246=	1.022636
ZDT4	2	1481	0.681873+	0.681873+	0.681780=	0.681782
ZDT6	2	1209	0.309905–	0.319846–	0.325169–	0.333011
Total (+/=/–) $\longrightarrow$			04/31/**14**	02/38/**09**	00/41/**08**	of 49 probs.

- With $k = n_{\mathrm{var}}$ (proposed), the performance was either statistically better than or equivalent to other settings of k in all (28 out of 28) instances. As evident, the proposed setting of k performed well compared to other settings.
- With $k = 0.5n_{\mathrm{var}}$, the performance was statistically better than or equivalent to other settings of k in only 22 out of 28 instances. This deteriorated performance could be attributed to the lower value of k than desired, implying that the performance of the IP3 operator is sensitive towards variation in k, for $k < n_{\mathrm{var}}$.
- With $k = 1.5n_{\mathrm{var}}$, the performance was statistically better than or equivalent to other settings of k in 27 out of 28 instances. Despite this slight deterioration in the overall performance, it is fair to infer that the IP3 operator's performance is not very sensitive towards variation in k, for $k > n_{\mathrm{var}}$.

The above summary of results endorses the use of $k = n_{\mathrm{var}}$ for the kNN method in the IP3 operator. Notably, only a limited variation in k has been investigated here, owing to the scope of this paper that focuses on providing the proof-of-concept that ML methods could be used for such performance enhancements in RV-EMâOAs, rather than tuning the parameters of these ML methods.

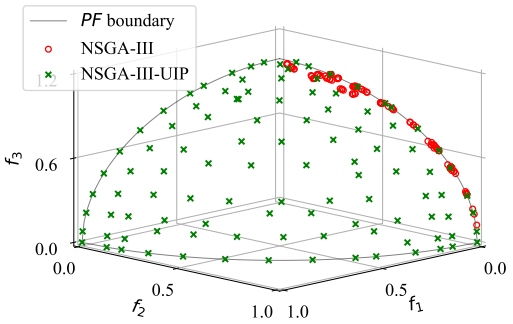
S7. Additional Results

Table S6 represents the results of LHFiD-UIP vis-à-vis its variants: LHFiD, LHFiD-IP2$^+$ and LHFiD-IP3. These results

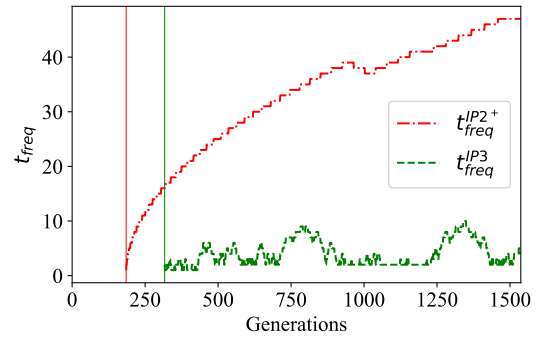
are discussed in Section VII of the main paper and are an extension to Table II, presented in the main paper. Owing to the page width limit, these results have been tabulated here.

S8. Insights into the UIP Operator: $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ Adaptation

The adaptation of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ in the UIP operator, has been discussed in Section V (main paper) and Section S3-B. While the adaptation of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ is based on the survival of the offspring Q^{IP2^+} and Q^{IP3}, respectively, it is imperative to investigate if the resulting adaptation is as desired. Towards this, a sample illustration with the three-objective diversity-hard DASCMOP9 problem is presented in this section (Figure S2). In that, Figure S2a shows the final obtained solutions in the respective median runs of NSGA-III and NSGA-III-UIP. It can be observed that base NSGA-III offers a reasonable convergence to the PF, while offering only partial coverage across the PF. This suggests that in DASCMOP9: (a) the pro-convergence IP2$^+$ operator is less desired, implying a higher $t^{\text{IP2}^+}_{\text{freq}}$, since base NSGA-III can achieve a good convergence by itself, and (b) the pro-diversity IP3 operator is more desired, implying a lower $t^{\text{IP3}}_{\text{freq}}$, since base NSGA-III could not achieve a reasonable diversity by itself. A similar trend of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$, as desired, can be observed in Figure S2b, where the adaptation function led to a higher $t^{\text{IP2}^+}_{\text{freq}}$ and a lower $t^{\text{IP3}}_{\text{freq}}$. This suggests that the adaptation of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$, based on the offspring survival, aligns with the need for emphasizing on convergence and diversity.



(a) Final obtained solutions in the median run of NSGA-III and NSGA-III-UIP



(b) Variation of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$, across generations, of the median NSGA-III-UIP run.

Fig. S2: Analysis of $t^{\text{IP2}^+}_{\text{freq}}$ and $t^{\text{IP3}}_{\text{freq}}$ on the diversity-hard DASCMOP9 problem.

S9. Computational Run-time of an RV-EMâOA with UIP Operator

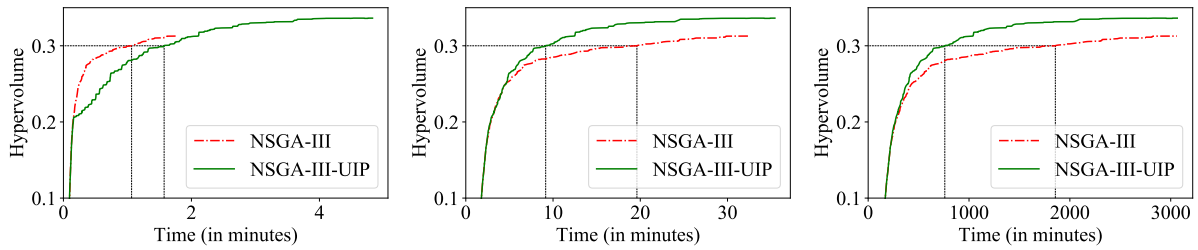
For real-world problems, the time spent on solution evaluations quite often constitutes a dominant fraction of the overall run time for the optimization process. Moreover, these solution evaluations are costly in terms of the resources (experimental or computational solvers, other than the optimizer) needed for evaluation. Hence, any methodological intervention that could help ensure a reasonably good PF-approximation in fewer solution evaluations could have immense utility. However, it cannot be ignored that any such intervention may require additional solution evaluations and computational time, for the underlying methodology to come into effect. Hence, the total number of solution evaluations (collectively needed by the original optimizer and the methodology), and the total time (required for actual solution evaluations and methodology implementation), need to be factored in. In this context, it is notable that for a desired or pre-fixed quality of PF-approximation, any methodological intervention may pose two promising scenarios vis-à-vis the base case (without any intervention), where:

- fewer total solution evaluations may be needed, and also the total run time may be lower, and
- fewer total solution evaluations may be needed, but the total run time may be higher.

In the context of the UIP operator, the results discussed above testify to its promise for fewer total solution evaluations for the desired quality of PF-approximation. These include:

- the fact that no new solution evaluations are required,
- the fact that when integrated with an RV-EMâOA, it promises a better or equivalent PF-approximation than the stand-alone RV-EMâOA, *at any given generation*.

Critically, any specific generation of an RV-EMâO-UIP run, where either or both of IP2$^+$ and IP3 are invoked, will take more time than any generation of the base RV-EMâOA (without UIP), which may be attributed to the construction of underlying training-dataset(s) and subsequently time-consuming training of ML model(s). Hence, *a better PF-approximation after a fixed number of generations (equivalently, after a fixed number of solution evaluations), may not necessarily translate to a better PF-approximation in lower total run time*. This sets up the motivation for investigating the UIP operator with regard to its associated run time. Towards that end, a sample analysis focusing on the performance of NSGA-III and NSGA-III-UIP on the $\tilde{\text{Z}}$DT6 problem is presented here, assisted by the following terminology. Let $\mathcal{T}_{\text{SE}}$ denote the time, in *seconds*, required for one solution evaluation. Also, let $\mathcal{T}_{\text{base}}$ and $\mathcal{T}_{\text{UIP}}$ denote the time, in *minutes*, required to complete one algorithmic run of NSGA-III and NSGA-III-UIP, respectively, till t_{TM} generations. Under the computational set-up employed and experimental settings highlighted earlier, it turns out that $\mathcal{T}_{\text{SE}}$ = 1.05e-04 (0.105 milliseconds), $\mathcal{T}_{\text{base}}$ = 1.752, and $\mathcal{T}_{\text{UIP}}$ = 4.829. Notably, $\rho = \mathcal{T}_{\text{UIP}}/\mathcal{T}_{\text{base}} = 2.756$, is significant, as base NSGA-III may lead to a better PF-approximation than NSGA-III-UIP, if allowed more genera-



(a) $\mathcal{T}_{\rm SE} \approx 0.0001$, $\rho = 2.756$ (b) $\mathcal{T}_{\rm SE} = 0.01$, $\rho = 1.095$ (c) $\mathcal{T}_{\rm SE} = 1.0$, $\rho = 1.001$

Fig. S3: ZDT6: Total run time analysis for NSGA-III versus NSGA-III-UIP. The black horizontal lines mark a pre-fixed quality of PF-approximation to assess relative performance. Here $\rho = \mathcal{T}_{\rm UIP}/\mathcal{T}_{\rm base}$ and $t_{\rm TM} = 1836$.

tions with a run time equivalent of $(\mathcal{T}_{\rm UIP} - \mathcal{T}_{\rm base})$ minutes.

The above suggests the possibility that *even though a pre-fixed quality of PF approximation may be offered by NSGA-III-UIP in fewer generations or equivalently fewer solution evaluations than NSGA-III, it may still require higher overall run time*. Figure S3a, which represents the median run of ZDT6, testifies that the above possibility has indeed come true. Notably,

- the horizontal axis in Figure S3a represents the total run time (instead of the number of generations) since the current focus is to evaluate NSGA-III-UIP versus NSGA-III, based on the total run time, and
- the total time required by NSGA-III to complete $t_{\rm TM} = 1836$ generations is significantly lower than the corresponding time required by NSGA-III-UIP.

The specific instance above could *mistakenly* lead to the inference that *for a pre-fixed quality of PF-approximation, though NSGA-III-UIP may require fewer generations or equivalently fewer solution evaluations than NSGA-III, its total run time can never be comparable or lower than that of NSGA-III*. The basis for such a misconception has been countered below through variation in $\mathcal{T}_{\rm SE}$. To symbolically emulate real-world scenarios where each solution evaluation may take a significantly larger time, two hypothetical values of $\mathcal{T}_{\rm SE} = 0.01$ and $\mathcal{T}_{\rm SE} = 1.0$ have been considered. Though, in the case of ZDT6, actual $\mathcal{T}_{\rm SE} = 1.05\text{e-}04$, the solution evaluations have been inter-spaced by 0.1 and 10.0 seconds, respectively, (using the *sleep* function available in the computational set-up used) to emulate the two scenarios. It may be noted that, as $\mathcal{T}_{\rm SE}$ values rise from $1.05\text{e-}04$, through 0.01, to 1.0 (seconds), the ρ values fall from 2.756, through 1.095, to 1.001. Interestingly, with $\mathcal{T}_{\rm SE} = 0.1$ or 1.0, which could be quite common in real-world problems, the time required for a complete algorithmic run of NSGA-III-UIP becomes nearly comparable to that of NSGA-III. These instances also facilitate another insightful revelation. For a pre-fixed quality of PF-approximation, represented by, say, a hypervolume of 0.3, as highlighted by horizontal lines in Figure S3:

- the solution evaluations required by NSGA-III-UIP are $45,400$ (independent of $\mathcal{T}_{\rm SE}$ values), while standalone NSGA-III requires $111,200$ solution evaluations,
- the total run time for NSGA-III-UIP is relatively higher than that of NSGA-III, if $\mathcal{T}_{\rm SE} = 1.05\text{e-}04$, and
- the total run time for NSGA-III-UIP is relatively lower than that of NSGA-III, if $\mathcal{T}_{\rm SE} = 0.1$ or 1.0.

Based on the above investigation, it can be inferred that *NSGA-III-UIP promises to offer a pre-fixed quality of PF-approximation, in fewer generations or equivalently fewer solution evaluations, than NSGA-III, while its total run time may be higher or lower than that of NSGA-III depending on the time that each solution evaluation requires*. Clearly, the utility of incorporating the UIP operator may be far more significant in problems where each solution evaluation requires considerable time.

REFERENCES

[1] S. Mittal, D. K. Saxena, K. Deb, and E. D. Goodman, "Enhanced *Innovized*' progress operator for evolutionary multi-and many-objective optimization," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[2] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 16–37, 2016.

[3] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.

[4] D. K. Saxena, S. Mittal, S. Kapoor, and K. Deb, "A localized high-fidelity-dominance based many-objective evolutionary algorithm," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

[5] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part II: Handling constraints and extending to an adaptive approach," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2014.

[6] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000. [Online]. Available: https://doi.org/10.1162/106365600568202

[7] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*. London: Springer London, 2005, pp. 105–145. [Online]. Available: https://doi.org/10.1007/1-84628-137-7_6

[8] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao, "A benchmark test suite for evolutionary many-objective optimization," *Complex & Intelligent Systems*, vol. 3, pp. 67–81, 2017.

[9] J. Lin, H. Liu, and C. Peng, "The effect of feasible region on imbalanced problem in constrained multi-objective optimization," in *2017 13th International Conference on Computational Intelligence and Security (CIS)*, 2017, pp. 82–86.

[10] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, "Difficulty Adjustable and Scalable Constrained Multiobjective Test Problem Toolkit," *Evolutionary Computation*, vol. 28, no. 3, pp. 339–378, 09 2020. [Online]. Available: https://doi.org/10.1162/evco_a_00259

[11] Z. Ma and Y. Wang, "Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 972–986, 2019.