Hybridizing Hypervolume-Based Evolutionary Algorithms and Gradient Descent by Dynamic Resource Allocation

Ha, Damy M.F.; Deist, Timo M.; Bosman, Peter A.N.

DOI
10.1007/978-3-031-14721-0_13

Publication date
2022

Document Version
Final published version

Published in
Parallel Problem Solving from Nature – PPSN XVII - 17th International Conference, PPSN 2022, Proceedings

Hybridizing Hypervolume-Based Evolutionary Algorithms and Gradient Descent by Dynamic Resource Allocation

Damy M. F. Ha[1,2(✉)], Timo M. Deist[2], and Peter A. N. Bosman[1,2]

[1] Delft University of Technology, Delft, The Netherlands
d.m.f.ha@student.tudelft.nl, P.A.N.Bosman@tudelft.nl
[2] Centrum Wiskunde and Informatica, Life Sciences and Health Research Group, Amsterdam, The Netherlands
{dmfh,timo.deist,peter.bosman}@cwi.nl

Abstract. Evolutionary algorithms (EAs) are well-known to be well suited for multi-objective (MO) optimization. However, especially in the case of real-valued variables, classic domination-based approaches are known to lose selection pressure when approaching the Pareto set. Indicator-based approaches, such as optimizing the uncrowded hypervolume (UHV), can overcome this issue and ensure that individual solutions converge to the Pareto set. Recently, a gradient-based UHV algorithm, known as UHV-ADAM, was shown to be more efficient than (UHV-based) EAs if few local optima are present. Combining the two techniques could exploit synergies, i.e., the EA could be leveraged to avoid local optima while the efficiency of gradient algorithms could speed up convergence to the Pareto set. It is a priori however not clear what would be the best way to make such a combination. In this work, therefore, we study the use of a dynamic resource allocation scheme to create hybrid UHV-based algorithms. On several bi-objective benchmarks, we find that the hybrid algorithms produce similar or better results than the EA or gradient-based algorithm alone, even when finite differences are used to approximate gradients. The implementation of the hybrid algorithm is available at https://github.com/damyha/uncrowded-hypervolume.

Keywords: Real-valued optimization · Multi-objective · Hybrid algorithm

1 Introduction

In real-valued multi-objective (MO) optimization, multiple conflicting objectives need to be optimized. The goal of MO optimization often is to find a diverse set

Supported by Open Technology Programme (nr. 15586) financed by Dutch Research Council (NWO), Elekta, and Xomnia. Cofunding by Ministry of Economic Affairs: public-private partnership allowance for top consortia for knowledge and innovation (TKIs).

G. Rudolph et al. (Eds.): PPSN 2022, LNCS 13399, pp. 179–192, 2022.
https://doi.org/10.1007/978-3-031-14721-0_13

of (near-)Pareto optimal solutions, and usually to do so as efficiently as possible. Evolutionary algorithms (EAs) (e.g. [6,9]) are known to be well suited for MO optimization [8]. However, in real-valued MO optimization, classic domination-based approaches lose selection pressure when approaching the Pareto set [2]. Indicator-based approaches, such as optimizing the hypervolume (HV) [21] or the uncrowded hypervolume (UHV) [12,17,19] can overcome this issue and ensure that individual solutions converge to the Pareto set. Recently, a gradient-based UHV algorithm known as UHV-ADAM [10] was shown to be more efficient than (UHV-based) EAs if few local optima are present. EAs generally remain more efficient if many local optima are present. Combining the two techniques could exploit synergies, especially in problems with many local optima, i.e., the EA could be leveraged to avoid local optima while the gradient algorithms could be leveraged to efficiently converge to the Pareto set. It is however unknown a priori, how the techniques should be combined to get the best results.

Attempts in the literature have been successful at creating efficient MO hybrid algorithms (also known as memetic algorithms). In [18] a hybrid algorithm was proposed that probabilistically executes different variation operators of EAs. Gradient algorithms however have not been integrated into their work. In [3] a domination-based EA was combined with gradient-based algorithms that exploit either the gradient of a single-objective or a combination thereof that corresponds to maximum improvement in a multi-objective sense. In [3], resources are furthermore dynamically assigned to the gradient algorithms via a resource allocation scheme (RAS). A HV-based hybrid algorithm was introduced in [13], which combines both an EA and gradient algorithm that aim to maximize the HV. In contrast to [3] however, [13] executes the gradient algorithm after the EA is finished. Supplementing the EA during evolution however might be of key value.

In this work, we study the potential of unifying the convergence properties of UHV-based MO algorithms with a hybrid interleaving optimization scheme. Specifically, we formulate a new UHV-based hybrid algorithm and show that the hybrid algorithm is capable of performing better than the worst of the original algorithms or in some cases better than both algorithms. For this, we combine a UHV-based EA called UHV-GOMEA [17] with UHV-ADAM [10] by extending the RAS of [3]. The resulting hybrid algorithm is consequently UHV-based. The UHV distinguishes itself in that a set of solutions is optimized instead of individual solutions. Concretely, this means that the UHV-based hybrid (and EA) employ a population of solution sets, not a population of individual solutions. Each solution set is optimized towards the Pareto set. In this work, we empirically determine the hybrid's architecture using a similar set of benchmarks as in [17] and [10]. We then compare the final algorithm with its component algorithms, UHV-GOMEA and UHV-ADAM, and another UHV-based gradient algorithm on the Walking Fish Group (WFG) benchmark set [15]. The remainder of this document is organized as follows: In Sects. 2 and 3 we introduce the UHV indicator and existing UHV algorithms. In Sect. 4 we introduce the hybrid algorithm with its RAS. The experiments follow in Sect. 5, with a discussion and conclusion in Sects. 6 and 7 respectively.

2 Uncrowded Hypervolume Optimization

We consider an MO optimization problem to be a problem where m objective functions need to be minimized. Let $\mathbf{f} : \mathcal{X} \to \mathbb{R}^m$, with $\mathbf{f} = [f_0, ..., f_{m-1}]$, be an m-dimensional vector of objective functions, where $\mathcal{X} \subseteq \mathbb{R}^n$ is an n-dimensional search space. In this work we focus on bi-objective problems ($m = 2$). A solution $\mathbf{x} \in \mathcal{X}$, where $\mathbf{x} = [x_0, ..., x_{n-1}]$, will be called an MO-solution. The goal of MO optimization is to find a set $\mathbb{S}$ of diverse and (near-)Pareto-optimal MO-solutions. To achieve this, we assess the quality of $\mathbb{S}$ via the UHV indicator function in Eq. 1. The UHV measures the hypervolume (HV), i.e. the area in objective space enclosed by the non-dominated solutions of $\mathbb{S}$ and reference point $\mathbf{r} = (r_0, r_1)$, and penalizes the dominated solutions of $\mathbb{S}$ via the uncrowded distance (ud). We refer the reader to [17] for the reasons behind scaling and exponentiation operations on the ud. To calculate the HV, let $\mathcal{A}$ be the approximation set that contains all non-dominated solutions of $\mathbb{S}$. $\mathcal{A}$ then forms an approximation boundary $\partial\mathbf{f}(\mathbb{S})$ in objective space. The reader is referred to [21] on how $\partial\mathbf{f}(\mathbb{S})$ is calculated. The HV is the region encapsulated between approximation boundary $\partial\mathbf{f}(\mathbb{S})$ and reference point $\mathbf{r}$, as shown in Fig. 1. The aforementioned uncrowded distance $\text{ud}(\mathbf{x}, \mathbb{S})$ is the closest Euclidean distance between MO-solution $\mathbf{x}$'s objective values $\mathbf{f}(\mathbf{x})$ and the approximation boundary $\partial\mathbf{f}(\mathbb{S})$. By definition, $\text{ud}(\mathbf{x}, \mathbb{S})$ is zero for a non-dominated solution. Using the UHV indicator, an MO problem is effectively reformulated as a single-objective problem. The goal of UHV-based algorithms is to maximize the UHV, as maximization leads directly to the minimization of the original objective functions as well improving the diversity [1].

$$\text{UHV}(\mathbb{S}) = \text{HV}(\mathbb{S}) - \frac{1}{|\mathbb{S}|} \sum_{\mathbf{x} \in \mathbb{S}} \text{ud}(\mathbf{x}, \mathbb{S})^m \tag{1}$$

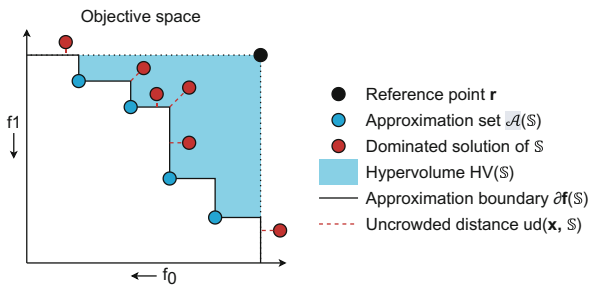

Fig. 1. Illustration of the UHV of $\mathbb{S}$ for an arbitrary bi-objective problem.

3 UHV-Based Algorithms

3.1 UHV-ADAM

UHV-ADAM [10] is based on the single objective stochastic gradient algorithm ADAM [16]. UHV-ADAM parameterizes a single solution set $\mathbb{S}$ of p number of

MO-solutions as ϕ^0, such that $\phi^0 = [\mathbf{x_0}, ..., \mathbf{x_{p-1}}] \in \mathbb{R}^{p \cdot n}$. Let $\mathbf{F}(\phi^0)$ be the operator that assesses the objective functions for every MO-solution in ϕ^0, as displayed in Eq. 2. UHV-ADAM starts by randomly initializing the MO-solutions of ϕ^0 and evaluates the objective values ($f_0(\mathbf{x_i})$, $f_1(\mathbf{x_i})$) and objective gradients ($\nabla f_0(\mathbf{x_i})$, $\nabla f_1(\mathbf{x_i})$) for every MO-solution $\mathbf{x_i}$ of solution set ϕ^0. Using the objective values and objective function gradients, the gradient of the UHV indicator $\nabla \text{UHV}(\phi^0)$ is calculated. $\nabla \text{UHV}(\phi^0)$ indicates how MO-solutions in the search space must move to (locally) obtain the most UHV gain. The reader is referred to [10,11] on how $\nabla \text{UHV}(\phi^0)$ is exactly calculated. UHV-ADAM then determines the direction in which solutions are moved in the next iteration via a variance-corrected weighted average of $\nabla \text{UHV}(\phi^0)$. How far the solutions are moved is determined by step size factor γ and the variance correction. γ is determined by a shrinking scheme which reduces γ by 1% if no UHV improvement is found. The initial γ is computed by taking 1% of the average initialization range. This initialization method will be used later to reinitialize UHV-ADAM within the hybrid algorithm. UHV-ADAM repeats the process of calculating the UHV gradient and moving the solutions until all computation resources, e.g., a time or function evaluation budget, have been spent or a desired UHV value has been reached.

$$\phi^0 = \begin{bmatrix} \mathbf{x_0} \\ ... \\ \mathbf{x_{p-1}} \end{bmatrix} \rightarrow \mathbf{F}(\phi^0) = \begin{bmatrix} \mathbf{f(x_0)} \\ ... \\ \mathbf{f(x_{p-1})} \end{bmatrix} = \begin{bmatrix} f_0(\mathbf{x_0}) & ... & f_{m-1}(\mathbf{x_0}) \\ \vdots & \ddots & \vdots \\ f_0(\mathbf{x_{p-1}}) & ... & f_{m-1}(\mathbf{x_{p-1}}) \end{bmatrix} \tag{2}$$

3.2 UHV-GOMEA

The Uncrowded Hypervolume Gene-pool Optimal Mixing Evolutionary Algorithm (UHV-GOMEA) [17] is a recently introduced UHV-based EA that leverages strengths of the single-objective model-based EA known as RV-GOMEA [6]. UHV-GOMEA starts off by randomly initializing and evaluating a population of N solution sets: $\phi = \left[\phi^0, \cdots, \phi^{N-1}\right]$, where each individual ϕ^i $(i = 0, \cdots, N-1)$ has p MO-solutions. Gradient information is not used nor calculated. UHV-GOMEA then selects the best 35% of the solution sets with the highest UHV value as parents. A variation operator is applied on the parents to create new offspring solution sets. This process is repeated until termination. UHV-GOMEA's variation operator makes use of linkage models. In this work, only the marginal product linkage model (Lm) is used. Lm greedily rearranges the MO-solutions of each solution set such that all i'th MO-solution $\mathbf{x_i}$ $(i = 0, \ldots, p-1)$ of each solution set is in the same region of the approximation front. It then groups all variables pertaining to $\mathbf{x_i}$ into sets. These sets together compromise a FOS (Family Of Subsets) denoted as $\mathcal{F}$. For each $\mathcal{F}$, a Gaussian distribution is estimated. These Gaussians are used to create offspring by sampling MO-solutions from this marginal product distribution and to inject the new MO-solutions into each individual of the population. If the UHV improves, changes are kept. Otherwise,

they are rejected. For more details, including how the Gaussians are estimated and adapted during evolution, the reader is referred to [6,17].

4 Hybridization

4.1 Changes Made to UHV-ADAM

In this work UHV-ADAM has been extended such that the single-solution set solving algorithm is compatible with population-based UHV-GOMEA. To this end, UHV-ADAM steps are applied to population members after a run of UHV-GOMEA. UHV-ADAM instances are assigned to each solution set of the population, allowing the weighted moving average and γ to be tuned accurately and differently to the environment of each solution set in the population. UHV-ADAM instances are reset every time the variation operator of UHV-GOMEA is applied to prevent γ and the moving averages of UHV-ADAM instances to become inaccurate if UHV-GOMEA makes big leaps in the search space. Resetting the UHV-ADAM instances comes at the cost of warming up the moving averages again as well as redetermining γ. γ is re-estimated by creating the tightest box that contains all MO-solutions of the population and to take 1% of the average box width. Finally, a RAS will be used to adaptively determine which algorithm (ADAM or GOMEA) should be used more during a run. After determining the resource distribution, the resources assigned to UHV-ADAM must be distributed over the population members. Early experiments have shown that distributing among the 3 solutions with the highest UHV works the best, but this will be further investigated in Sect. 5.3.

4.2 Resource Allocation Scheme

The hybrid created in this work is based on [3], where a resource allocation scheme (RAS) is used. In this work, only UHV-GOMEA and the modified UHV-ADAM are hybridized. The hybrid algorithm executes UHV-GOMEA and UHV-ADAM sequentially. UHV-GOMEA is always executed once per generation, while the RAS determines the number of UHV-ADAM steps. The RAS of [3] is extended to accommodate the modified UHV-ADAM and works as follows: let the actual number of evaluations and improvements found in generation t by optimizer $o \in \{\text{GOMEA}, \text{ADAM}\}$ be $E_o(t)$ and $I_o(t)$ respectively. An evaluation occurs when one MO-solution $\mathbf{x_i}$ is evaluated. What entails an improvement will be discussed later in Experiment 1. Let the number of evaluations and improvements to be considered for redistribution be $\mathcal{E}_o(t)$ and $\mathcal{I}_o(t)$ respectively. For UHV-ADAM, only the values of the current generation are of interest, that is: $\mathcal{E}_{\text{ADAM}}(t) = E_{\text{ADAM}}(t)$ and $\mathcal{I}_{\text{ADAM}}(t) = I_{\text{ADAM}}(t)$. The number of evaluations and improvements to be considered for UHV-GOMEA is a sum of values of previous generations, that is: $\mathcal{E}_{\text{GOMEA}}(t) = \sum_{t'=t_{min}}^{t} E_{\text{GOMEA}}(t)$ and $\mathcal{I}_{\text{GOMEA}}(t) = \sum_{t'=t_{min}}^{t} I_{\text{GOMEA}}(t)$, where $t_{min} \geq 0$ and t_{min} is chosen as large as possible such that $\mathcal{E}_{\text{GOMEA}}(t) \geq \mathcal{E}_{\text{ADAM}}(t)$ still holds. UHV-GOMEA includes

past values for two reasons: it makes the comparison between the gradient algorithm and EA fairer and also allows the number of gradient algorithm calls to grow [4]. Following [4], the EA's variation operator is executed once per generation while the number of executions of the gradient algorithms are related to the respective reward they receive. The reward, displayed in Eq. 3, is the efficiency of finding improvements. The reward is 0 if $\mathcal{E}_o(t) = 0$.

$$\mathcal{R}_{\mathrm{o}}(t) = \frac{\mathcal{I}_o(t)}{\mathcal{E}_o(t)} \tag{3}$$

Let the evaluations to be redistributed to UHV-ADAM be $\mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)$. $\mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)$ is the ratio of UHV-ADAM's contribution to the total reward times the total sum of evaluations to be considered in generation t as shown in Eq. 4.

$$\mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t) = \frac{\mathcal{R}_{\mathrm{ADAM}}(t)}{\sum_{o'} \mathcal{R}_{o'}(t)} \sum_{o'} \mathcal{E}_{o'}(t) \tag{4}$$

To calculate the number of iterations UHV-ADAM can execute with budget $\mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)$, let the number of calls be $\mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)$, where $\mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)$ can be calculated by dividing the resources assigned to UHV-ADAM by the average number of evaluations required per call. The average evaluations per call are estimated using the resources and calls of generation t, resulting in Eq. 5.

$$\mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t) = \frac{\mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t)}{\frac{\mathcal{E}_{\mathrm{ADAM}}(t)}{\mathcal{C}_{\mathrm{ADAM}}(t)}} = \frac{\mathcal{C}_{\mathrm{ADAM}}(t)}{\mathcal{E}_{\mathrm{ADAM}}(t)} \mathcal{E}^{\mathrm{Red}}_{\mathrm{ADAM}}(t) \tag{5}$$

To ensure a smooth decrease in the number of gradient calls, memory decay is implemented in Eq. 6. If the number of calls after redistribution is smaller than the number of calls executed in the current generation, a running average is used to decrease the number of calls. If the number of calls increases, memory decay is not applied in order to stimulate the use of gradient Algorithms [4]. The (memory) decay factor η is kept at the original value of 0.75 [4].

$$\mathcal{C}^{\mathrm{Run}}_{\mathrm{ADAM}}(t+1) = \begin{cases} \mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t), & \text{if } \mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t) \geq \mathcal{C}^{\mathrm{Run}}_{\mathrm{ADAM}}(t) \\ \eta \mathcal{C}^{\mathrm{Run}}_{\mathrm{ADAM}}(t) + (1-\eta)\mathcal{C}^{\mathrm{Red}}_{\mathrm{ADAM}}(t), & \text{otherwise} \end{cases} \tag{6}$$

The number of UHV-ADAM calls to execute next generation could be set to $\mathcal{C}_{\mathrm{ADAM}}(t+1) = \left\lfloor \mathcal{C}^{\mathrm{Run}}_{\mathrm{ADAM}}(t+1) \right\rfloor$, However, if at some point $\mathcal{C}_{\mathrm{ADAM}}(t) = 0$ holds, UHV-ADAM cannot be activated any more. As UHV-ADAM could become useful again in the future, a waiting scheme is used that makes UHV-ADAM wait $\mathcal{W}_{\mathrm{ADAM}}(t)$ generations. In [4], gradient algorithms are only allowed to be executed at most once per individual per generation. Furthermore, at most (population size) N number of total calls can be executed per generation. Early experiments have shown that executing one UHV-ADAM call per individual does not substantially affect convergence. For this reason, multiple gradient calls can be applied to the same individual. Furthermore, a lower bound is introduced such that if UHV-ADAM is to be executed, it executes at least $\mathcal{C}^{\mathrm{min}}_{\mathrm{ADAM}}$ calls.

This ensures that the performance of UHV-ADAM is assessed after it warms up its internal parameters. $\mathcal{C}^{\min}_{\text{ADAM}}$ is set to 10 and has not been further optimized. The cap on total gradient calls is kept and set to N. The modified waiting scheme is shown in Eq. 7. UHV-ADAM is forced to wait for some generations when $\mathcal{C}^{\text{Run}}_{\text{ADAM}}(t+1) \leq \mathcal{C}^{\min}_{\text{ADAM}}$. Because the extended UHV-ADAM executes a minimum number of calls, $\mathcal{C}^{\min}_{\text{ADAM}}$ has been added to prevent the waiting scheme from triggering too early. The actual number of calls to be executed is shown in Eq. 8, where $\mathcal{C}^{\min}_{\text{ADAM}}$ has also been added to the original Equation.

$$\mathcal{W}_{\text{ADAM}}(t+1) = \begin{cases} \left\lfloor \frac{\mathcal{C}^{\min}_{\text{ADAM}}}{\mathcal{C}^{\text{Run}}_{\text{ADAM}}(t+1)} \right\rfloor, & \text{if } \mathcal{W}_{\text{ADAM}}(t) = 0 \\ \mathcal{W}_{\text{ADAM}}(t) - 1, & \text{otherwise} \end{cases} \tag{7}$$

$$\mathcal{C}_{\text{ADAM}}(t) = \begin{cases} \mathcal{C}^{\min}_{\text{ADAM}}, & \text{if } \mathcal{W}_{\text{ADAM}}(t-1) = 1 \\ \min(\left\lfloor \mathcal{C}^{\text{Run}}_{\text{ADAM}}(t) \right\rfloor, N), & \text{otherwise} \end{cases} \tag{8}$$

5 Experiments

5.1 Experimental Setup

The problems used in the experiments are given in Table 1, where n is the problem dimensionality. Problem 0 is uni-modal, objective-wise decomposable [17] and can be quickly solved with gradient Algorithms [10]. Problem 1 is a low multi-modal problem based on the Rosenbrock function which has pair-wise dependencies [5]. It is known for pulling algorithms towards the optimum of the more easily solvable Sphere function while potentially getting solutions stuck in a local optimum of the Rosenbrock function. Problem 2 contains the multi-modal Rastrigin [14] problem, where many local optima are evenly scattered around the solution space. Problem 3 is multi-modal in both objectives where the Pareto set is enveloped by basins. The Pareto sets of all problems lie on a line between the respective optima.

Table 1. The bi-objective benchmark problems selected for the experiments.

#	Problem name	Objectives	Properties
0	Convex bi-sphere	$f_0 = f_{\text{sphere}}(\mathbf{x})$, with $f_{\text{sphere}}(\mathbf{x}) = \sum_{i=0}^{n-1}(x_i)^2$ $f_1 = f_{\text{sphere}}(\mathbf{x} - \mathbf{c_0})$ $\mathbf{c}_0 = [1, 0, \cdots, 0]$	Uni-modal, decomposable
1	Convex sphere Rosenbrock	$f_0 = \frac{1}{n} f_{\text{sphere}}(\mathbf{x})$ $f_1 = \frac{1}{n-1} f_{\text{ros}}(\mathbf{x})$, with $f_{\text{ros}}(\mathbf{x}) = \sum_{i=0}^{n-1}(100(x_i - x_{i-1}^2)^2 + (1 - x_{i-1})^2)$	Multi-modal, attraction to f_0
2	Convex sphere Rastrigin	$f_0 = f_{\text{sphere}}(\mathbf{x})$ $f_1 = f_{\text{rast}}(\mathbf{x} - \mathbf{c_2})$, with $f_{\text{rast}}(\mathbf{x}) = An + \sum_{i=0}^{n-1} x_i^2 - A\cos(2\pi x_i)$ $A = 10, \mathbf{c}_2 = [0.5, 0, \cdots, 0]$	Multi-modal
3	Bi-cosine sphere	$f_0 = f_{\cos}(\mathbf{x})$, with $f_{\cos}(\mathbf{x}) = f_{\text{sphere}}(\mathbf{x})(1 - \beta\cos(2\pi f \lvert\mathbf{x}\rvert))$ $f_1 = f_{\cos}(\mathbf{x} - \mathbf{c_0})$ $\beta = 0.6, f = 0.1$	Multi-modal in f_0 and f_1

5.2 Experiment 1: The Effect of the Improvement Metric

In experiment 1, the problems from Table 1 are used to assess the effects of different improvement metrics $I_o(t)$. Problem 0 is excluded from this experiment as tuning the hybrid algorithm on this easily solvable problem is undesirable. Metrics ΔBestUHV and ΔAverageUHV are the difference between the best found UHV and average population UHV respectively in subsequent generations. CountUHVImproved and CountBestUHVImproved count the number of times the UHV of a solution and that of the best solution have improved respectively. In [3], the number of MO-solutions added to an elitist archive is counted. We will identify that metric with Bosman2012. Here, we use an infinitely large elitist archive to encourage counting MO-solutions that improve the UHV, which otherwise are potentially rejected by a (nearly) full, finite sized elitist archive. For clarity, the elitist archive is not used for anything but the improvement metric.

Gradient calls are applied to the best 3 solutions of the population that have the highest UHV. The solution set size is set to $p = 9$. As we do not know the HV of Pareto set $\mathcal{A}^\star$ analytically, $\text{HV}(\mathcal{A}^\star)$ is set to the maximum HV obtained from running all algorithms 30 times, while initializing the algorithms near the Pareto set. In experiment 1 we run each improvement metric on problems $P = [1, 2, 3]$ for the following problem dimensionalities $D = [2, 5, 10, 20, 40, 80]$. For each dimension, we determine the best population N by running the following population sizes $N = [40, 80, 160, 320, 640, 1280]$ 30 times and select the most efficient population size that reaches a target HV accuracy of $\Delta\text{HV}_p < 10^{-6}$ with a success rate of at least 29 out of 30 runs. We consider runs that need more than 10^7 MO-evaluations to have failed in finding the target HV. We then compute a performance score, which sums the relative performance of improvement metric *imp* with respect to the best performing improvement metric amongst all improvement metrics I, over all problems P and problem dimensionalities D in Eq. 9.

$$\text{score(imp)} = \sum_{\text{pr}\in\text{P}} \sum_{\text{d}\in\text{D}} \frac{\text{median(MO-Evaluations(pr, d, imp))}}{\min_{\text{imp}'\in\text{I}}(\text{median(MO-Evaluations(pr, d, imp}'))))} \tag{9}$$

Table 2 shows the results. Using ΔBestUHV obtains the best score in all problems except Problem 1. ΔAverageUHV is consistently performing the worst. ΔAverageUHV is generally biased towards rewarding UHV-GOMEA as UHV-ADAM is not designed to efficiently optimize an entire population. Experiment 1 shows that it is not trivial to select an improvement metric that is superior for all problems. Instead, improvement metrics appear to be problem specific. However, as ΔBestUHV has the best average score, it will be used in further experiments.

5.3 Experiment 2: The Effect of the Choice of Method to Distribute Gradient Resources

The effect of the choice of method to distribute the resources assigned to the modified UHV-ADAM, on the required number of MO-evaluations to reach a target HV accuracy of $\Delta\text{HV}_p < 10^{-6}$ and the corresponding success rate (SR)

Table 2. The scores assigned to each improvement metric. The lower the score, the better. The numbers in bold are the lowest scores of a category.

Problem	ΔBestUHV	ΔAverage UHV	Count UHVImproved	CountBestUHV improved	Bosman 2012
Convex sphere & Rosenbrock (1)	9.0	9.0	**8.3**	9.2	9.0
Convex sphere & Rastrigin (2)	**6.1**	9.8	7.9	6.4	7.1
Bi-cosine sphere (3)	**6.1**	8.6	7.1	6.4	6.4
Average	**7.1**	9.1	7.8	7.3	7.5

is shown in Fig. 2. UHV-GOMEA and UHV-ADAM have also been added as a reference. Table 3 shows the scores obtained by the distribution methods using Eq. 9. Following experiment 1, Problem 0 is excluded from Table 3. Distribution methods that are unable to find a population size that meets the SR threshold of 29 out of 30 runs are disqualified and denoted as "DQ". The evaluation budget, population optimization method and solution set size are the same as in experiment 1. The hybrid uses the ΔBestUHV improvement metric. Among the distribution methods, Best-m-Solutions and Best-m%Population apply gradient calls on the best solutions of the population. The former applies calls to a fixed number of solutions sets and the latter to a percentage of the population. ALL applies calls on all solution sets, starting from $\phi^0, \phi^1, \cdots$ until all calls have been distributed. RANDOM applies calls randomly with replacement. In Fig. 2, UHV-GOMEA is generally amongst the worst performing implementations along with UHV-ADAM, which fails to reach the target SR threshold in all problems except Problem 0. In Problem 1 of Fig. 2, the statistics of the successful runs of UHV-ADAM have been displayed despite not meeting the SR threshold. Interestingly, in [10], UHV-ADAM is able to solve Problem 1 when initialized near the global optima ($[0, 2]^n$). In this experiment however, UHV-ADAM gets stuck on local optima due to a larger initialization range. Among the distribution methods, Best3Solutions and BestSolution are on average among the best performing distribution methods according to Table 3. Distribution methods: ALL, RANDOM, Best5%Population and Best10%Population are disqualified for not reaching the target SR. Interestingly, analysis shows that for Problem 3 the improvement metric chosen generally remains in the waiting state until most local optima are no longer within the scope of the population, after which it maximizes the number of UHV-ADAM calls, resulting in similar performance amongst the Best-m-Solutions and Best-m%Population distribution methods. Table 3 clearly shows that concentrating gradient calls on the best solutions is more efficient than diluting gradient calls over the population.

5.4 Experiment 3: The WFG Benchmark

We use the WFG suite [15] as an independent method to benchmark the results of the hybrid algorithm. For detailed characteristics of these 9 benchmark functions,

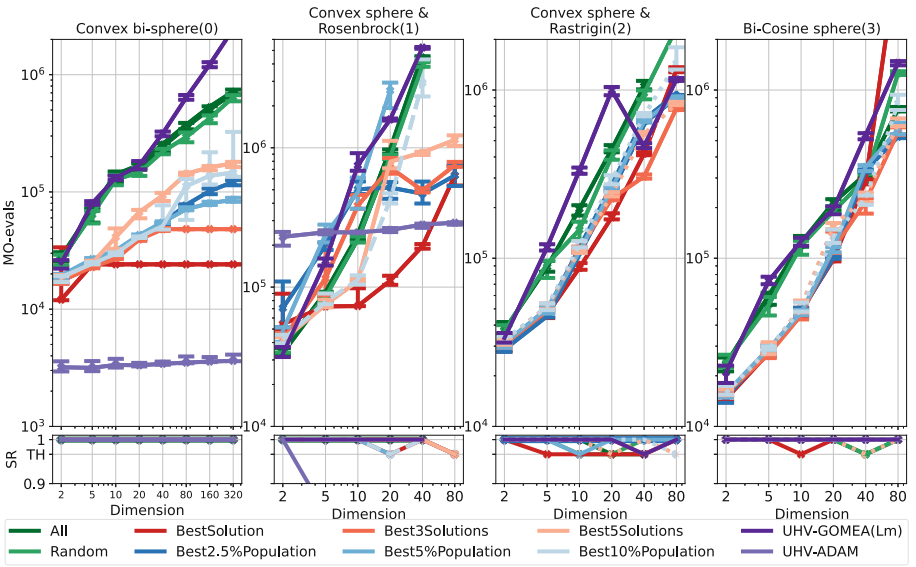

Fig. 2. The effect of different distribution methods on the required number of MO-evaluations to reach a target accuracy of $\Delta\text{HV}_p < 10^{-6}$ for various problems. The success rate (SR) measures the fraction of runs that reach the target accuracy out of 30 runs, where the target threshold (TH) is 29/30 runs.

Table 3. The scores assigned to each distribution method. The lower the score, the better. Numbers in bold are the lowest scores of a category. "DQ" denotes distribution methods that fail one or more success rate thresholds.

Problem	All	Random	Best solution	Best3 solutions	Best5 solutions	Best2.5% population	Best5% population	Best10% population
Convex sphere & Rosenbrock(1)	DQ	DQ	**6.6**	18.3	17.4	19.7	DQ	DQ
Convex sphere & Rastrigin(2)	DQ	13.4	7.1	**6.7**	7.7	8.1	8.2	9.4
Bi-Cosine sphere(3)	11.6	11.9	17.8	**6.3**	7.1	6.7	7.1	7.2
Average	DQ	DQ	10.5	**10.4**	10.7	11.5	DQ	DQ

the reader is referred to [15]. WFG1 is a separable problem, with a flat region which can stagnate the search. WFG2 has a uni-modal disconnected convex front. WFG3 is multi-modal and has a linear front. WFG4-9 all have concave fronts, where WFG4 and WFG9 are multi-modal. Following [17], the benchmark is used in a bi-objective setting with $k_{\text{WFG}} = 4$ position variables and $l_{\text{WFG}} = 20$ distance variables, resulting in $n = 24$ decision variables. The HV reference point $\mathbf{r}$ is set to $r = (11, 11)$. The computation budget is set to 10^7 MO-evaluations for each algorithm. For the solution set size, we use $p = 9$.

The algorithms we consider in this experiment include the base algorithms: UHV-GOMEA(Lm), UHV-ADAM, the constructed hybrid algorithms: ΔBestUHV with distribution methods BestSolution and Best3Solutions, as well

as another UHV-based gradient algorithm called UHV-GA-MO [10]. UHV-GA-MO is based on the GA-MO scheme [20]. We refer the reader to [10] for the exact details of UHV-GA-MO. Each algorithm is executed 30 times. Algorithms that use populations have their population sizes set to 200 following [10,17]. Gradient-based algorithms use finite difference gradient approximations (indicated by the suffix "-FD" in Table 4). Finite difference approximations come at the cost of $(1 + n) \cdot p$ MO-evaluations [10]. Per problem, outcomes are compared to the result with the highest mean value and tested for statistical significance up to 4 decimals by a Wilcoxon two-sided rank-sum test where the initial $\alpha' = 0.05$. α' is Bonferroni corrected by a factor of 36, making the final α to be $\alpha = 0.05/36$.

Table 4 shows that, on average, the best results were obtained with Hybrid-Best3Solutions-FD, followed by Hybrid-BestSolution-FD. Interestingly, the hybrids never obtain a rank worse than 2, indicating that in this experiment, the hybrids on average perform better than the original component algorithms. Furthermore, for problems: WFG1, WFG2, WFG4, WFG6 and WFG8, at least one of the hybrids obtains statistically better HVs than the original component algorithms. Interestingly, in WFG 4, Hybrid-Best3Solutions-FD preforms better than the UHV-GOMEA (Lm) despite WFG 4 being a multi-modal problem.

Table 4. The WFG benchmark for 10^7 MO-evaluations. Hypervolume values are shown (mean, ± standard deviation(rank)). Finite differences (FD) are used for the gradient-based algorithms. Scores in bold are the best or not statistically different from the other bold scores, indicated per problem.

Problem	UHV-GOMEA(Lm)	UHV-ADAM-FD	UHV-GA-MO-FD	Hybrid-BestSolution-FD	Hybrid-Best3Solutions-FD
WFG1	94.63±1.73(5)	97.32±0.60(3)	96.74±0.60(4)	98.90±0.29(2)	**101.57**±0.49(1)
WFG2	110.13±0.03(3)	106.26±5.09(5)	**109.60**±6.68(4)	**110.36**±1.20(2)	**110.84**±2.04(1)
WFG3	**116.50**±0.00(4)	**116.50**±0.00 (1)	114.78±0.33(5)	**116.50**±0.00(1)	**116.50**±0.00(1)
WFG4	112.75±0.58(3)	103.34±3.61(5)	107.21±0.97(4)	113.46±0.35(2)	**114.02**±0.13(1)
WFG5	**112.19**±0.10(4)	112.21±0.03(3)	111.32±0.68(5)	**112.22**±0.00(1)	**112.22**±0.00(2)
WFG6	114.38±0.03(3)	113.79±0.10(4)	110.52±2.00(5)	**114.40**±0.00(1)	114.40±0.00(2)
WFG7	**114.40**±0.01(3)	114.37±0.03(4)	113.88±0.16(5)	**114.40**±0.00(2)	**114.40**±0.00(1)
WFG8	111.43±0.28(3)	110.57±0.81(4)	109.48±1.06(5)	111.70±0.23(2)	**111.82**±0.01(1)
WFG9	**111.46**±0.16(3)	107.54±1.10(4)	103.18±5.31(5)	**111.49**±0.03(2)	**111.51**±0.02(1)
Rank	3.44(3)	3.67(4)	4.67(5)	1.67(2)	1.22(1)

6 Discussion

A real-valued multi-objective (MO) hybrid algorithm was created by combining two uncrowded hypervolume (UHV) indicator-based algorithms via a dynamic resource allocation scheme. In Experiment 1 it was shown that for UHV optimization, picking an improvement metric is not trivial, as problem dependency has been observed. The results of experiment 1 however, also showed that if the hybrid is tasked to do UHV optimization, on average it benefits most from using the ΔBestUHV improvement metric, followed by CountBestUHVImproved. Both

metrics quantify the improvement of the best UHV, while the remaining metrics (Bosman2012, CountUHVImproved, ΔAverageUHV) measure the improvement over all solution sets. This opens the question why resource allocation towards the algorithms which improve fewer solution sets with higher UHV is preferable over the full runtime of the hybrid.

Experiment 2 has shown that concentrating gradient calls on a select number of solutions is preferred over diluting calls over the entire population. Distributing resources to the solutions with the top 3 highest UHV performed the best on average. Analysis on this distribution method however, has shown that during convergence, the hybrid frequently stalls due to an inaccurately estimated γ. Substantial improvement in convergence could be obtained by improving γ estimates at reinitialization of UHV-ADAM after executing UHV-GOMEA.

Experiment 2 also provided additional insight in the properties of UHV-ADAM. Figure 2 confirms that problems with few local optima (e.g. Convex sphere & Rosenbrock) can be solved by UHV-ADAM, while problems with many local optima (e.g. Convex sphere & Rastrigin) are not solvable.

One of the limitations of this work is that the problems that were used to tune the hybrid, all share the commonality of having a connected Pareto set. A connected Pareto set simplifies finding all other Pareto optimal solutions as soon as one solution has been determined. If one of the objectives then happens to be easily solvable (e.g. Sphere), it potentially creates situations where even algorithms that are not suited to solve multi-modal problems can still find the Pareto set by first solving the easy objective before moving over to the other objective, bypassing any local optimum. Future work should thus consider disconnected Pareto sets. Another limitation of this work, is that only a single EA, i.e. UHV-GOMEA, has been selected for hybridization. In [17], it was already observed that domination-based EA MO-RV-GOMEA [7] initially performs better than UHV-GOMEA. An even more efficient hybrid algorithm could potentially be created with MO-RV-GOMEA. However, as MO-RV-GOMEA is a domination-based EA, compatibility issues are likely to occur with UHV-based algorithms. Introducing a different EA could furthermore test the robustness of the RAS.

7 Conclusion

In this work, for the first time, a multi-objective optimization algorithm was introduced that hybridizes an uncrowded hypervolume-based (UHV) evolutionary algorithm with a UHV-based gradient algorithm via a dynamic resource allocation scheme (RAS). Experiments used to study the RAS showed that selecting a reward metric for the RAS is not trivial as it was observed that the best metric is problem-dependent. Experiments also showed that concentrating gradient steps on a select number of solutions of the population, outweighs dispersing gradient steps over the entire population. Implementations of the hybrid algorithm have also been compared to other UHV-based algorithms. It was shown that even if finite difference approximations are used to calculate gradients, it is still able to obtain competitive or better results than the original component algorithms as well as other UHV-based algorithms. We conclude that the resulting

hybrid is therefore a promising addition to the existing spectrum of evolutionary algorithms for multi-objective optimization.

References

1. Auger, A., Bader, J., Brockhoff, D., Zitzler, E.: Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point. In: Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms, pp. 87–102 (2009)
2. Berghammer, R., Friedrich, T., Neumann, F.: Convergence of set-based multi-objective optimization, indicators and deteriorative cycles. Theoret. Comput. Sci. **456**, 2–17 (2012)
3. Bosman, P.A.: On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization. IEEE Trans. Evol. Comput. **16**(1), 51–69 (2011)
4. Bosman, P.A., De Jong, E.D.: Combining gradient techniques for numerical multi-objective evolutionary optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 627–634 (2006)
5. Bosman, P.A., Grahl, J., Thierens, D.: Benchmarking parameter-free amalgam on functions with and without noise. Evol. Comput. **21**(3), 445–469 (2013)
6. Bouter, A., Alderliesten, T., Witteveen, C., Bosman, P.A.: Exploiting linkage information in real-valued optimization with the real-valued gene-pool optimal mixing evolutionary algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 705–712 (2017)
7. Bouter, A., Luong, N.H., Witteveen, C., Alderliesten, T., Bosman, P.A.: The multi-objective real-valued gene-pool optimal mixing evolutionary algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 537–544 (2017)
8. Deb, K., Kalyanmoy, D.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons Inc., USA (2001)
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
10. Deist, T.M., Maree, S.C., Alderliesten, T., Bosman, P.A.N.: Multi-objective optimization by uncrowded hypervolume gradient ascent. In: Bäck, T., et al. (eds.) PPSN 2020. LNCS, vol. 12270, pp. 186–200. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58115-2_13
11. Emmerich, M., Deutz, A.: Time complexity and zeros of the hypervolume indicator gradient field. In: EVOLVE-a Bridge Between Probability, Set Oriented Numerics, And Evolutionary Computation III, pp. 169–193. Springer (2014). https://doi.org/10.1007/978-3-319-01460-9_8
12. Emmerich, M., Deutz, A., Beume, N.: Gradient-based/evolutionary relay hybrid for computing pareto front approximations maximizing the S-metric. In: Bartz-Beielstein, T., et al. (eds.) HM 2007. LNCS, vol. 4771, pp. 140–156. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75514-2_11
13. Hernández, V.A.S., Schütze, O., Wang, H., Deutz, A., Emmerich, M.: The set-based hypervolume newton method for bi-objective optimization. IEEE Trans. Cybern. **50**(5), 2186–2196 (2018)
14. Hoffmeister, F., Bäck, T.: Genetic algorithms and evolution strategies: similarities and differences. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 455–469. Springer, Heidelberg (1991). https://doi.org/10.1007/BFb0029787

15. Huband, S., Barone, L., While, L., Hingston, P.: A scalable multi-objective test problem toolkit. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 280–295. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_20
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
17. Maree, S.C., Alderliesten, T., Bosman, P.A.: Uncrowded hypervolume-based multi-objective optimization with gene-pool optimal mixing. Evolutionary Comput. 1–24 (2021)
18. Sharma, S., Blank, J., Deb, K., Panigrahi, B.K.: Ensembled crossover based evolutionary algorithm for single and multi-objective optimization. In: 2021 IEEE Congress on Evolutionary Computation (CEC), pp. 1439–1446. IEEE (2021)
19. Touré, C., Hansen, N., Auger, A., Brockhoff, D.: Uncrowded hypervolume improvement: Como-cma-es and the sofomore framework. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 638–646 (2019)
20. Wang, H., Deutz, A., Bäck, T., Emmerich, M.: Hypervolume indicator gradient ascent multi-objective optimization. In: Trautmann, H., et al. (eds.) EMO 2017. LNCS, vol. 10173, pp. 654–669. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54157-0_44
21. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)