Issues with Performance Measures for Dynamic Multi-objective Optimisation

Mardé Helbig
CSIR: Meraka Institute
0184, Brummeria
South Africa
Email: mhelbig@csir.co.za

Andries P. Engelbrecht
University of Pretoria
Computer Science Department
South Africa
Email: engel@cs.up.ac.za

***Abstract*—In recent years a number of algorithms were proposed to solve dynamic multi-objective optimisation problems. However, a major problem in the field of dynamic multi-objective optimisation is a lack of standard performance measures to quantify the quality of solutions found by an algorithm. In addition, the selection of performance measures may lead to misleading results. This paper highlights issues that may cause misleading results when comparing dynamic multi-objective optimisation algorithms with performance measures that are currently used in the field.**

Keywords: dynamic multi-objective optimisation, performance measures

I. Introduction

In recent years various types of computational intelligence (CI) algorithms were proposed to solve dynamic multi-objective optimisation problems (DMOOPs). These algorithms can be categorised as evolutionary algorithms [1], [3], [4], [17], [21], [28], [26], particle swarm optimisation algorithms [6], [9], [13], [14], new CI algorithms proposed for dynamic multi-objective optimisation (DMOO) [5], [26] (for example membrane computing or P-systems [19]), approaches that transform a DMOOP into various single-objective optimisation problems (SOOPs) [16], [18], [25] and prediction-based algorithms [7], [12], [16], [23].

However, a major problem in the field of DMOO is a lack of standard performance measures. The set of performance measures chosen to quantify the quality of solutions found by various dynamic multi-objective optimisation algorithms (DMOAs) influences the results and effectiveness of comparative studies. Many performance measures used for DMOO are static multi-objective optimisation (MOO) performance measures that were adapted for DMOO. However, some of these performance measures may lead to misleading results when used for DMOO. This paper highlights issues with performance measures that are currently used in the field of DMOO that may cause misleading results.

The rest of the paper is outlined as follows: Section II presents formal definitions of concepts that are required as background for this paper. A short overview of MOO and DMOO performance measures are also presented. Issues with current DMOO performance measures are presented in Section III. Finally, the conclusions are discussed in Section IV.

II. Background

This section provides definitions of MOO and DMOO concepts required as background for the rest of the paper. In addition, it provides a short overview of performance measures used for DMOO. It should be noted that, due to space limitations, this paper does not discuss DMOAs, DMOOPs and DMOO performance measures in detail. However, the reader is referred to [8] for a comprehensive overview of DMOAs, DMOOPs and DMOO performance measures.

A. Formal Definitions

This section presents formal definitions with regards to MOO and DMOO.

Multi-objective Optimisation

Normally, improvement in one objective of a multi-objective optimisation problem (MOOP) leads to a worse solution for at least one other objective. Therefore, the definition of optimality used for SOOPs has to be adjusted when solving MOOPs.

A non-dominated solution is considered an optimal solution of a MOOP. Therefore, when one decision vector dominates another, the dominating decision vector is considered the better decision vector.

Let the n_x-dimensional search space (also referred to as the *decision space*) be represented by $S \subseteq \mathbb{R}^{n_x}$ and the feasible space represented by $F \subseteq S$, where $F = S$ for unconstrained optimisation problems. Let $\mathbf{x} = (x_1, x_2, \ldots, x_{n_x}) \in S$ represent a vector of the decision variables, i.e. the *decision vector*, and let a single objective function be defined as $f_k \colon \mathbb{R}^{n_x} \to \mathbb{R}$. Then $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_{n_k}(\mathbf{x})) \in O \subseteq \mathbb{R}^{n_k}$ represents an *objective vector* containing n_k objective function evaluations, and O is the *objective space*.

Using the notation above, vector domination is defined as follows:

Definition 1. Vector Domination: Let f_k be an objective function. Then, a decision vector $\mathbf{x}_1$ dominates another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \prec \mathbf{x}_2$, if and only if

- $f_k(\mathbf{x}_1) \leq f_k(\mathbf{x}_2)$, $\forall k = 1, \ldots, n_k$, i.e. $\mathbf{x}_1$ is at least as good as $\mathbf{x}_2$ for all the objectives; and
- $\exists i = 1, \ldots, n_k \colon f_i(\mathbf{x_1}) < f_i(\mathbf{x_2})$, i.e. $\mathbf{x}_1$ is strictly better than $\mathbf{x}_2$ for at least one objective.

The term Pareto-optimal is used to refer to the best decision vectors. Pareto-optimality is defined as:

Definition 2. Pareto-optimal: A decision vector $\mathbf{x}^*$ is Pareto-optimal if there does not exist a decision vector $\mathbf{x} \neq \mathbf{x}^* \in F$ that dominates it, i.e. $\nexists k$: $f_k(\mathbf{x}) \prec f_k(\mathbf{x}^*)$. If $\mathbf{x}^*$ is Pareto-optimal, the objective vector, $\mathbf{f}(\mathbf{x}^*)$, is also Pareto-optimal.

The set of all the Pareto-optimal decision vectors is referred to as the Pareto-optimal set (POS), defined as:

Definition 3. Pareto-optimal Set: The POS is formed by the set of all Pareto-optimal decision vectors, i.e.

$$POS = \{\mathbf{x}^* \in F \,|\, \nexists \mathbf{x} \in F\colon \mathbf{x} \prec \mathbf{x}^*\} \tag{1}$$

The best trade-off solutions for the MOOP are contained in the POS. The set of corresponding objective vectors are referred to as the Pareto-optimal front (POF), defined as follows:

Definition 4. Pareto-optimal Front: For the objective vector $\mathbf{f}(\mathbf{x})$ and the POS, the POF, $POF \subseteq O$, is defined as

$$POF = \{\mathbf{f} = (f_1(\mathbf{x}^*), f_2(\mathbf{x}^*), \ldots, f_{n_km}(\mathbf{x}^*)) \,|\, \mathbf{x}^* \in POS\} \tag{2}$$

Dynamic Multi-objective Optimisation

Using the notation defined above for MOO, an unconstrained DMOOP is defined as:

$$\begin{aligned} &minimise: && \mathbf{f}(\mathbf{x}, \mathbf{W}(t)) \\ &subject\ to: && \mathbf{x} \in [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x} \end{aligned} \tag{3}$$

where $\mathbf{W}(t)$ is a matrix of time-dependent control parameters of an objective function at time t, n_x is the number of decision variables, $\mathbf{x} = (x_1, \ldots, x_{n_x}) \in \mathbb{R}^{n_x}$ and $\mathbf{x} \in [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x}$ refers to the boundary constraints.

When solving a DMOOP the goal of an algorithm is to track the POF over time, i.e. for each time step, to find

$$\begin{aligned} POF(t) = \{\mathbf{f}(t) = (&f_1(\mathbf{x}^*, \mathbf{w}_1(t)), f_2(\mathbf{x}^*, \mathbf{w}_2(t)), \ldots, \\ &f_{n_k}(\mathbf{x}^*, \mathbf{w}_{n_k}(t))) \,|\, \mathbf{x}^* \in POS(t)\} \end{aligned} \tag{4}$$

B. DMOO Performance Measures

This section provides a short overview of the DMOO performance measures that are required as background for the rest of the paper.

Generational Distance and Variational Distance

The generational distance (GD) measures the convergence of the approximated POF (POF^*) towards the true POF (POF'). Unless stated otherwise, throughout this article POF' refers to a sampled set of the true POF.

Goh and Tan [5] adapted GD for DMOO as follows:

$$VD = \frac{1}{\tau}\sum_{t=1}^{\tau} VD(\tau)I(t),$$

with

$$VD(\tau) = \frac{\sqrt{n_{POS^*}\sum_{i=1}^{n_{POS^*}} d_i^2\;(\tau\%\,\tau_t)}}{n_{POS^*}}, \text{ and}$$

$$I(t) = \begin{cases} 1, & \text{if } \tau\%\tau_t = 0 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where n_{POS^*} is the number of solutions in the approximated POS (POS^*) and d_i is the Euclidean distance in the decision space between solution i of POS^* and the nearest member of the true POS (POS'), τ is the current iteration number, τ_t is the frequency of change, and $\%$ is the modulus operator. The performance measure, referred to as variational distance (VD), is calculated in the decision space every iteration just before a change in the environment occurs. Similar to GD, a low VD value indicates a good POS^*.

Hypervolume

The hypervolume (HV) or S-metric measures the volume of the objective space that is dominated by a non-dominated set [30], [31].

The reference vector used to calculate the HV can be any vector outside the feasible objective space. This selection of the reference vector results in a non-negative value for all possible non-dominated sets in the feasible objective space. Usually, the reference vector is selected as the vector that consists of the worst value for each objective of the union of all non-dominated solutions of all POF^* that are compared against each other. Since all of the non-dominated sets use the same reference vector, the selected reference vector will affect the ordering of the non-dominated sets that are compared against each other [11]. A high HV value indicates a good POF^*.

Hypervolume Ratio

To overcome the bias of the HV towards convex regions of the POF, Van Veldhuizen [24] proposed the hypervolume ratio (HVR) for MOO, defined for DMOO as follows:

$$HVR(t) = \frac{HV(POF^*(t))}{HV(POF'(t))} \tag{6}$$

The HVR normalises the HV and, assuming that the maximum HV is obtained by POF', the value of the HVR will be between 0 and 1. A high HVR indicates a good POF^*. For the calculation of HVR, the reference vector is selected as the worst objective values for each objective from the union of POF' and the non-dominated solutions of all POF^* that are compared against each other.

Maximum Spread Measure

A measure of maximum spread (MS) for MOO that measures the length of the diagonal of the hyperbox that is created by the extreme function values of the non-dominated set was introduced by Zitzler [29]. For DMOO MS is defined as:

$$MS(t) = \sqrt{\sum_{k=1}^{n_k} \left(\overline{POF_k^*(t)} - \underline{POF_k^*(t)}\right)^2} \quad (7)$$

where $\overline{POF_k^*}$ and $\underline{POF_k^*}$ are the maximum and minimum value of the k-th objective in POF^* respectively. A high MS value indicates a good extend (or spread) of solutions.

Adapted Maximum Spread Measure
An adapted version of MS, MS', that measures how well POF^* covers POF' was introduced by Goh and Tan [5]. Contrary to MS, MS' takes into account the proximity of POF^* to POF'. For DMOO MS' is defined as follows:

$$MS'(t) = \sqrt{\frac{1}{n_k}\sum_{k=1}^{n_k}\left[\frac{m(t)}{\overline{POF_k'(t)} - \underline{POF_k'(t)}}\right]^2},$$

with

$$m(t) = \min[\overline{POF_k^*(t)}, \overline{POF_k'(t)}] - \max[\underline{POF_k^*(t)}, \underline{POF_k'(t)}] \quad (8)$$

Accuracy Measure
Cámara *et al.* [22] adapted an dynamic single-objective optimisation (DSOO) accuracy measure introduced by Weicker [27] for DMOO. The alternative accuracy measure (acc_{alt}) is defined as:

$$acc_{alt}(t) = |HV(POF'(t)) - HV(POF^*(t))| \quad (9)$$

where $acc_{alt}(t)$ is the absolute difference between the HV of POF' and the HV of POF^* at time t. The absolute values ensures that $acc_{alt}(t) \geq 0$, even if $HV(POF^*) > HV(POF')$ (refer to Section III-A).

Stability Measure
Weicker [27] introduced a measure of stability for DSOO that quantifies the effect of the changes in the environment on the accuracy (acc) of the algorithm. Cámara *et al.* [22] adapted stability for DMOO as follows:

$$stab(t) = \max\{0, acc(t-1) - acc(t)\} \quad (10)$$

where a low $stab$ value indicates good performance.

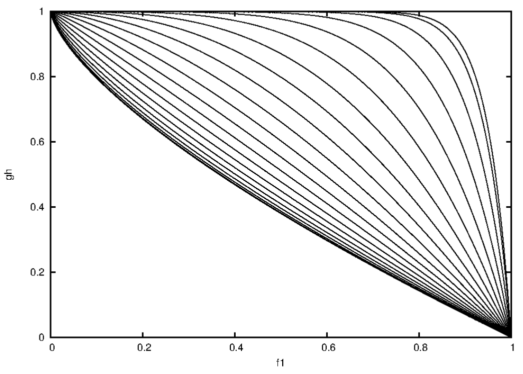
III. Issues with Current Performance Measures

This section discusses issues with current DMOO performance measures. Issues arise when an algorithm loses track of the changing POF as discussed in Section III-A, when there are outliers in the found or approximated POF (POF^*) as discussed in Section III-B and when the performance of various DMOAs are compared with one another as discussed in Section III-C. Throughout this section, unless stated otherwise, POF' and POF^* refer to $POF'(t)$ and $POF^*(t)$ for a specific time t respectively.

A. Losing Track of the Pareto-optimal Front

The goals when solving a DMOOP is to find for each defined time step an approximated POF that is as close as possible to the true POF and to find a diverse set of solutions. Therefore, in order to solve a DMOOP, a DMOA has to track the POF over time. However, it may happen that a DMOA loses track of the changing POF over time. This can occur when a DMOA does not detect changes in the environment and therefore gets stuck in a previous POF. If a DMOA loses track of the changing POF, and POF' of a DMOOP changes over time in such a way that the HV [30], [31] value decreases over time, many of the current performance measures will cause misleading results.

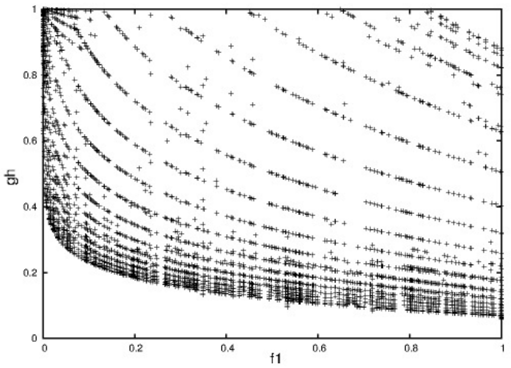
Figure 1 illustrates POF' for FDA2 [4] for 1000 iterations with the severity of change $n_t = 10$ and frequency of change $\tau_t = 10$. The POF of FDA2 changes in a cyclic manner over time, moving from the top line to the bottom line for certain time steps and from the bottom line to the top line for other time steps. When POF' moves from the bottom line to the top line (refer to Figure 1) over time, the HV of POF' decreases over time. Therefore, if a DMOA loses track of the changing POF during the cycle where the HV of POF' decreases over time, the HV value of the DMOA's POF^* will be greater than the HV value of POF', and therefore higher than the HV value of a DMOA that tracks the changing POF'.



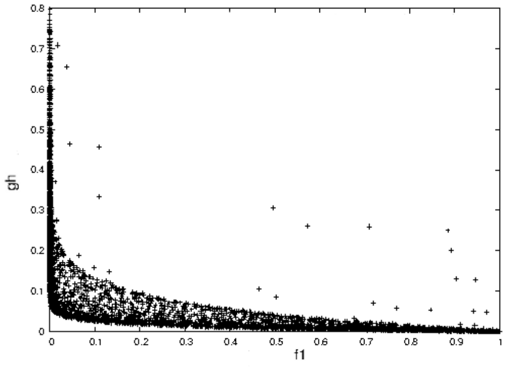
(a) POF' of FDA2

Fig. 1: POF' for FDA2 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

An example of a DMOA losing track of POF' is illustrated in Figure 2. Figure 2 illustrates POF^*s found for FDA2 [4] by two DMOAs, namely the dynamic non-dominated sorting genetic algorithm II-A (DNSGA-II-A) [3] and the dynamic vector evaluated particle swarm optimisation (DVEPSO) algorithm [10]. In Figure 2, DNSGA-II-A lost track of the changing POF (refer to Figure 2(b)) and DVEPSO tracked the changing POF more accurately than DNSGA-II-A (refer to Figure 2(a)). The average performance measure values for these two POF^*s are presented in Table I. In Table I, HVR refers to the HV ratio [15], acc and $stab$ refer to measures of accuracy and stability presented by Cámara *et al.* [2], and VD and MS' refer to the adapted generational distance and maximum spread performance measures for DMOO proposed by Goh and Tan [5]. However, in this study VD was measured in the objective space.



(a) POF^* found by DVEPSO



(b) POF^* found by DNSGA-II-A

Fig. 2: POF^* found by various DMOAs for FDA2 with $n_t = 10$, $\tau_t = 10$ for 1000 iterations

TABLE I
PERFORMANCE MEASURE VALUES FOR FDA2

Algorithm	HVR	acc	stab	VD	MS′
DVEPSO	0.99905	0.98157	0.00029	**0.43234**	**0.88916**
DNSGAII-A	**1.0044**	**0.98681**	**9.565x10^{-06}**	0.71581	0.77096

From Table I, it is interesting to observe that the performance measures VD and MS' indicate that DVEPSO performed better than DNSGA-II-A. However, the measures that make use of the HV (HVR, acc and $stab$) rank DNSGA-II-A's performance as being better than DVEPSO. The reason for these contradicting results is that DNSGA-II-A lost track of POF', while DVEPSO kept on tracking the changing POF' over time (refer to Figure 2). In this case, where one DMOA lost track of the POF, using a performance measure that uses the HV (such as acc and $stab$) will wrongly indicate the POF^* of a DMOA that lost track of the changing POF as the better set. Using the HVR will not necessarily address the problem. For static MOOPs it is assumed that HVR will always be less or equal to one. However, when solving DMOOPs, DMOAs that lose track of the changing POF may obtain HVR values greater than one (since the HV value of their POF^* is higher than the HV of POF'), while the DMOAs that successfully track the changing POF will obtain HVR values that are less or equal to one.

The following are suggested to overcome the problem of DMOAs losing track of POF':

- If POF' is known, acc_{alt} proposed by Cámara *et al.* [2] (refer to Equation (9)) should be used. Furthermore, since the choice of the performance measure used to measure acc influences the reliability of $stab$ [2] (refer to Equation (10)), using acc_{alt} for acc in Equation (10), will ensure that $stab$ will also be reliable even if a DMOA loses track of POF'.
- If POF' is unknown, as is the case with most real-world problems, acc_{alt} cannot be used, since the HV of POF' is unknown. In this situation the deviation of the performance measures that use the HV should be calculated. If some of the DMOAs' performance measure values have a deviation that is close to zero while the other DMOAs obtain performance measure values with a higher deviation, it may indicate that one or more of the DMOAs have lost track of the changing POF. Plotting and checking the graphs of POF^*s found by the various DMOAs can then be used to determine whether any DMOA lost track of POF'.

B. Outliers in the Pareto-optimal Front

In a frequently changing environment a DMOA has a limited number of generations or iterations to find non-dominated solutions for the current environment before a change occurs. Therefore, the POF^* found by the DMOA may contain outlier solutions. POF^* will contain outliers when in the number of iterations or generations available to the DMOA to solve the specific DMOOP, the DMOA found solutions that are further away from POF', but did not have enough time to find solutions closer to POF' that dominate the outlier solutions. Figure 3 illustrates an example of a POF^* that contains outlier solutions.

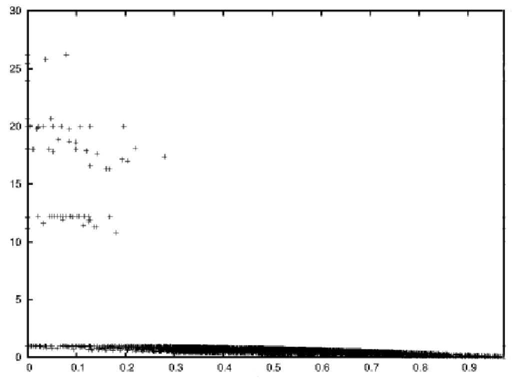

(a)

Fig. 3: Example of a POF^* for dMOP2 [5] that contains outlier solutions

Outliers in POF^* will skew the values of the following types of performance measures:

- distance-based performance measures, such as GD and VD;
- performance measures that measure the spread of the solutions, such as MS and MS'; and
- the HV performance measure.

The effect of outliers on these three types of performance measures are discussed below. For the discussion, an example POF^* and POF' are illustrated in Figure 4. In Figure 4, the sampled solutions of POF' are represented by triangles and the non-dominated solutions of POF^* are represented by stars.

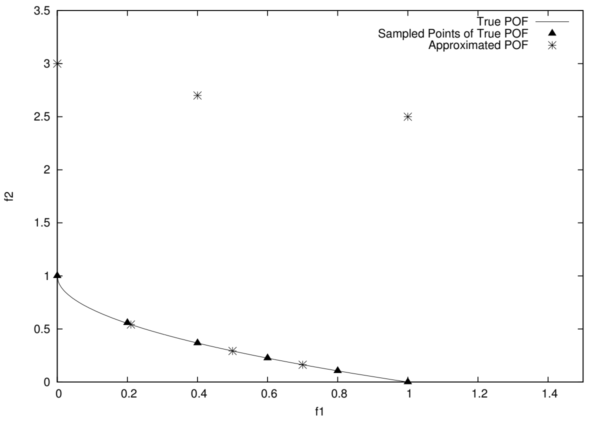


Fig. 4: POF^* of FDA1 [4] with outlier solutions

Distance-based performance measures

Distance-based performance measures calculate the distance between the found non-dominated solutions (POF^*) and POF'. If there are outlier solutions that are far away from POF', the distance between the outlier solutions and POF' will be large. Therefore, the distance-based performance measure will be much larger with outliers present compared to when the outliers are not present. GD and VD are calculated for the POF^* in Figure 4 and presented in Table II. The values in Table II confirm the influence of outliers on the GD and VD values. It should be noted that the severity of the influence of outlier solutions on the distance calculations depend on the number of outlier solutions and the distance of the outlier solutions from POF'.

TABLE II
GD AND VD VALUES FOR FDA1

Outliers	**GD**	**VD**
Yes	0.643504	1.576257
No	0.056536	0.097923

When DMOAs' POF^*s are compared using a distance-based performance measure, it may occur that if Algorithm A's POF^* contains solutions that are all reasonably close to POF' and Algorithm B's POF^* contains solutions that are mostly on POF' and only a few outliers, that Algorithm A's POF^* is ranked as being better than that of Algorithm B. One solution to manage outlier solutions is a recently proposed performance measure for MOO by Schütze *et al.* [20], namely the averaged Hausdorff distance measure. The averaged Hausdorff distance is calculated as:

$$\Delta_p(X,Y) = max\{GD_p(X,Y), IGD_p(X,Y)\} \qquad (11)$$

where $X = \{x_1, \ldots, x_n\}$, $Y\{y_1, \ldots, y_m\} \subset \mathbb{R}^{n_k}$ are finite non-empty sets, GD_p is the power mean of GD and IGD_p is the power mean of the inverse GD (IGD). The choice of the p-norm influences how much the outliers are penalised. When p is small, the averaging effect is higher and the influence of outliers are smaller. However, for larger values of p, the influence of outliers are larger and therefore outliers are penalised more [20]. In contrast to the GD and IGD values that are skewed by outliers, the averaged Hausdorff distance penalises outlier solutions and how much the outlier solutions are penalised is controlled by the value of p. However, knowledge about POF' is required for the calculation of the averaged Hausdorff distance measure.

Performance measures that measure the spread of the solutions

Performance measures that measure the spread of POF^* will produce skewed results in the presence of outliers. Figure 4 illustrates a POF^* for FDA1 [4] that contains outliers. In Figure 4 POF^* has f_1 values in the range of $[0, 1.0]$ and f_2 values in the range of $[0.16, 3.0]$ with the outlier solutions. Therefore, with the outlier solutions POF^* obtains a MS value of 3.0078 and a MS' value of 0.92195. However, without the outlier solutions POF^* has f_1 values in the range of $[0.21, 0.7]$ and f_2 values in the range of $[0.16, 0.54]$ with the outliers. Therefore, without outliers POF^* obtains a MS value of 0.62 and a MS' value of 0.4385. Both MS and MS' values are much larger when the outliers are taken into account. Therefore, the question arises whether including outliers in the spread calculation of POF^* leads to a true representation of the spread of POF^* or not.

The hypervolume

Many researchers use the HV to measure the performance of DMOAs, especially when POF' is unknown.

For POF^* in Figure 4, the reference vector used for the calcluation of the HV is $[1.1, 3.1]$. The HV values for POF^* in Figure 4 with and without the outlier solutions are presented in Table III.

TABLE III
HV, HVR AND ACC$_{alt}$ VALUES FOR FDA1

Outliers	**HV**	**HVR**	**acc$_{alt}$**
Yes	2.49898	0.84461	0.45974
No	2.47798	0.83752	0.48074

Table III indicates that POF^* obtained better HV, HVR and acc_{alt} values when the outlier solutions are taken into account. Therefore, outlier solutions may lead to misleading results and DMOAs being ranked incorrectly.

Managing outliers

One approach to manage outliers is to remove the outliers from POF^*. However, no consensus exists on the approach that should be followed to classify non-dominated solutions in POF^* as outliers. Furthermore, as the number of objectives increases, the number of solutions that are found by DMOAs that are non-dominated with regards to the other solutions in POF^* increases. Therefore, an increase in the number of objectives may lead to an increase in the number of outlier solutions in POF^*. In addition, when solving static MOOPs, outliers in POF^* will cause the same problems. However, the possibility of the occurrence of outliers increases with DMOO, since DMOAs generally have shorter time to converge towards POF' with DMOOPs than with MOOPs where the environment remains static.

C. Comparison of Algorithms' Performance

The performance of a newly proposed DMOA is typically compared with that of other DMOAs. However, when DMOAs are compared against one another it is important to check that all DMOAs manage boundary constraint violations. If a DMOA does not manage boundary constraint violations, its POF^* may contain infeasible solutions. The infeasible solutions may produce misleading performance measure values and may dominate optimal feasible solutions, preventing the feasible solutions from becoming part of POF^*.

Various performance measures are used when DMOAs are compared against one another. Normally, some DMOAs will perform really well with regards to certain performance measures and not so well with regards to the other performance measures. Typically, each DMOA is ranked according to its performance with regards to each performance measure and then the DMOA's average rank is calculated. The averaged ranks are then used to determine how well each DMOA performed with regards to the other DMOAs. This approach has the following flaws:

- if the wrong performance measures are selected, it may lead to incorrect ordering [8].
- calculating a DMOA's average value for a performance measure, does not provide information with regards to the DMOA's performance for the various environments during the run.

In stead of calculating a DMOA's average performance measure value and then ranking the DMOAs for that specific performance measure, the following approach is suggested:

1) Calculate the average performance measure value for each time step just before a change occured, over the number of runs, for each DMOA.
2) Perform a Kruskal-Wallis test on the performance measure values to determine whether there is a statistical significant difference between the DMOAs' performance.
3) If there is a statistical significant difference, perform pair-wise Mann-Whitney U tests to determine whether there is a a statistical significant difference between the two DMOAs' performance. If there is a statistical significant difference, award a win to the DMOA with the best average performance measure value and a loss to other DMOA.
4) Calculate for each DMOA the difference between its number of wins and number of losses ($Diff = \#wins - \#losses$)
5) Rank the DMOAs based on $Diff$.

This approach will indicate which DMOAs produced solutions that are statistical significantly better than the solutions obtained by other DMOAs. In addition,

- if a DMOA obtained more losses than wins, the DMOA did not perform well when compared against the other DMOAs. The smaller the value of $Diff$, the poorer the DMOA performed.
- if a DMOA obtained more wins than losses, it performed well when compared against the other DMOAs. The higher the value of $Diff$, the better the DMOA performed.

When solving DMOOPs, $Diff$ can be calculated:

- per performance measure over all DMOOPs.
- per environment type (for example per n_t-τ_t combination when using the FDA functions [4]) over all DMOOPs.
- over all performance measures and all DMOOPs.
- per DMOOP type:
  - per performance measure over all DMOOPs of the specific DMOOP type.
  - per environment type (for example per n_t-τ_t combination when using the FDA functions [4]) over all DMOOPs of the specific DMOOP type.
  - over all performance measures and all DMOOPs of the specific DMOOP type.

By analysing the results as indicated above, information is gained about the DMOO DMOAs' performance with regards to various environments, various performance measures and various DMOOP types. A comprehensive study comparing the performance of five DMOAs using the above mentioned approach can be found in [8]. The five DMOAs are the DNSGA-II-A [3], the DNSGA-II-B (DNSGA-II-B) [3], the dynamic cooperative-competitive evolutionary algorithm (dCOEA) [5], the dynamic multi-objective particle swarm optimisation (DMOPSO) DMOA [13] and the DVEPSO DMOA [6]. A small extract from the study is presented in Tables IV and V. In Tables IV and V, acc and $stab$ refer to the alternative accuracy measure and the stability measure proposed by Cámara *et al.* [2] and NS refers to the number of non-dominated solutions in POF^*. Table IV presents the overall performance of the five DMOAs over 18 DMOOPs and five different environments (with a varying frequency and severity of change).

The following observations are made from the results:

- DMOPSO obtained the best performance for acc and $stab$, but ranked the worst for NS.
- DNSGA-II-B performed the best for NS.
- The worst rank for acc and $stab$ was obtained by dCOEA. dCOEA was awarded much more losses than wins for

TABLE IV
OVERALL WINS AND LOSSES FOR VARIOUS PERFORMANCE MEASURES

PM	**Results**	**DMOO Algorithm**				
		DNSGA-II-A	**DNSGA-II-B**	**dCOEA**	**DMOPSO**	**DVEPSO**
acc	Wins	94	109	129	118	119
acc	Losses	120	104	164	86	95
acc	Diff	-26	5	-35	32	24
acc	Rank	4	3	5	**1**	2
$stab$	Wins	67	94	39	117	96
$stab$	Losses	89	66	200	39	50
$stab$	Diff	-22	28	-161	78	46
$stab$	Rank	4	3	5	**1**	2
NS	Wins	185	187	116	53	111
NS	Losses	83	78	202	195	129
NS	Diff	102	109	-86	-142	-18
NS	Rank	2	**1**	4	5	3

TABLE V
OVERALL WINS AND LOSSES SOLVING TYPE II DMOOPS FOR VARIOUS PERFORMANCE MEASURES

$\mathbf{n_t}$	τ_t	**PM**	**Results**	**DMOO Algorithm**				
				DNSGA-II-A	**DNSGA-II-B**	**dCOEA**	**DMOPSO**	**DVEPSO**
all	all	acc	Wins	55	55	36	56	63
all	all	acc	Losses	43	41	106	41	34
all	all	acc	Diff	12	14	-70	15	29
all	all	acc	Rank	4	3	5	2	**1**
all	all	$stab$	Wins	36	45	18	53	59
all	all	$stab$	Losses	43	30	104	20	14
all	all	$stab$	Diff	-7	15	-86	33	45
all	all	$stab$	Rank	4	3	5	2	**1**
all	all	NS	Wins	96	92	60	31	90
all	all	NS	Losses	48	50	116	105	50
all	all	NS	Diff	48	42	-56	-74	40
all	all	NS	Rank	**1**	2	4	5	3

$stab$ and therefore performed really poor with regards to $stab$ when compared against the performance of the other DMOAs.

- DNSGA-II-A and dCOEA were awarded more losses than wins for both acc and $stab$. In addition, dCOEA also obtained more losses than wins for NS. Therefore, dCOEA did not perform well with regards to any performance measure and DNSGA-II-A only performed well with regards to NS.
- Both DNSGA-II DMOAs obtained more wins than losses for NS. All other DMOAs obtained more losses than wins for NS. Therefore, the DNSGA-II DMOAs outperformed the other DMOAs with regards to NS. Furthermore, dCOEA and DMOPSO performed really poor with regards to NS, obtaining much more losses than wins for NS.

The wins and losses for Type II DMOOPs obtained by the DMOAs over all environment types are presented in Table V. The following are observed from the resutls:

- DVEPSO performed the best with regards to acc and $stab$. The best performance with regards to NS was obtained by DNSGA-II-A.
- The worst performance for both acc and $stab$ was obtained by dCOEA. For NS, DMOPSO performed the worst.
- dCOEA performed poorly with regards to acc, being awarded much more losses than wins and being the only DMOA that obtained more losses than wins.
- For $stab$, both DNSGA-II-A and dCOEA obtained more losses than wins. However, DNSGA-II-A was awarded only 7 more losses than wins, while dCOEA was awarded 86 more losses than wins. Therefore, dCOEA performed really poor with regards to $stab$ and its performance was affected more by changes in the environment than the other DMOAs.
- Only DMOPSO and dCOEA obtained more losses than wins for NS. All other DMOAs performed well, obtaining more wins than losses for NS. Therefore, DMOPSO and dCOEA performed poorly with regards to NS.

IV. CONCLUSION

This paper discussed issues with performance measures that are currently used to measure the performance of DMOAs. These issues occur when DMOAs lose track of the changing POF, when there are outlier solutions in the approximated POF (POF^*) and when DMOAs are compared against one another.

The effect of DMOAs losing track of the changing POF on the values of the hypervolume performance measures was discussed. It was illustrated how the value of the HV may wrongly indicate an algorithm's approximated POF as the better set of solutions when the algorithm lost track of the changing POF. If the true POF is known, acc_{alt} can be used, since it correctly classifies the best POF^*, even if DMOAs lose track of the changing POF. However, more research is required to determine which performance measures to use when the true POF is unknown.

Outliers are present in POF^* when the algorithm does not find solutions closer to the true POF that dominate the outliers in the time that is available to the algorithm, i.e. before a change in the environment occurs. Outliers will skew the results of distance-based performance measures, performance measures that measure the spread or extent of POF^* and the HV. One way of dealing with outliers is removing the outlier solutions from POF^*. However, identifying which solutions in POF^* can be classified as outliers is not a trivial task.

When comparing the performance of various DMOAs with one another, calculating the average performance measure value and then ranking the DMOAs according to the average value does not provide any information with regards to how well the algorithm tracked the changing POF over time. Therefore, a new approach using statistical analysis was proposed.

References

[1] Z. Avdagić, S. Konjicija, and S. Omanović, *Evolutionary approach to solving non-stationary dynamic multi-objective problems*, Foundations of Computational Intelligence Volume 3 (A. Abraham, A-E. Hassanien, P. Siarry, and A. Engelbrecht, eds.), Studies in Computational Intelligence, vol. 203, Springer Berlin/Heidelberg, 2009, pp. 267–289.

[2] M. Cámara, J. Ortega, and F.J. de Toro, *Parallel processing for multi-objective optimization in dynamic environments*, International Parallel and Distributed Processing Symposium **0** (2007), 243–250.

[3] K. Deb, U.B. Rao N., and S. Karthik, *Dynamic multi-objective optimization and decision-making using modied nsga-ii: a case study on hydro-thermal power scheduling*, Proceedings of International Conference on Evolutionary Multi-criterion optimization (Matsushima, Japan), 2007, pp. 803–817.

[4] M. Farina, K. Deb, and P. Amato, *Dynamic multiobjective optimization problems: test cases, approximations, and applications*, IEEE Transactions on Evolutionary Computation **8** (2004), no. 5, 425–442.

[5] C-K. Goh and K.C. Tan, *A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization*, IEEE Transactions on Evolutionary Computation **13** (2009), no. 1, 103–127.

[6] M. Greeff and A.P. Engelbrecht, *Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation*, Proceedings of World Congress on Computational Intelligence (WCCI): Congress on Evoluationary Computation (Hong Kong), june 2008, pp. 2917–2924.

[7] I. Hatzakis and D. Wallace, *Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach*, Proceedings of the Conference on Genetic and Evolutionary Computation (New York, NY, USA), ACM, 2006, pp. 1201–1208.

[8] M. Helbig, *Solving dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation*, Ph.D. thesis, University of Pretoria, 2012, Available online at: http://upetd.up.ac.za/thesis/available/etd-09242012-211127.

[9] M. Helbig and A.P. Engelbrecht, *Archive management for dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation*, Proceedings of Congress on Evolutionary Computation (New Orleans, U.S.A.), June 2011, pp. 2047–2054.

[10] ———, *Metaheuristics for dynamic optimization*, Studies in Computational Intelligence, vol. 433, ch. Dynamic multi-objective optimisation using PSO, pp. 147–188, Springer Verlag, 2013.

[11] J.D. Knowles, *Local-search and hybrid evolutionary algorithms for pareto optimisation*, Ph.D. thesis, Department of Computer Science The University of Reading, 2002.

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

[13] M.S. Lechuga, *Multi-objective optimisation using sharing in swarm optimisation algorithms*, Ph.D. thesis, University of Birmingham, July 2009.

[14] X. Li, *Better spread and convergence: particle swarm multiobjective optimization using the maximin fitness function*, Proceedings of Conference on Genetic and Evolutionary Computation (K. Deb, ed.), Lecture Notes in Computer Science, vol. 3102, Springer Berlin/Heidelberg, 2004, pp. 117–128.

[15] X. Li, J. Branke, and M. Kirley, *On performance metrics and particle swarm methods for dynamic multiobjective optimization problems*, Proceedings of Congress on Evolutionary Computation, sept. 2007, pp. 576–583.

[16] C-A. Liu and Y. Wang, *New evolutionary algorithm for dynamic multiobjective optimization problems*, Advances in Natural Computation (L. Jiao, L. Wang, X-B. Gao, J. Liu, and F. Wu, eds.), Lecture Notes in Computer Science, vol. 4221, Springer Berlin/Heidelberg, 2006, pp. 889–892.

[17] J. Mehnen, G. Rudolph, and T. Wagner, *Evolutionary optimization of dynamic multiobjective functions*, Tech. Report CI-204/06, Universität Dortmund, Universität Dortmund, Fachbereich Informatik/XI, 44221, Dortmund, Germany, May 2006.

[18] P. Moscato, *On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms*, Tech. report, California Institute of Technology, Pasadena, California, U.S.A., 1999.

[19] G. Păun, *Computing with membranes*, Journal of Computer and System Sciences **61** (2000), no. 1, 108–143.

[20] O. Schütze, X. Esquivel, A. Lara, and C.A. Coello Coello, *Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization*, IEEE Transactions on Evolutionary Computation **16** (2012), no. 4.

[21] R. Shang, L. Jiao, M. Gong, and B. Lu, *Clonal selection algorithm for dynamic multiobjective optimization*, Computational Intelligence and Security (Y. Hao, J. Liu, Y. Wang, Y-M. Cheung, H. Yin, L. Jiao, J. Ma, and Y-C. Jiao, eds.), Lecture Notes in Computer Science, vol. 3801, Springer Berlin/Heidelberg, 2005, pp. 846–851.

[22] M. Cámara Sola, *Parallel processing for dynamic multi-objective optimization*, Ph.D. thesis, Universidad de Granada, Dept. of Computer Architecture and Computer Technology, Universidad de Granada, Spain, April 2010.

[23] A.K.M. Talukder, A. Khaled, M. Kirley, and R. Buyya, *A pareto following variation operator for fast-converging multiobjective evolutionary algorithms*, Proceedings of the annual conference on Genetic and evolutionary computation (New York, NY, USA), ACM, 2008, pp. 721–728.

[24] D.A. van Veldhuizen, *Multiobjective evolutionary algorithms: classification, analyses, and new innovations*, Ph.D. thesis, Graduate School of Engineering Air University, 1999.

[25] Y. Wang and C. Dang, *An evolutionary algorithm for dynamic multi-objective optimization*, Applied Mathematics and Computation **25** (2008), 6–18.

[26] Y. Wang and B. Li, *Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization*, Memetic Computing **2** (2010), no. 1, 3–24.

[27] K. Weicker, *Performance measures for dynamic environments*, Parallel Problem Solving from Nature (J. Guervós, P. Adamidis, H-G. Beyer, H-P. Schwefel, and J-L. Fernández-Villaca nas, eds.), Lecture Notes in Computer Science, vol. 2439, Springer Berlin/Heidelberg, 2002, pp. 64–73.

[28] Z. Zhang, *Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control*, Applied Soft Computing **8** (2007), 959–971.

[29] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: methods and applications*, Ph.D. thesis, Swiss Federal Institute of Technology (ETH) Zurich Switzerland, 1999.

[30] E. Zitzler and L. Thiele, *Multiobjective optimization using evolutionary algorithms a comparative case study*, **1498** (1998), 292–301.

[31] ———, *Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach*, IEEE Transactions on Evolutionary Computation **3** (1999), no. 4, 257–271.