Automated and Surrogate Multi-Resolution Approaches in Genetic Algorithms

Abdulaziz T. Almutairi
Department of Computer Science
University of Exeter
Exeter, UK
aa685@exeter.ac.uk

Jonathan E. Fieldsend
Department of Computer Science
University of Exeter
Exeter, UK
J.E.Fieldsend@exeter.ac.uk

***Abstract*—Recent work on multi-resolution optimisation (varying the fidelity of a design during a search) has developed approaches for automated resolution change depending on the population characteristics. This used the standard deviation of the population, or the marginal probability density estimation per variable, to automatically determine the resolution to apply to a design in the next generation. Here we build on this methodology in a number of new directions. We investigate the use of a complete estimated probability density function for resolution determination, enabling the dependencies between variables to be represented. We also explore the use of the multi-resolution transformation to assign a surrogate fitness to population members, but without modifying their location, and discuss the fitness landscape implications of this approach. Results are presented on a range of popular uni-objective continuous test-functions. These demonstrate the performance improvements that can be gained using an automated multi-resolution approach, and surprisingly indicate the simplest resolution indicator is often the most effective, but that relative performance is often problem dependant. We also observe how population duplicates grow in multi-resolution approaches, and discuss the implications of this when comparing algorithms (and efficiently implementing them).**

***Index Terms*—evolutionary computation; genetic algorithms, multi-modal optimisation; multi-resolution; multi-scale**

I. Introduction

The search space in a traditional genetic algorithm (GA) optimisation of a problem is limited since the chromosome length is fixed *a priori* and usually unaltered during the optimisation process. One effect of this is the best fitness achievable is restricted to that in the fitness image of the subset of the domain that the particular resolution maps to. We may end up with solutions which are very close to the optimal, but most of the time due to the chromosome representation limitation, the optimum is unobtainable [1]. However, an obvious advantage of a chromosome representation rather than a real-coded representation is the search domain is greatly reduced, simplifying the optimisation process (especially for expensive optimisation problems). There is an obvious balance to be struck therefore between granularity of a representation (and therefore size of problem), and the quality attainable of the eventual solution. Multi-resolution algorithms, popular in many engineering design applications of genetic algorithms (see e.g. [2], [3]), tackle this issue by varying the resolution *during* an optimisation run. Variation of resolution is typically on a fixed schedule, or guided by the practitioner. However, recent work has begun to explore *automated* approaches to multi-resolution, which we investigate and extend here.

The main contributions of this work are:

1) Investigating the use of a full estimated probability density function when conducting automated multi-resolution search.
2) Exploring the use of automated multi-resolution to assign *surrogate* values to population members to encourage diversity maintenance.
3) Compensating for duplicate bias in assessment.
4) Empirical comparison of a range of automated multi-resolution approaches.

The paper proceeds as follows. In Section II we review some of the previous literature on multi-resolution optimisation, leading up to recent advances in automated multi-resolution. We follow this by discussing our extension to the approach, and additionally propose their use as surrogate fitness method in Section III. In Section IV we detail the experimental work and results, and the paper concludes with Section V.

II. Multi-resolution

Multi-resolution (also refereed to as multi-scale, or in some contexts multi-fidelity optimisation) is most regularly employed in engineering design optimisation problems, where tackling a problem immediately at the highest design resolution is infeasible due to the resultant search space size [4]–[8].

Figure 1 illustrates in a 2D domain how a resolution may change depending on the bit-string length used to represent two real values. Here the same bit-string length is used for both x_1 and x_2, however it is quite possible to use different resolutions for different design variables. Attainable designs lie on the vertices of the grid at the different resolutions — any designs on the interior of a cell being unobtainable as they cannot be represented at the displayed resolution.

The effect is for the search landscape to be discretised, with the position of minima/maxima (modes) shifting depending on the resolution used. In the illustration in Figure 1 at 2^1 and 2^2 resolutions there is an optima at $(1.0, 0.0)$. At 2^4 this has shifted to $(0.0\dot{6}, 0.9\dot{3})$. Additionally a mode at $(0.6\dot{6}, 0.6\dot{6})$ is observed at 2^2 and 2^4, but is unobservable at 2^1. The *granularity* g_i imposed by a resolution (the length of one grid

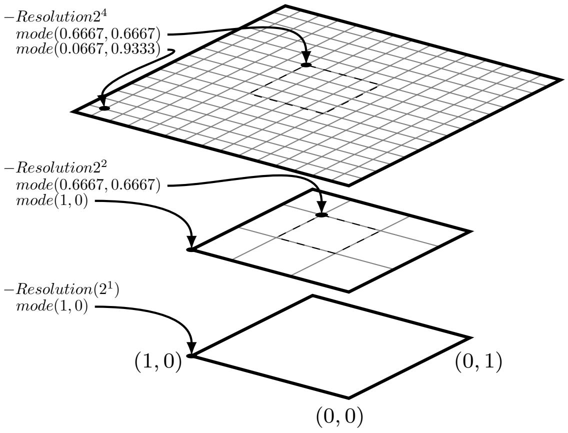

Fig. 1: Illustration of multiple resolutions in same domain (using bit-strings). Optima locations under different resolutions indicated with filled circles.

cell along one dimension) is determined, for the ith dimension, as:

$$g_i = \frac{upper\_bound_i - lower\_bound_i}{base^{resolution_i} - 1}. \tag{1}$$

Although bit-strings (i.e. $base = 2$) are often used to represent variables in standard GAs, there is an issue using this representation in the multi-resolution context. Using a larger resolution with a base 2 does not guarantee the new domain is a superset of the lower resolution domain (meaning high quality solutions in a lower resolution may be unobtainable at some higher resolutions). In the binary context the higher resolution power must be an integer multiple of the lower resolution power to guarantee all values represented by the lower resolution are present in the higher resolution (e.g. $2^1, 2^2, 2^4, 2^8, 2^{16}, \ldots$). As such, base 10 representations are more attractive for variable resolutions, as on each resolution increment (increasing r_i by 1), the search domain induced by the new resolution contains *all* designs representable by the previous resolution.

A. Prior work on adaptive multi-resolution genetic algorithms

Typically in multi-resolution optimisation the optimiser will start from a coarse resolution, and over time the scale will become finer, so that by the end of the optimisation process the search population will converge to high-fidelity design. Examples of work where improved GA performance has been demonstrated by varying the resolution during a search include [2], [3], [9]–[12].

In [2], [9] (semi-)independent island sub-populations are used, which after working independently for a few generations exchange their best solutions with each other. These sub-populations can be connected by using different topologies such as bi-directional rings, unidirectional rings, hypercube, tree structure, etc. In [9] three sub-populations where used, one employing a high resolution (fine grid) while the other two investigated a low resolution (coarse grid). In [2] each of the three populations where initialised separately, but rather than having distinct resolutions, each population began from low resolution with a defined percentage of the best solutions to be evaluated on the high resolution. They then exchanged their best solution at the fine grid after a number of generations by replacing the worst solutions, using the approach developed in [3].

Lopez Jaimes and Coello Coello proposed a novel multi-objective evolutionary algorithm based on the island model with heterogeneous nodes to obtain a run-time performance advantage from parallelisation [11]. Here each island (sub-population) had a different resolution and worked independently. They started from a coarse resolution, and gradually increase the resolution employed each generation until convergence.

Kim and Weck investigated how the computational run time of an optimisation process may be reduced by employing a variable length chromosome GA. The progressive refinement in topology proposed is effectively also a multi-resolution approach [10].

In [13] a control variable was used to set the length of the genes in their GA (the Multi-Scale Algorithm), for optimising the power flow in a power system network. The algorithm commences searching the entire feasible space with a large-scale search parameter (coarse resolution), with the resolution growing finer as the variable range decreases (as the population converges). Conversely, if a variable range in the population widens, the corresponding resolution coarsens

Most recently, research has been conducted on adaptive (automatic) multi-resolution, for multi-objective optimisation, using a single search population [14]. Here each solution's resolution on each variable can differ, via discretisation, with every other population member. This is achieved by using one of two different indicators which specify the resolution for any variable location at a particular generation (by discretising real-coded design-vectors). The two indicators investigated were the standard deviation (SD) and estimated probability density function (ePDF) — using kernel density estimation.

An algorithm describing the process is provided in Algorithm 1. On lines 2 and 7 designs are discretised before they are subsequently evaluated. The resolution used for this discretisation is either based on the standard deviation of each variable in the discretised set, or on an estimated probability density function.

In the first case, all designs have the same discretisation, determined by the decimal place d_i which is calculated as:

$$d_i = \left\lceil \left(1 - \frac{\sigma_i}{\sigma_{max}}\right)(d_{max} - d_{min}) + d_{min} \right\rceil \tag{2}$$

where d_{max} and d_{min} denote the highest and lowest resolutions to be considered, σ_i is the standard deviation of the ith variable, and σ_{max} is the standard deviation of the uniform distribution in the same space. When the values of the ith design variable are well spread out in the population, the d_i will be near its minimum, and as it becomes more tightly distributed the d_i will correspondingly increase.

Algorithm 1 Automated discretisation method of Kondo & Tatsukawa [14].

```
$P_0 \leftarrow$ initialise()    ▷ Get the initial population
$P_0 \leftarrow$ discretise($P_0$)    ▷ Discretise $P_0$ using its distribution
$E_0^P \leftarrow$ evaluate($P_0$)    ▷ Evaluate discretised designs
for each $t \leftarrow 0, ..., t_{max}$ do
    $Parents_t \leftarrow$ tournament($P_t, E_t^P$)    ▷ Tournament selection
    $Q_t \leftarrow$ evolve($Parents_t$)    ▷ Conduct crossover (SBX) and mutation (polynomial mutation)
    $Q_t \leftarrow$ discretise($Q_t$)    ▷ Discretise $Q_t$ using its distribution
    $E_t^Q \leftarrow$ evaluate($Q_t$)    ▷ Evaluate discretised
    $R_t \leftarrow P_t \cup Q_t$
    $E_t^R \leftarrow E_t^P \cup E_t^Q$
    $\{P_{t+1}, E_{t+1}^P\} \leftarrow$ select($R_t, E_t^R$)
return Final population
```

The discretisation using the ePDF in [14] enables the resolution to vary across designs, not just variables. Given the (marginal) estimated probability density value of individual j on design variable i, $\hat{f}(x_{i,j})$, the discretisation $d_{i,j}$ is calculated as:

$$d_i = \left\lceil \left(\frac{\hat{f}(x_{i,j})}{\max_{x \in [lower_i, upper_i]} \hat{f}(x)} \right) (d_{max} - d_{min}) + d_{min} \right\rceil \quad (3)$$

given the range of the ith variable being $lower_i$ to $upper_i$.[1]

III. Extensions to adaptive multi-resolution

We now outline our proposed extensions to the previous work on adaptive multi-resolution GAs, which we evaluate empirically in later sections.

A. Using a full ePDF for discretisation

Previous work has employed the (marginal) standard deviation of a variable across a population to fix its resolution for a particular individual, and also used the marginal estimated PDF [14].

Here we investigate using the full (complete) estimated PDF, and, as in [14], we employ a kernel density estimator for this. We are interested in using a full ePDF here as using a marginal variation measure obliterates any relationships *between* variables in a population. Consider the simple 2D illustration in Figure 2. The top panel shows a population denoted with black dots. The marginal ePDF approach would result in a design proposed around (-0.5,-0.5) having a high resolution applied, as both the marginal ePDFs (shown in the middle two panels) have high density for the corresponding values. However, a full ePDF (shown in the bottom-left panel) would assign it a very low resolution, as it is far from existing population members. The full ePDF has the benefit therefore of representing the density of combinations of variables, however, unlike the marginal ePDF approach it results in the same resolution being applied to all variables of a particular proposed design, rather than being able to vary the resolution for each design variable of a particular individual.

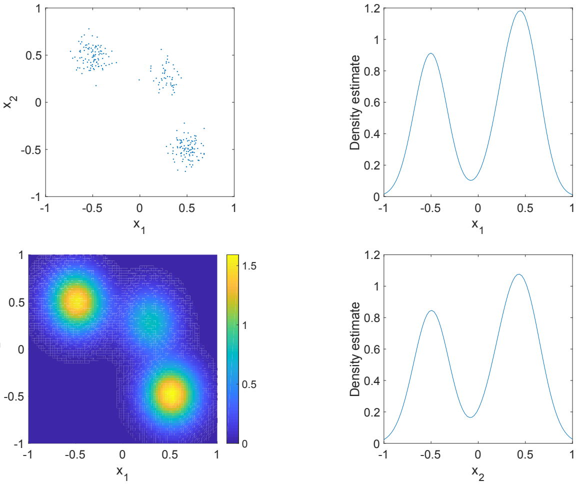

Fig. 2: Illustration of problems with marginal densities for automatic multi-resolution. Top-left: designs in 2D space. Right panels: marginal densities. Bottom-left: full ePDF.

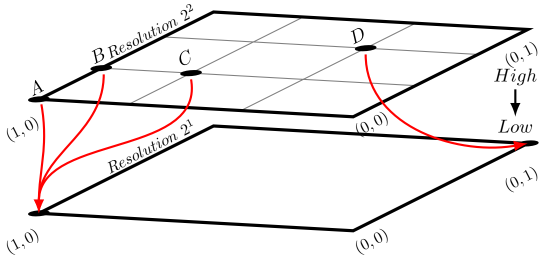

Fig. 3: Illustration the processes for duplication in multi-resolution. Change from high to low resolution.

The discretisation of the jth set member using the complete ePDF is calculated as:

$$d_j = \left\lceil \left(\frac{\hat{f}(\mathbf{x}_j)}{\max_{\mathbf{x} \in X} \hat{f}(\mathbf{x})} \right) (d_{max} - d_{min}) + d_{min} \right\rceil \quad (4)$$

Where X is the search domain.

B. Using multi-resolution mappings for surrogate fitnesses

One potential issue with using multi-resolution approaches is that diversity can be quickly lost when discretisation is applied, as solutions can only lie on the vertices on the grid induced by a discretisation. We now confront this problem using a surrogate fitness value.

Figures 3–4 illustrates the two main causes for diversity loss in a population. In Figure 3 this is due to the resolution change

[1] Reported as $\left\lceil \left(1 - \hat{f}(x_{i,j}) / \max_{x \in [lower_i, upper_i]} \hat{f}(x)\right) (d_{max} - d_{min}) + d_{max} \right\rceil$ in [14]. However, this appears to be an error, as when $\hat{f}(x_{i,j}) / \max_{x \in [lower_i, upper_i]} \hat{f}(x)$ is maximised the authors' infer the highest resolution occurs.

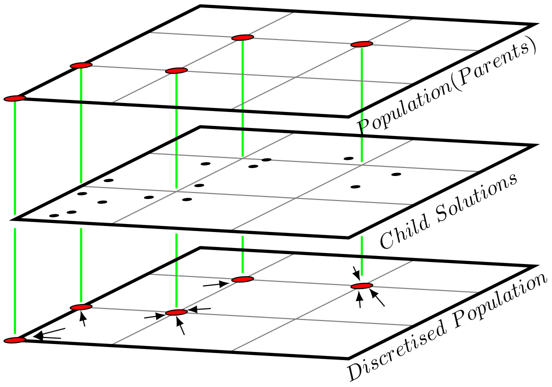

Fig. 4: Illustration of the duplication process in multi-resolution. Evolved solutions pulled to resolution grid vertices.

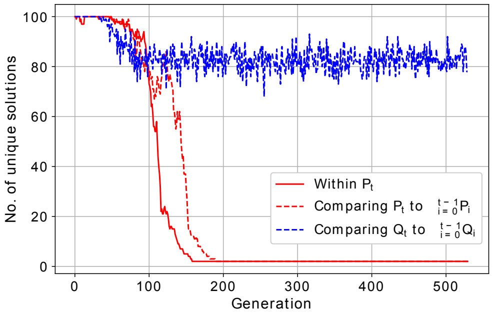

Fig. 5: Proportion of the parent population which is unique with respect to the other current parents (red line) and with respect to all previous parents (dashed red line). Proportion of the child population which is unique with respect to previous children (dotted blue line).

from high to low and the discretisation applied. In Figure 4 real-valued crossover and mutation may vary child solutions from their parents, but the subsequent discretisation can shift them to duplicated locations. If this isn't compensated for, a steady growth of duplicates in the population can additionally incur a wasted cost of duplicated solution evaluations.

Figure 5, shows the proportion of unique solutions that exist with respect to a population itself, and with respect to designs evaluated over time (search population ($|P| = 100$), using (2) when optimising the Schwefel function). Clearly the proportion of revisited child solutions will lead to *wasted* computation if a separate map is not maintained. We discuss this further in Section III-C.

As an alternative approach, we also propose to use adaptive multi-resolution to assign a *surrogate* fitness to designs. In this case the domain size is unchanged (and continuous) however the landscape observed by the optimiser varies according to the resolution induced by the surrogate discretisation. As such, diversity is maintained in the population between generations, and duplicated parents in the same population are avoided, however each location has an alternative *inferred* location, which depends on the resolution that it was evaluated under for each design variable. This approach is detailed in Algorithm 2. The key differences with Algorithm 1 occurs at the discretisation and fitness assignment stages, where a temporary discretised copy of a set is made (lines 2 and 8), whose evaluation is subsequently used as a surrogate for the non-discretised locations (lines 4 and 10).

Algorithm 2 Surrogate multi-resolution optimisation

```
1:  $P_0 \leftarrow \texttt{initialise}()$
2:  $P'_0 \leftarrow \texttt{discretise}(P_0)$  ▷ Temporary discretised copy of the population
3:  $E_0^{P'} \leftarrow \texttt{evaluate}(P'_0)$
4:  $E_0^{P} \leftarrow E_0^{P'}$  ▷ Assign fitness from discretised locations to population
5:  for each $t \leftarrow 0, ..., t_{max}$ do
6:      $Parents_t \leftarrow \texttt{tournament}(P_t, E_t^P)$  ▷ Tournament selection
7:      $Q_t \leftarrow \texttt{evolve}(Parents_t)$  ▷ Conduct crossover (SBX) and mutation (polynomial mutation)
8:      $Q'_t \leftarrow \texttt{discretise}(Q_t)$  ▷ Temporary discretised $Q_t$ using $Q_t$'s distribution
9:      $E_t^{Q'} \leftarrow \texttt{evaluate}(Q'_t)$
10:     $E_t^{Q} \leftarrow E_t^{Q'}$  ▷ Assign fitness from discretised locations to children
11:     $R_t \leftarrow P_t \cup Q_t$
12:     $E_t^R \leftarrow E_t^P \cup E_t^Q$
13:     $\{P_{t+1}, E_{t+1}^P\} \leftarrow \texttt{select}(R_t, E_t^R)$
    return Final population
```

C. Duplicates over time

Often the major cost in optimisation is querying the quality function for a design. As we identified in section III-B there tends to be an increased amount of duplication of proposed designs in the multi-resolution GA case. This is not just between parents selected at each generation, it is also for proposed children, in comparison to previously evaluated designs. As such, we may leverage this and run an optimiser for a maximum number of *unique* designs evaluated, rather than a fixed generation length, and exploit a map of previously evaluated designs to quality value when a duplicate design is proposed during the search. There is a risk if a population converges to a single point that all children would be duplicates repeatedly (as the map saturates with all possible mutations from this point). As such we also employ a heuristic where no new unique designs are proposed for n generations in a row search also terminates.

IV. Empirical results

We now conduct a set of experiments, comparing Algorithm 1 using the previously proposed (2) and (3) to a variant employing (4), and also our proposed surrogate multi-resolution Algorithm 2, using (2), (3) and (4) for discretisation. [2]

[2] Python code will be made available at the first author's `GitHub` repository. https://github.com/azeezx/ssciga

TABLE I: The benchmark problems used in this study.

Problem	Range [min,max]	Dimensionality
Schwefel	[-500,500]	2,5,10,50
Ackley	[-5.12,5.12]	2,5,10,50
Rastrigin	[-5.12,5.12]	2,5,10,50
Griewank	[-600,600]	2,5,10,50
CF1 1	[-5,5]	2,5,10,50
CF2 2	[-5,5]	2,5,10,50

We use a range of well-known uni-objective test problems [15]–[20], detailed in Table I. We utilise the test function implementations from the DEAP framework [21]. All six problems are assessed on four different design dimensions, $|\mathbf{x}| = \{2, 5, 10, 50\}$. The first four are minimisation problems, and the last two maximisation problems.

We apply the different variants of the genetic algorithm to each of the test problems 30 times, using the parameter setting based on the suggestions from previous work [14], specifically:

- Population size: 100.
- Termination condition: $50,000$ evaluation.
- Parent selection: tournament selection.
- Crossover: simulated binary crossover (SBX) $\eta = 30$. Probability of 1.
- Mutation: polynomial mutation $\eta = 20$. Probability of $1/(|\mathbf{x}|)$.
- d_{min}: 2 (base 10).
- d_{max}: 8 (base 10).
- Kernel function (KDE): Gaussian, bandwidth value $h = 0.05$.
- σ_{max}: $\sigma(lower\_bound, upper\_bound)$.

In each run all the methods start with the same population (i.e. are paired), so we used a non-parametric statistical hypothesis test (Wilcoxon signed-rank test) to assess if there is a significant difference between the results, with the Holm-Bonferronoi correction to compensate for multiple hypothesis being tested.

Figure 6 shows shows the average proportion of the optimisation run length that an algorithm is the median best performing, or not significantly different than the median best performing across all problems and problem dimensions. Broadly this appears to show that SD, SD with surrogate value and no multi-resolution are all fairly similar (within 5% performance), the eDPF variants being markedly lower. However, when we pull apart performance across problems and dimensions we see much larger variations, suggesting the interplay of problem type, problem dimension and algorithm are all important.

Figure 7 shows the proportion of run size that each algorithm is either the best, or not significantly different from the best, for each problem and problem size individually (first six rows and four columns) and averages across dimensions (final column), and across problems for a particular dimension size (bottom row). Interesting trends are readily apparent. Although in general not well-performing, the Full ePDF approach is consistently the best approach to take for the Schwefel function, with a 100% score on all dimension sizes (though at 50D many approaches become statistically indistinguishable). For the other problems SD (Ackley, Griewank, CF2) or SD with surrogates (Rastrigin, CF1) tend to be the best, although for CF2 ePDF tends to do better for the lower dimensional variants. Not using a multi-resolution approach is also a good choice for Rastrigin and Griewak, and across problems at 50D. However, if we look at the convergence plots in Figure 8 we see this latter property is may be simply because many of the problems are still poorly converged at this large problem size even after 50 000 function evaluations.

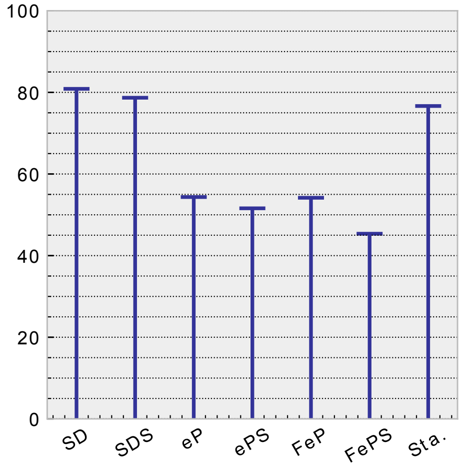

Fig. 6: Proportion of run-length that each optimiser is not significantly worse than the median best performing — averaged across all problems and problem sizes. SDS – SD with surrogate; ePS – ePDF with surrogate; FeP – Full ePDF; FePS – Full ePDF with surrogate; Sta. – standard (real valued with no multi-resolution).

Table II summarises the results across problems and problem dimensions at the run end. Interesting the simple standard deviation-based update mechanism is consistently well-performing across problems and dimensions on all bar Schwefel, having the lowest median, or performance not significantly different from the algorithm with the lowest median, on 16 out of the 24 problem instances, with the surrogate version a marginally better 17 out of 24. Using a real-encoding with no multi-resolution scores 11 out of 24 on this measure.

Perhaps the most surprising result is the performance of the standard deviation-based approach on the composite functions 1 and 2, which we believed *a priori* it would find most taxing. However, as the basins of attraction for the maxima can vary considerably in scale on these problems, our conjecture is that it simply does not locate the narrow optima and the population drawn into a larger basin where it can the increase its fidelity and still achieve near optimal results. Indeed, the Schewfel function where the basins of attraction are all of similar size is the only problem where the standard deviation-based approach does struggle compared to the estimated PDF approach.

On the whole the marginal and full estimated PDF ap-

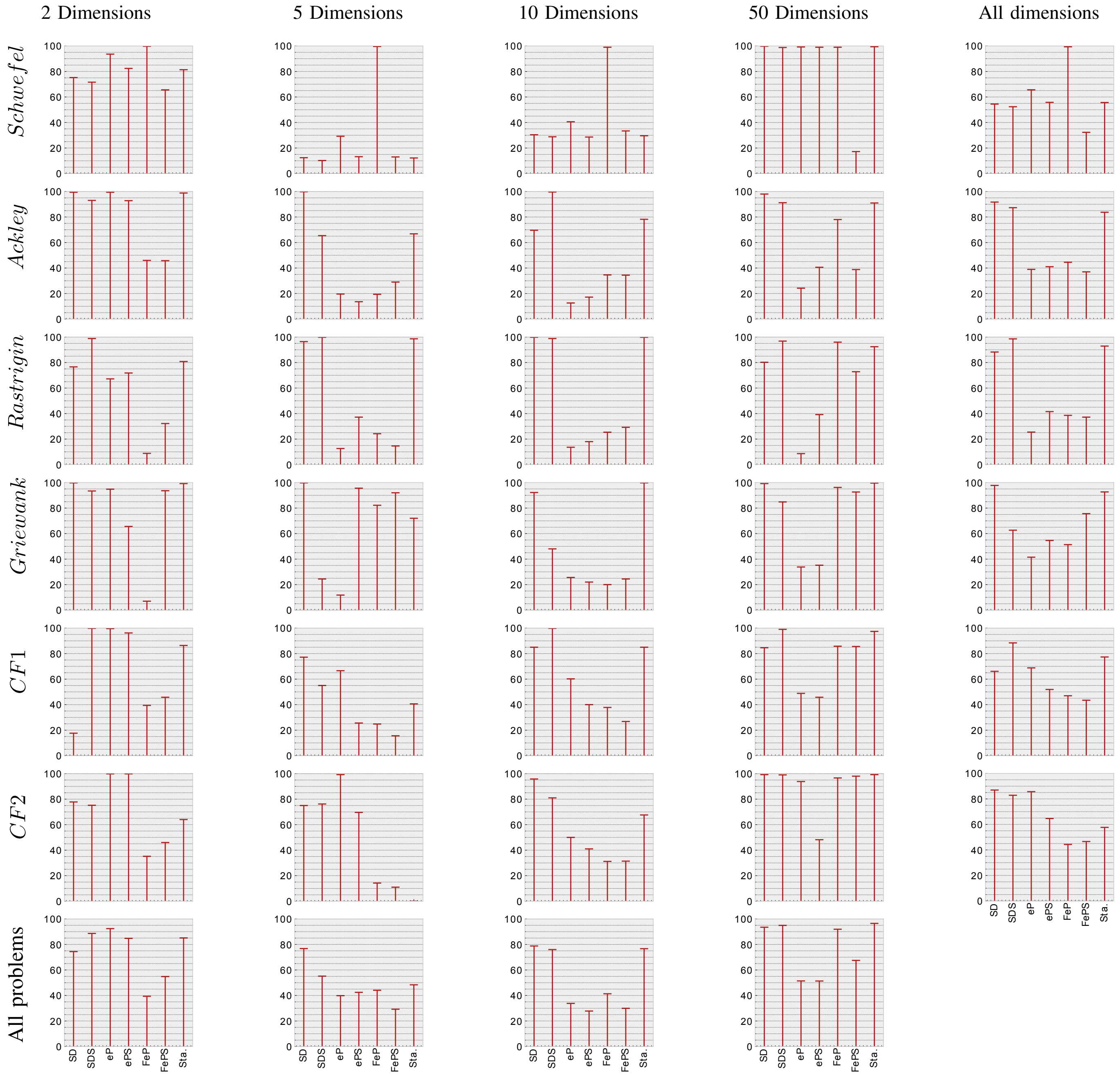

Fig. 7: Proportion of run that each optimiser is not significantly worse than the median best performing.

proaches perform less well, and certainly in terms of consistent good performance across dimensionality only the marginal estimated PDF appears to warrant attention on Schewfel, composite function 1 and composite function 2.

The diversity supplied by using surrogate value versions seems to have a marginal effect in general, though on particular problems there appears to be a consistent advantage/disadvantage to use the surrogate version of an algorithm (across dimension sizes) which indicates there is some interplay between the dimension-independent landscape features and this approach within in algorithm, which is worthy of further study.

V. Conclusions

We present extensions to recent work on multi-resolution GAs using automatic discretisation of real-valued decision variables. We explore the utility of a complete estimated PDF (rather than a marginal estimated PDF) in determining the resolution to use for a solution, and we also investigate the use of multi-resolution for surrogate fitness assignment.

We found on the uni-objective test problems employed (ranging from two to 50 design dimensions) a simple standard

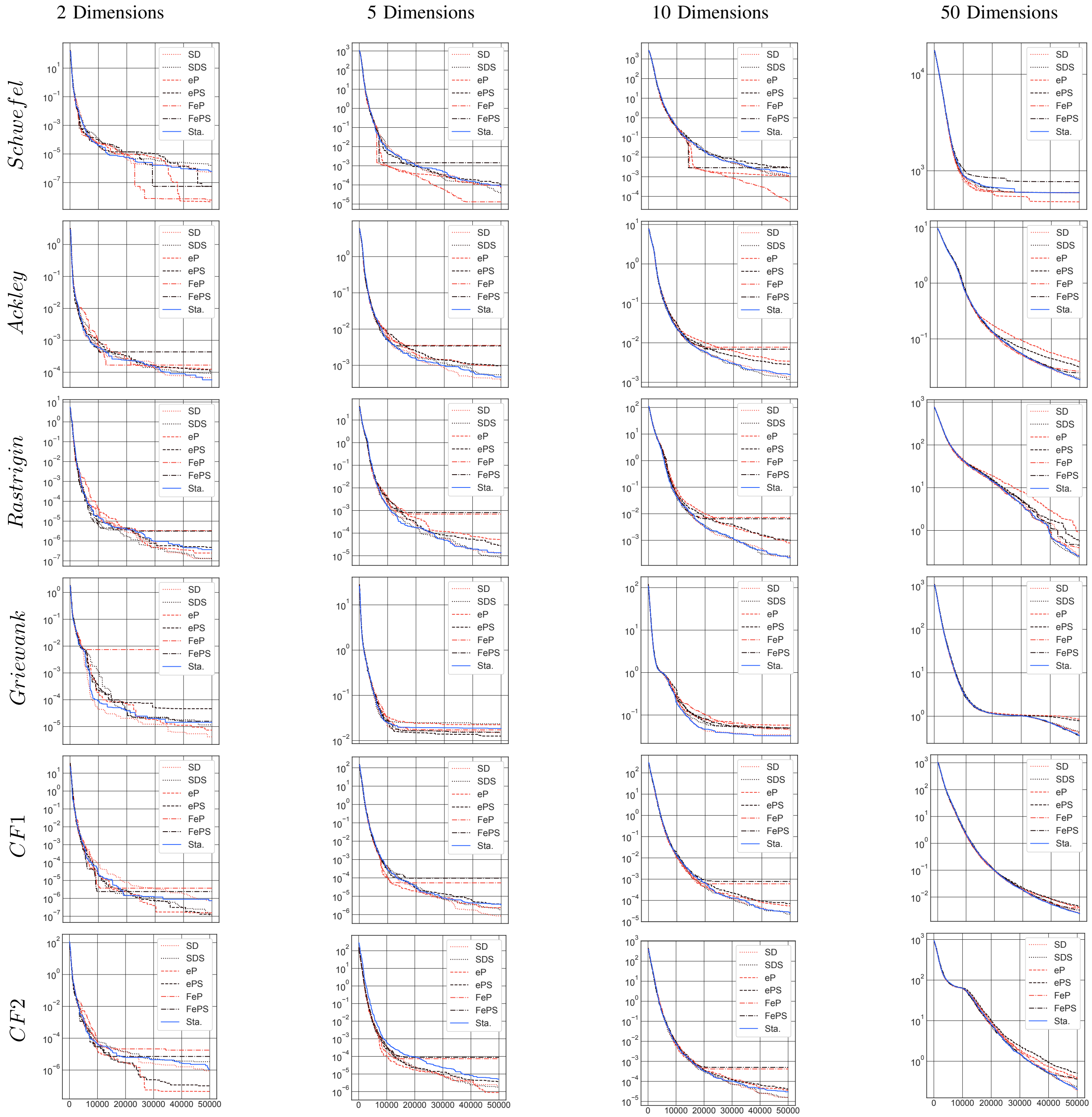

Fig. 8: Number of evaluations against median fitness value (log scale), for each problem and problem size (CF1&2 results are $\times -1$).

deviation-based resolution determination was almost always the most effective, only consistently being troubled by a problem with multiple modes with similar basin sizes. Some form of on-line landscape analysis when using multi-resolution approaches would appear to be vital if performance is to be maximised in such instances.

The use of surrogate values seems to improve performance only marginally in general, but relative performance was seen to vary considerably from problem to problem compared to not using a surrogate. We have also identified that multi-resolution approaches can lead to a high level of duplicate designs, which should lead to significant performance improvements if this property is accounted for in an algorithm implementation. It is worth noting therefore that many studies in this area

TABLE II: The median of the best solutions in the last generation of 30 runs with each indicator (SD, SD-S, ePDF, ePDF-S, Full-ePDF and Full-ePDF-S). Best medians are in bold underlined. Medians from results sets where samples are not statistically worse in performance than the algorithm with the best-performing median are also highlighted in bold. (Significance level of 5% used in the Wilcoxon signed-rank test.) The fist four are minimisation problems, the remainder are maximisation problems.

Problem	Var.	SD	SD-S	ePDF	ePDF-S	Full-ePdf	Full-ePdf-S	Standard
Schwefel	2	**5.36E-07**	**1.59E-06**	**4.48E-09**	**5.42E-08**	**6.04E-09**	**5.42E-08**	**6.14E-07**
	5	9.66E-05	3.86E-05	7.16E-05	1.08E-04	**1.30E-05**	1.44E-03	8.19E-05
	10	1.01E-03	1.15E-03	1.07E-03	2.99E-03	**5.27E-05**	2.89E-03	1.51E-03
	50	**5.93E+02**	**5.93E+02**	**4.77E+02**	**5.94E+02**	**5.93E+02**	7.71E+02	**5.93E+02**
Ackley	2	**6.79E-05**	**8.89E-05**	**1.24E-04**	**1.07E-04**	1.67E-04	4.34E-04	**5.78E-05**
	5	**3.98E-04**	**5.50E-04**	9.50E-04	9.71E-04	3.56E-03	3.43E-03	**4.76E-04**
	10	1.39E-03	**1.17E-03**	3.42E-03	2.76E-03	7.75E-03	6.89E-03	1.58E-03
	50	**1.85E-02**	**1.96E-02**	3.93E-02	3.14E-02	2.65E-02	2.48E-02	**1.86E-02**
Rastrigin	2	**1.34E-07**	**1.30E-07**	**2.48E-07**	4.78E-07	3.32E-06	3.15E-06	3.69E-07
	5	1.28E-05	**8.10E-06**	4.96E-05	2.67E-05	6.93E-04	8.09E-04	1.36E-05
	10	**2.40E-04**	**2.26E-04**	7.92E-04	1.02E-03	7.25E-03	6.46E-03	**2.20E-04**
	50	2.62E-01	**2.35E-01**	9.10E-01	6.04E-01	3.94E-01	4.57E-01	2.50E-01
Griewank	2	**4.14E-06**	**1.15E-05**	**7.40E-06**	4.64E-05	7.40E-03	**1.59E-05**	**1.45E-05**
	5	**1.50E-02**	2.35E-02	2.24E-02	**1.26E-02**	**1.73E-02**	**1.53E-02**	1.85E-02
	10	**3.40E-02**	4.98E-02	5.78E-02	4.91E-02	4.37E-02	4.95E-02	**3.24E-02**
	50	**3.75E-01**	4.29E-01	8.51E-01	7.88E-01	4.13E-01	**3.60E-01**	**3.57E-01**
CF1	2	-7.28E-07	**-1.20E-07**	**-1.54E-07**	**-1.30E-07**	-3.71E-06	-2.39E-06	-7.42E-07
	5	**-8.48E-07**	-1.84E-06	-2.38E-06	-3.81E-06	-5.36E-05	-9.67E-05	-3.41E-06
	10	-2.85E-05	**-2.27E-05**	-5.55E-05	-7.13E-05	-6.05E-04	-7.81E-04	-2.60E-05
	50	**-2.67E-03**	**-2.41E-03**	-4.02E-03	-4.55E-03	-3.70E-03	-3.29E-03	**-2.48E-03**
CF2	2	-8.94E-07	-3.34E-06	**-4.53E-08**	**-1.02E-07**	-1.78E-05	-7.14E-06	-1.14E-06
	5	**-2.17E-06**	**-1.79E-06**	**-8.89E-07**	-3.65E-06	-7.39E-05	-8.84E-05	-5.09E-06
	10	**-1.56E-05**	**-1.48E-05**	-3.48E-05	-4.12E-05	-4.17E-04	-5.02E-04	-3.01E-05
	50	**-1.94E-01**	**-2.34E-01**	-3.90E-01	-5.07E-01	-3.87E-01	**-3.64E-01**	**-2.02E-01**

could be under-reporting the performance of multi-resolution approaches if they did not take this property into account.

Acknowledgment

Abdulaziz Almutairi gratefully thank the Shaqra University in Saudi Arabia for his PhD Scholarship.

References

[1] I. Y. Kim and O. de Weck, "Variable chromosome length genetic algorithm for structural topology design optimization," in *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, 2004, p. 1911.

[2] E. Sinha and B. S. Minsker, "Multiscale island injection genetic algorithms for groundwater remediation," *Advances in water resources*, vol. 30, no. 9, pp. 1933–1942, 2007.

[3] M. Babbar and B. S. Minsker, "Groundwater remediation design using multiscale genetic algorithms," *Journal of water resources planning and management*, vol. 132, no. 5, pp. 341–350, 2006.

[4] T. F. Chan, J. Cong, J. R. Shinnerl, K. Sze, M. Xie, and Y. Zhang, "Multiscale optimization in vlsi physical design automation," in *Multiscale Optimization Methods and Applications*. Springer, 2006, pp. 1–67.

[5] W. Sun and Y. Dong, "Study of multiscale global optimization based on parameter space partition," *Journal of global optimization*, vol. 49, no. 1, pp. 149–172, 2011.

[6] T. H. Nguyen, G. H. Paulino, J. Song, and C. H. Le, "A computational paradigm for multiresolution topology optimization (MTOP)," *Structural and Multidisciplinary Optimization*, vol. 41, no. 4, pp. 525–539, 2010.

[7] ——, "Improving multiresolution topology optimization via multiple discretizations," *International Journal for Numerical Methods in Engineering*, vol. 92, no. 6, pp. 507–530, 2012.

[8] J. Park and A. Sutradhar, "A multi-resolution method for 3D multi-material topology optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 285, pp. 571–586, 2015.

[9] M. Babbar, "Multiscale parallel genetic algorithms for optimal groundwater remediation design," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2002.

[10] I. Y. Kim and O. De Weck, "Variable chromosome length genetic algorithm for progressive refinement in topology optimization," *Structural and Multidisciplinary Optimization*, vol. 29, no. 6, pp. 445–456, 2005.

[11] A. Lopez Jaimes and C. Coello Coello, "MRMOGA: Parallel evolutionary multiobjective optimization using multiple resolutions," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 3. IEEE, 2005, pp. 2294–2301.

[12] H. Li and K. Deb, "Challenges for evolutionary multiobjective optimization algorithms for solving variable-length problems," in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 2217–2224.

[13] J. A. Alshawawreh, "Multi-scale optimization using a genetic algorithm," Ph.D. dissertation, Western Michigan University, 2011.

[14] T. Kondo and T. Tatsukawa, "Efficient search techniques using adaptive discretization of design variables on real-coded evolutionary computations," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2018, pp. 697–704.

[15] H.-P. Schwefel, *Numerical optimization of computer models*. John Wiley & Sons, Inc., 1981.

[16] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," *RMIT University, Evolutionary Computation and Machine Learning Group, Australia, Tech. Rep*, 2013.

[17] D. H. Ackley, "An empirical study of bit vector function optimization," *Genetic algorithms and simulated annealing*, pp. 170–204, 1987.

[18] A. O. Griewank, "Generalized descent for global optimization," *Journal of optimization theory and applications*, vol. 34, no. 1, pp. 11–39, 1981.

[19] J. Schaffer, "A study of control parameters affecting online performance of genetic algorithms for function optimization," *San Meteo, California*, 1989.

[20] H. Mühlenbein, M. Schomisch, and J. Born, "The parallel genetic algorithm as function optimizer," *Parallel computing*, vol. 17, no. 6-7, pp. 619–632, 1991.

[21] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.