The Rolling Tide Evolutionary Algorithm: A Multi-Objective Optimiser for Noisy Optimisation Problems

Jonathan E. Fieldsend *Member, IEEE* and Richard M. Everson

***Abstract*—As methods for evolutionary multi-objective optimisation (EMO) mature and are applied to a greater number of real world problems, there has been gathering interest in the effect of uncertainty and noise on multi-objective optimisation. Specifically, how algorithms are affected by it, how to mitigate its effects, and whether some optimisers are better suited to dealing with it than others. Here we address the problem of uncertain evaluation, where the uncertainty can be modelled as additive noise in objective space. We develop a novel algorithm, the rolling tide evolutionary algorithm (RTEA), which progressively improves the accuracy of its estimated Pareto set, whilst simultaneously driving the front towards the true Pareto front. It can cope with noise whose characteristics change as a function of location (both design and objective), or which alter during the course of an optimisation. Four state-of-the-art noise-tolerant EMO algorithms, as well as four widely used standard EMO algorithms, are compared to RTEA on 70 instances of ten continuous space test problems from the CEC'09 multi-objective optimisation test suite. Different instances of these problems are generated by modifying them to exhibit different types and intensities of noise. RTEA is seen to provide competitive performance across both the range of test problems used, and noise types.**

I. Introduction

Evolutionary computation (EC) techniques are now extensively used when attempting to discover the optimal or near-optimal parametrisation for problems with complex function transformation from parameters (design variables) to objective(s). Almost all optimisation procedures search the parameter space by evaluating the objectives for a given parametrisation before proposing a new, hopefully better, parametrisation. It is generally assumed that repeated evaluation of the objectives for a single parametrisation yields the same objective values. However, in many problems there is additional uncertainty in the veracity of the results obtained from the system model. Clear examples arise in "embodied" optimisation [1], [2], [3], where measurement error or stochastic elements in a physical system leads to different results for repeated evaluations at the same parameter values [4], [5] or when the objectives are derived from Monte Carlo simulations or data-driven systems [6], [7].

Work in the scalar optimisation EC community to tackle these types of problem has indicated that elitist techniques often prove fragile [8], [9], [10], as there is no longer the guarantee that fitness of the elite solution improves as an optimiser progresses. As most modern multi-objective evolutionary algorithms (MOEAs) rely heavily on elitism, this should be of concern when optimising uncertain multi-objective problems.

As early as 1993, the effect of noise on multi-objective optimisation was being considered, with Horn & Nafpliotis (at the end of [11]), discussing the ways in which uncertainty in an evaluation might be tackled. However, it was not until the early 2000s that a significant number of multi-objective optimisers began to be developed specifically to tackle uncertain optimisation [12], [13], [14], [5], [6], [15], [7], [16], [17], [18], [19], [20], [21], [22], [23]. Typically uncertainty is modelled as noise added to the function evaluations, although recent work has begun looking at the situation where the objective functions themselves are uncertain [24].

In this paper we model the uncertainty in the objectives as observational noise. For the reader interested in the effect and mitigation of parametrisation uncertainty (often referred to as robust optimisation, or reliablity-based optimisation [25]), we recommend the work of Deb & Gupta [26]. This is also distinct from *dynamic* optimisation [27], in which the objectives change with time, but there is no uncertainty about each objective evaluation. The algorithm we examine here does, however, allow for dynamic noise characteristics; see also [6]. From here on we will use the term 'optimisation with uncertainty' as shorthand for objective uncertainty, which is the focus of this work.

We describe the multi-objective optimisation problem and some different types of observational noise which may be experienced in Section II, and in Section III we examine the effect of noise on a number of popular multi-objective evolutionary algorithms (MOEAs) – namely the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [28], the Pareto Archived Evolution Strategy (PAES) [29], the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [30] and the Indicator-Based Evolutionary Algorithm (IBEA$_{\epsilon+}$) [31]. In Section IV we briefly describe some MOEAs that have been developed to cope with different types of observational noise prior to introducing our new algorithm, the rolling tide evolutionary algorithm (RTEA) in Section V.

In Section VI we evaluate our new optimiser along with four

The authors are with the College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, EX4 4QF, UK (e-mail: {J.E.Fieldsend,R.M.Everson}@exeter.ac.uk).

other recent noise-tolerant MOEAs [6], [16], [21], [22], on standard test problems modified by the addition of noise. In particular we show that our proposed optimiser performs well in the presence of noise with a wide range of characteristics, including when the noise varies spatially and temporally during the course of the optimisation. We conclude with a discussion in Section VII.

II. Multi-objective optimisation and noise

Before discussing the effect of uncertainty in the evaluation of objectives, we briefly review the ideas of dominance and Pareto optimality which are central to multi-objective optimisation.

A general multi-objective optimisation problem seeks to simultaneously extremise D objectives: $f_d(\mathbf{x})$, $d = 1, \ldots, D$ where each objective depends upon a vector $\mathbf{x} = (x_1, \ldots, x_k, \ldots, x_K)$ of K parameters or design variables.

The parameters may also be subject to equality and inequality constraints which, for simplicity, we assume can be evaluated precisely. When the objectives are to be minimised, the multi-objective optimisation problem may thus be expressed as:

$$\text{minimise} \qquad \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_D(\mathbf{x})) \tag{1}$$

with $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$, the feasible domain as defined by the constraints. When faced with only a single objective an optimal solution is one which minimises the objective given the model constraints. However, when there is more than one objective to be minimised, solutions may exist for which performance on one objective cannot be improved without reducing performance on at least one other. Such solutions are said to be *Pareto optimal.*

The notion of *dominance* may be used to make Pareto optimality clearer. Assuming, without loss of generality, that the goal is to minimise the objectives and there is no noise, a design vector $\mathbf{x}$ is said to *strictly dominate* another $\mathbf{x}'$ iff

$$f_d(\mathbf{x}) \leq f_d(\mathbf{x}') \quad \forall d = 1, \ldots, D \quad \text{and} \quad \mathbf{f}(\mathbf{x}) \neq \mathbf{f}(\mathbf{x}'). \tag{2}$$

This is often denoted as $\mathbf{x} \prec \mathbf{x}'$ (as opposed to $\mathbf{f}(\mathbf{x}) \prec \mathbf{f}(\mathbf{x}')$). A set of design vectors A is said to be a *mutually non-dominating set* if no member of the set is dominated by any other member. A solution to the minimisation problem (1) is thus *Pareto optimal* if it is not dominated by any other feasible solution, and the non-dominated set of all Pareto optimal solutions is the Pareto set $\mathcal{P}$.

The Pareto dominance notation can also be extended to sets (see e.g. [32]). This leads to a larger number of distinctions; however the set comparison we need here is: $A \prec B$, which signifies that every $\mathbf{x} \in B$ is dominated by at least one member of A.

In this paper we are concerned with problems in which the objectives themselves are unobservable, but instead we have access to y_d the objectives contaminated by *observational* noise ϵ:

$$y_d = f_d(\mathbf{x}) + \epsilon_d. \tag{3}$$

We assume throughout that the observational noise encountered at each evaluation is independent of other evaluations.

The imposition of noise in an optimisation environment means that the dominance relationship calculated between solutions may be incorrect. Prior to discussing its effects it is useful to consider the different *types* of observational noise that might be experienced in a particular industrial problem.

A. Unknown noise

The most severe case is when the noise characteristics are completely unknown. It may be asymmetric, it may be multimodal, and there may be dependencies between noise in different objectives. Therefore, it could well be the case that the mean of a number of re-evaluations of $\mathbf{y}$ is a very poor approximation to $\mathbf{f}(\mathbf{x})$. In this case the degree of dominance between two solutions can only reasonably be estimated by repeated evaluation at the fixed parameter values $\mathbf{x}$ and $\mathbf{x}'$. Suppose that $Y(\mathbf{x}) = \{\mathbf{y}_i\}_{i=1}^n$, are the noisy objectives evaluated n times at $\mathbf{x}$, and $Y(\mathbf{x}') = \{\mathbf{y}'_i\}_{i=1}^{n'}$ are n' evaluations at $\mathbf{x}'$, then the probability that $\mathbf{x}$ dominates $\mathbf{x}'$ is estimated by the fraction of times that $\mathbf{y}_i$ dominates $\mathbf{y}'_j$:

$$p(\mathbf{x} \prec \mathbf{x}') = \frac{1}{nn'} \sum_{i=1}^{n} \sum_{j=1}^{n'} I(\mathbf{y}_i \prec \mathbf{y}'_j) \tag{4}$$

where $I(\cdot)$ is the indicator function.

Estimating dominance by this sampling method clearly requires several evaluations of the objectives at both $\mathbf{x}$ and $\mathbf{x}'$, which may be prohibitively expensive.

The cost of this approach can be substantially reduced if it is known or it can be assumed that the noise corrupting each objective is independent. In this case the evaluations for each objective dimension can be permuted to form additional samples, providing with n^D objective vectors for $\mathbf{x}$ and n'^D objective vectors for $\mathbf{x}'$ respectively.[1]

B. Independent noise for each objective

If the noise contaminating each objective can be assumed to be independent of the other objectives, then the probability of dominance can be decomposed into a product of probabilities for each dimension:

$$p(\mathbf{x} \prec \mathbf{x}') = \prod_{d=1}^{D} p(f_d(\mathbf{x}) < f_d(\mathbf{x}')). \tag{5}$$

Each of the constituent probabilities $p(f_d(\mathbf{x}) < f_d(\mathbf{x}'))$ is:

$$\int_{-\infty}^{\infty} p(f_d(\mathbf{x}) \,|\, Y_d(\mathbf{x})) \int_{f_d(\mathbf{x})}^{\infty} p(f_d(\mathbf{x}') \,|\, Y'_d(\mathbf{x}))\, df_d(\mathbf{x}')\, df_d(\mathbf{x}) \tag{6}$$

where $Y_d(\mathbf{x})$ and $Y'_d(\mathbf{x})$ represent the set of evaluations from repeated samples of (3) for $\mathbf{x}$ and $\mathbf{x}'$ respectively. These integrals can be computed if additional information about the noise

[1] This was incorrectly reported as 2^n and $2^{n'}$ in [6].

distributions is known. Hughes [14] has addressed the case of Gaussian noise with known noise variance, and his work has been built upon by a number of researchers, e.g. [6], [21], [22], [23]. An important simplification occurs when the noise is known to be bounded: Teich [13] has modelled the noise as uniform and has proposed modified archive acceptance schemes. Note however, that the boundedness of the noise means that if the solutions are sufficiently well separated in objective space then dominance or lack of dominance may be decided with no uncertainty.

C. Estimators of the noise-free objectives

In order to make as few assumptions about the noise as possible we assume that the noise-free objectives can be estimated from a collection of samples at $\mathbf{x}$. Let $Y(\mathbf{x}) = \{\mathbf{y}_i(\mathbf{x})\}_{i=1}^n$, be n evaluations of the objectives corresponding to $\mathbf{x}$. Then we denote the estimate of the noise-free objectives by $\mathrm{est}(Y(\mathbf{x}))$. Commonly $\mathrm{est}(\cdot)$ will be a maximum likelihood estimator and will thus depend on the noise distribution; often the estimator will be some sort of average, for example, the maximum likelihood estimator is the mean for Gaussian noise and the median for Laplacian noise. In general the accuracy of the estimate increases in proportion to the square root of the number of samples. We assume that the estimator is unbiased in the sense that with sufficiently many samples it converges to the noise-free objectives:

$$\lim_{n\to\infty} \mathrm{est}(\{\mathbf{y}_i(\mathbf{x})\}_{i=1}^n) = \mathbf{f}(\mathbf{x}). \tag{7}$$

Estimators for symmetrically distributed noise (e.g., the mean for Gaussian noise and the median for Laplacian noise) are usually unbiased, but with asymmetric noise distributions it is possible that the estimator will be biased, converging to $\mathbf{f}(\mathbf{x})$ plus a constant. Dominance-based optimisation can still be performed with biased estimators, because the dominance relation is not affected by the addition of a constant, however the resulting objective values can only be estimated up to the additive constant.

D. Time-varying noise and spatially-varying noise

Most work on noisy multi-objective optimisation has focussed on static noise, however there are many instances where the noise characteristics may *vary*, depending on either the location (in objective or design space), the time, or indeed both [6], [21], [23]. Example real world sources of temporal noise are sensors which degrade over time, or vary due to replacement during an optimisation run. Spatial noise may be derived from sensors whose noise varies with the value being measured (for example heat sensors).

III. The effect of noise on standard MOEAs

Elitist MOEAs generally maintain a mutually non-dominating set or archive A of solutions which form their estimate of the Pareto set, because the archive is also usually non-dominated with respect to all solutions evaluated by an optimiser.[2] As the optimisation proceeds new solutions are generated, either by copying and perturbing solutions in A (e.g., [29]) or by mutating and recombining solutions in a search population, often in combination with A (e.g., [33], [28], [31]). If a new solution $\mathbf{x}'$ is not dominated by a member of A then $\mathbf{x}'$ is added to A, and any solutions in A that are dominated by $\mathbf{x}'$ are deleted from A. In this way A is always a non-dominated set, whose image in objective space cannot move away from the Pareto front, $\mathcal{F}$, the image of $\mathcal{P}$ in objective space. Note however, that when the size of $|A|$ is limited, new additions to A can move the estimated front away from the true Pareto front [34], [35].

Taking as a starting point an elitist MOEA, the effect of noise is threefold. Firstly, solutions which should be added to A may be rejected due to the corrupting noise which may misrepresent them as being dominated by an element of A, and conversely, solutions that should not be inserted into the archive may be erroneously entered, as the noise makes them seem better than they actually are. This therefore has the effect of reducing the algorithm's efficiency. Secondly, the final archive may overestimate the true Pareto front due to outlier noise realisations making the performance of a parameterisation not only appear better than it is, but also better than is feasible by *any* point in $\mathcal{X}$. Finally the resultant archive may also contain a large number of solutions which, if evaluated without noise, would actually be dominated by other members of A or, indeed, by other solutions that were encountered earlier in the search and rejected from A.

A. Quality measures for noisy optimisation

The *quality* of the output of a MOEA is generally difficult to quantify, with different criteria often used to evaluate the non-dominated estimate of the Pareto set produced [36]. In a noisy optimisation, we are concerned with how well the estimated Pareto set matches the actual Pareto set, and also how well the objective values associated with solutions' $\mathrm{est}(Y(\mathbf{x}))$ match their noise-free objective values. Here we use the inverse generational distance and hypervolume measure to quantify how well the estimated non-dominated solutions match the true Pareto set. We also measure the average distance of the objectives associated with designs returned by an optimiser to the *true* noiseless objective values for the designs. We call this measure of quality the noise misinformation (NM) measure. Figure 1 provides an illustration of how these quality measures are calculated.

The noise misinformation measure (see Figure 1b) is the root mean squared Euclidean distance between the best estimate, $\mathrm{est}(Y(\mathbf{x}))$, of the objectives for a particular design and the

[2] Some MOEAs such as SPEA2 augment the archive with dominated solutions if insufficient non-dominated solutions are available; however, the non-dominated subset from such archives is easy to extract.

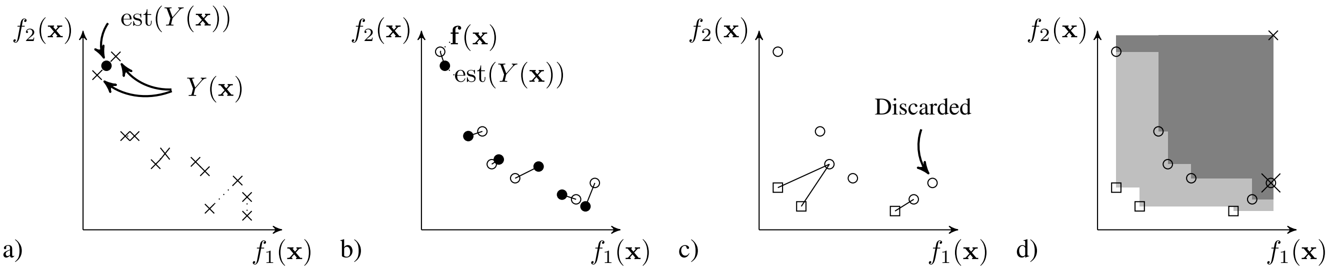

Figure 1. Illustrations of the calculation of NM, IGD, and hypervolume in noisy optimisation problems. *a)* First the estimated objective locations for each $\mathbf{x}$ in A needs to be determined. If there is only a single evaluation per $\mathbf{x}$ then this value is directly used. If multiple samples have been evaluated at each parametrisation (illustrated here with two samples for each archive solution), then these are used to generate the $\text{est}(Y(\mathbf{x}))$. *b)* Illustrates the calculation of the NM measure. Filled circles indicate the estimated Pareto front ($\text{est}(Y(\mathbf{x}))$ locations) returned by an algorithm. Unfilled circles indicate the actual (noise-free) objective values for each $\mathbf{x}$. NM is calculated as the root mean squared distance between the $\text{est}(Y(\mathbf{x}))$ and $\mathbf{f}(\mathbf{x})$. *c)* Illustrates the IGD_2 calculation. Unfilled squares indicate samples from the true/noise-free Pareto front and the unfilled circles the noise free evaluations of A. Solutions which are dominated by other elements of A in the noise-free evaluation are discarded. IGD_2 is the root mean squared distance between each member of the true Pareto front which is stored in $\mathcal{F}$ and the closest noise-free objective $\mathbf{f}(\mathbf{x})$ for $\mathbf{x} \in A$. *d)* Illustrates the hypervolume calculation. The region dominated by both the Pareto front and the elements of A assessed through the non-dominated noise-free evaluations $\mathbf{f}(\mathbf{x})$ for $\mathbf{x} \in A$ is filled with a dark grey, and the region dominated by only the true Pareto front $\mathcal{F}$ is shown in light grey (the reference point used is indicated with the small cross in the top right of the panel). As in *c*, dominated noise-free evaluations are discarded. The hypervolume is the ratio between the area/volume of the dark grey region and that of the combined dark grey and light grey regions.

noise-free value:

$$\text{NM} = \left(\frac{1}{|A|} \sum_{\mathbf{x} \in A} \| \text{est}(Y(\mathbf{x})) - \mathbf{f}(\mathbf{x}) \|_2^2 \right)^{\frac{1}{2}}. \tag{8}$$

The closer on average the estimated Pareto optimal solutions are to their actual noise-free location, the smaller the NM for a particular algorithm.

If the solutions in A are evaluated using the noise-free objectives, the set of objectives associated with the solutions may now contain some dominated solutions. After these dominated solutions have been rejected (see Figure 1c) the proximity and coverage of the estimated Pareto front is also assessed using two widely used quality measures, the inverse generational distance and the hypervolume. Let $\hat{F} = \text{nondom}(\mathbf{f}(\mathbf{x}) \,|\, \mathbf{x} \in A)$, where $\text{nondom}(E)$ is the function that returns the maximal set of non-dominated members of E:

$$\text{nondom}(E) = \{\mathbf{u} \in E \,|\, \nexists(\mathbf{v} \in E \wedge \mathbf{v} \prec \mathbf{u})\}. \tag{9}$$

We use the power mean version of the inverse generational distance (IGD_p) [37], which is not biased by the size of the estimated Pareto front unlike standard IGD. Given a set F of samples from the true front, this is calculated as:

$$IGD_p(\hat{F}, F) = \left(\frac{1}{|F|} \sum_{\mathbf{u} \in F} \text{dist}(\mathbf{u}, \hat{F})^p \right)^{\frac{1}{p}}. \tag{10}$$

Here the function $\text{dist}(\mathbf{u}, \hat{F})$ is the Euclidean distance between $\mathbf{u}$ and the closest member of $\hat{F}$, so that $IGD_p(\hat{F}, F)$ measures how well on average each sample from the true Pareto front has been approximated by the estimate. As is standard, we use $p = 2$ in our calculations.

The hypervolume measures the proportion of a given region (defined by $\mathcal{F}$ — or a representative subset F — along with a reference point) that is dominated by both the Pareto front *and* the (noise-free) estimate returned by an MOEA. Unlike the other two measures, the hypervolume is to be maximised. As with the selection of individuals for assessment in IGD_p, the solutions used from A in the calculation of the hypervolume are $\hat{F}$, the non-dominated subset when assessed using the noise-free test function (see Figure 1d).

B. *Empirical effects*

Here we evaluate the effect of different intensities of noise on the performance of four standard (unmodified) MOEAs — PAES, SPEA2, NSGA-II and IBEA$_{\epsilon+}$. For these optimisers, therefore, the estimated objective values for a solution are simply a single evaluation of that solution, i.e. $\text{est}(Y(\mathbf{x})) \equiv \mathbf{y}(\mathbf{x})$. We use the problems UP1-10 from the CEC'09 multi-objective optimisation competition suite [38], and modify them by adding zero-mean isotopic Gaussian noise to each objective, i.e., in (3): $\epsilon_d \sim \mathcal{N}(0, \sigma^2)$.

A number of the CEC'09 problems are scalable, and for illustration we use the number of parameters and objectives used in the competition itself [38]. Each MOEA was run 30 times for 300000 function evaluations (as in the competition) for four different σ levels, $\sigma = \{0.00, 0.01, 0.10, 1.00\}$. When $\sigma = 0.00$ the objective vector returned for a solution during a run is the true (noiseless) evaluation. The non-dominated members of an unconstrained passive elite archive are recorded for each algorithm every 500 function evaluations; these track the best locations (as perceived in the noisy objective space) that each MOEA has ever visited.

Figures 2 and 3 show the median performance of the algorithms using the IGD_2 and hypervolume measures. It is clear that noise inhibits convergence, and that the effect increases with increasing variance. Even for a very small level of noise, $\sigma = 0.01$, the performance of algorithms that previously found solutions in the vicinity of $\mathcal{F}$ all degrade. Interestingly one of the problems UP7, behaves slightly differently to the rest: when $\sigma = 0.10$ performance of SPEA2 actually *improves*. A similar effect is seen on UP9 for $\sigma = 0.01$. This effect

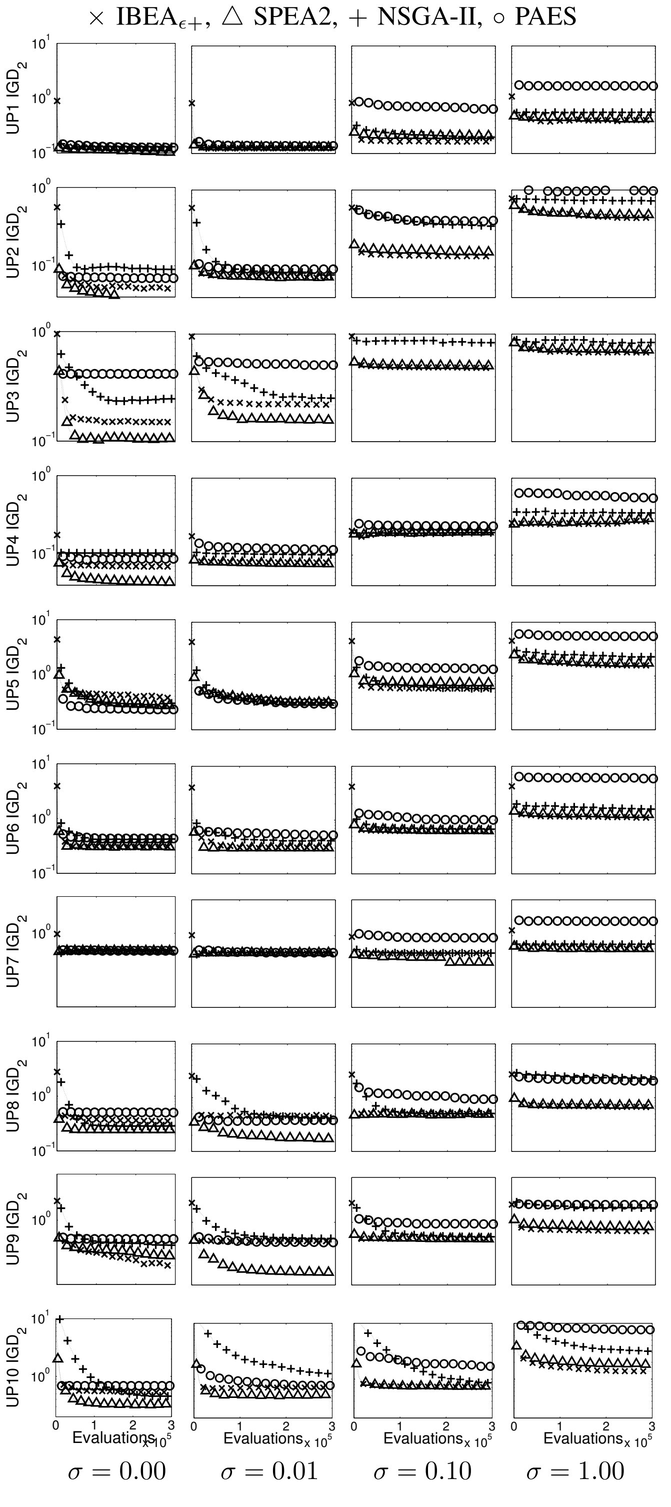

Figure 2. Effect of different noise intensities on popular MOEAs on the IGD_2 measure. Top to bottom: CEC'09 problems 1-10, Left to right: σ values of $\{0.00, 0.01, 0.10, 1.00\}$ used. Median results over 30 runs plotted on log scales.

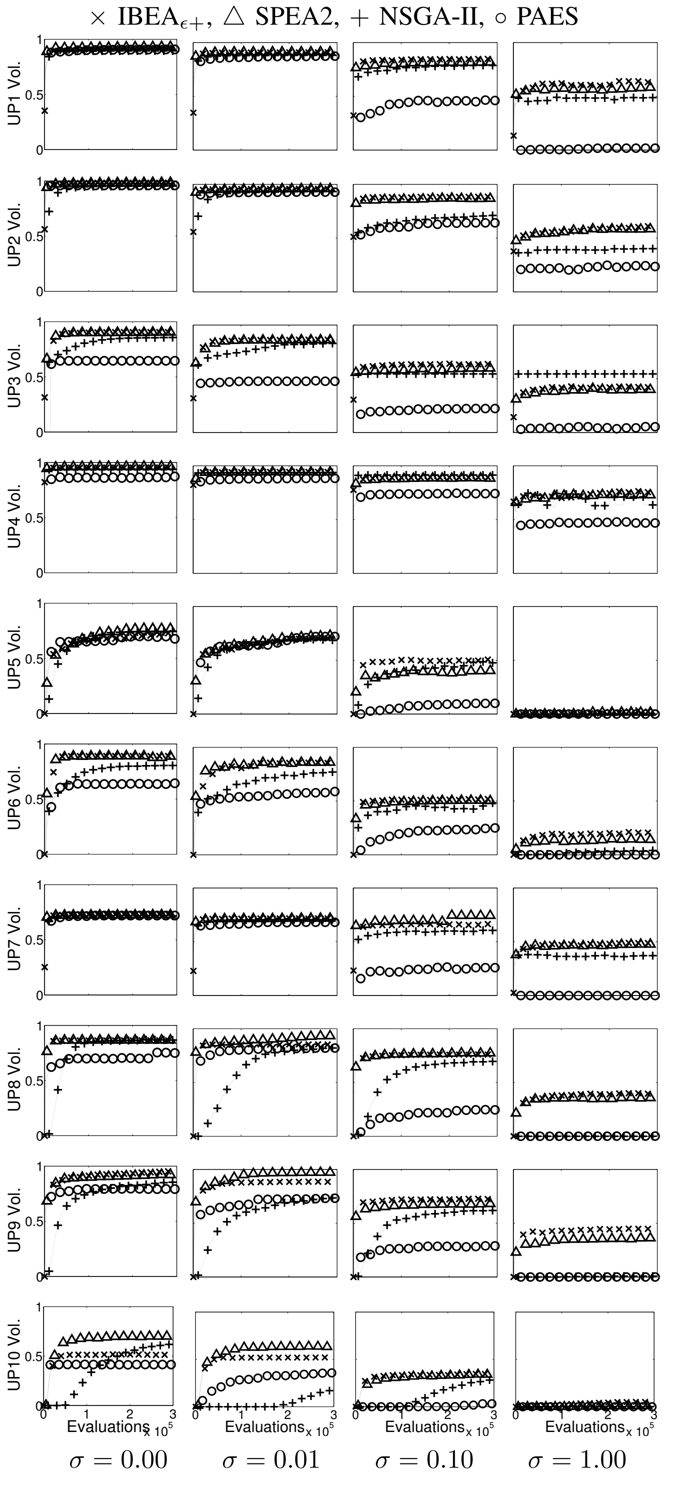

Figure 3. Effect of different noise intensities on popular MOEAs on the hypervolume measure. Top to bottom: CEC'09 problems 1-10, Left to right: σ values of $\{0.00, 0.01, 0.10, 1.00\}$ used. Median results over 30 runs plotted. Reference points used in the hypervolume calculation are (2, 2) and (2, 2, 2), for two-objective and three-objective test problems respectively.

is not peculiar to SPEA2 (as we shall see later, some noise-tolerant optimisers also exhibit this behaviour on UP7), and our conjecture is that the increased diversity in the elite population that results from low levels of noise actually makes UP7 and UP9 easier to search (albeit for *different* small values of noise).

The noise misinformation in the elite solutions for each algorithm is presented in Figure 4. All the results on this measure show a great degree of similarity, so we simply provide plots for UP1 and UP8 (prototypical two and three objective problems). On average, NSGA-II and PAES tend

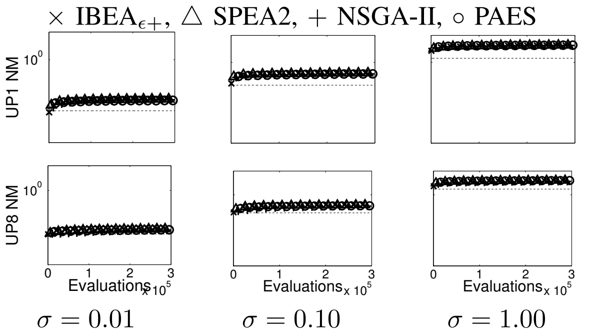

Figure 4. Effect of different noise intensities on popular MOEAs on the NM measure. Top to bottom: CEC'09 problems UP1,8, Left to right: σ values of $\{0.01, 0.10, 1.00\}$ used. Median results over 30 runs plotted on log scales. Horizontal dashed lines denote the *expected* NM values if the deviation of passive archived solutions from the noiseless values was unbiased. Note: when $\sigma = 0.00$ the MN measure is 0 and so is not plotted.)

to be a little lower, however all algorithms perform in a narrow performance band on this measure. As the optimisation proceeds the NM values can all be seen to rise above the value that would be expected if the solutions were not members of a non-dominated archive. That is, with D objectives it might be anticipated that $NM = \sqrt{D}\sigma$. In Figure 4 the initial NM is approximately $\sqrt{D}\sigma$, but then rapidly increases above this level. This phenomenon is well-known and is noted in e.g. [16], [17]. In noisy optimisation there are essentially *two* drivers pushing the estimated Pareto front forward. The first is (as in noiseless multi-objective optimisation) simply the selection of better solutions. The second driver is the selection of solutions which are affected by *outlier* noise [5], [16], [17], that is, noise of large magnitude that makes a solution appear much better than it actually is on one or more objectives. This can rapidly become the main driver for the movement of the estimated front, which quickly becomes filled with those solutions that due to the noise appear to dominate others. This results in the early divergence of the NM measure from the expected value, and highlights the bias introduced by noise which skews the assigned objective values in archives.

In this section we have examined the effect of noise on some popular MOEAs, and seen that it causes premature convergence and can exacerbate existing search difficulties for most problem types even with moderate noise. We now briefly look at some of the approaches that have been developed to tackle noisy multi-objective optimisation problems.

IV. Noise-tolerant MOEAs

A number of different avenues have been explored by researchers tackling noisy multi-objective optimisation problems. Broadly these can be categorised as those which modify how solutions are compared and stored (as an attempt to prevent 'bad' solutions polluting the archive), and those which accept that misleading solutions will exist in the elite set and therefore attempt to reduce the effect of these sub-optimal solutions in the elite population.

The dominance adjustment/redefining approaches almost all involve resampling at a particular parameter value. Resampling is used to refine the dominance estimation between pairs of solutions, such that the objective quality of a solution is calculated not on a single assessment at a location, but the average over a number of samples. In [5] a fixed number of samples is used and in [21] the number (up to a user-defined maximum) is dynamically varied, based on the estimated variance calculated for a solution; this allows dominance between solutions to be assessed at a particular confidence level prior to assignment to the estimated Pareto subset. The confidence-based dynamic resampling used by [21] is demonstrated to perform favourably compared to basic sampling with a fixed number of samples, fitness inheritance, modified ranking and dominance-dependant lifetime approaches (which are also discussed briefly below) on noisy versions of the ZDT test functions [39]. A variant incorporating a neural network surrogate is also demonstrated on a real world noisy industrial problem. In [6], rather than sample solutions repeatedly until a certain degree of confidence is obtained for a mutually non-dominating set, the probability of dominance is estimated from a fixed number of samples at each design location and used to define an envelope in objective space within which all solutions have a certain probability of *actually* being non-dominated (if they were to be evaluated without noise), even though the $\mathrm{est}(Y(\mathbf{x}))$ may suggest that some members are dominated. This results in an algorithm which maintains a 'thick' elite archive including some suboptimal solutions, but which has a low probability of excluding a solution which would actually be non-dominated without noise. These approaches all build on the work of [13], [14] for the calculation of dominance probabilities. In [7] each solution is modelled as having a probability distribution over objective space, which is estimated by drawing samples at locations and used to estimate the expected indicator function to drive the search process. In [23] the probability of dominance is calculated based on [14] (which assumes the noise is independent on each dimension and Gaussian in nature) and integrated into an estimation of distribution/particle swarm optimisation hybrid algorithm. No resampling is done however, meaning the variance must be known or estimated prior to the optimisation.

In [22] an adaptive number of samples is taken, however unlike [21] the number is not limited. Instead, at each generation all members of a fixed sized archive have additional resamples (alongside the resamples of new locations in the search population). This means that the archive members *accumulate* resamples as the search progresses, and, if the archived solution is found to perform worse than originally estimated after additional sampling, it may be removed from the archive. Selection for archive membership is based on (confidence-adjusted) Pareto rank and crowding, and selection for the parent population is based on the ranking and a weighted fitness derived from the number of probabilistically dominated points, reweighted by the number of resamples a solution has (thereby favouring 'younger' solutions). Promising results are presented on five of the ZDT problems, when compared to NSGA-II (and noise-tolerant variants thereof).

[16] introduces the MOEA with Robust Features (MOEA-RF).

This does not use resampling, but instead uses a range of other techniques to mitigate the effect of noise, principally experimental learning directed perturbation (which attempts to use learning early on in an optimisation run, when noise tends to have a less detrimental effect on an elite population, in order to learn and exploit 'good' directions for perturbations), gene adaptation selection strategy (GASS), which attempts to learn a model of the appropriate area in design space to search, and the use of *possibilistic archiving* which uses fuzzy membership functions to limit the size of the archive (which has an effect akin to ϵ-dominance [40]). Good results for MOEA-RF are reported in comparison with a range of standard MOEAs, as well as the noise-tolerant SPEA [5] and the multi-objective probabilistic selection evolutionary algorithm (MOPSEA) of [12] on the ZDT test problems corrupted with static noise.

In [15] a number of variants of NSGA-II are developed. These include a version which incorporates the probabilistic ranking method of [14], a version which uses resampling to refine the objective location estimate of each solution, and versions where the objective values associated with a new solution are first estimated based upon the mean fitness and noise experienced by its two parents (a form of fitness inheritance), before a decision of whether to resample is taken (this is implemented as an extension of the other approaches). If the observed objectives of the child on evaluation fall within a confidence interval defined by the estimated fitness and noise, then the assigned objective values are accepted, otherwise additional resamples are taken and its fitness is assigned to the mean of these. The performance of these algorithms and standard NSGA-II are compared on noisy ZDT test functions. The resampling approach performing best, with the fitness inheritance variant improving its performance further.

In [5] a number of variants of SPEA are presented and compared on test functions with different noise types. In the best performing version non-dominated solutions in the population are given a maximum lifespan (in terms of algorithm iterations). The lifespan is inversely proportional to the number of solutions in the general population that a non-dominated solution dominates. That is, if it dominates many, it has a short lifespan, but if it dominates just a few, its lifespan is longer. At the end of its life a solution is removed from the elite set, reevaluated, and its objective values *replaced* with the new values. The procedure continues as if the reevaluated solution was a newly proposed solution so that a good solution is likely to be included once more into the non-dominated set, but one that experienced outlier noise is likely to be discarded. Results are also presented on the optimisation of a burner in a gas turbine combustion test rig.

The algorithm we now introduce builds on a number of themes in these earlier works. It seeks to minimise the number of algorithm parameters that need tuning and maximise the range of noise types it can be applied to, as well as delivering estimated Pareto optimal solutions which are well converged to $\mathcal{F}$ and whose $\mathrm{est}(Y(\mathbf{x}))$ are highly accurate reflections of $\mathbf{f}(\mathbf{x})$.

V. The Rolling Tide Evolutionary Algorithm

In this study we eschew the use of variance learning techniques, and instead we focus on the observation that the best estimate for the noise free objectives associated with a design improves with the number of samples taken. Our resultant algorithm is effective not only for static noise, but also time varying and spatially varying noise. As such it can tackle a broader range of problems than many noise-tolerant MOEAs developed so far, with the only *a priori* assumption being that an unbiased estimator of the noise-free objectives is available; often this will just be the mean of the noisy evaluations for a particular design. It also does not require additional parameters for e.g. confidence levels, population size, truncation parameters, etc. that are required by other methods, and instead requires the typical parameters of an evolutionary optimiser, such as crossover rate, crossover type, mutation rate and mutation width, plus a variable determining the number of resamples per iteration, and the refinement length at the end of a run.

The algorithm we introduce here repeatedly hones the estimated fitness of the stored elite solutions, meaning that the final elite archive solutions have increased accuracy, but not at the cost of reevaluating *all* designs proposed by the optimiser in its search. This balances the explorative nature of the search with the accuracy of the solutions driving the search forward. The algorithm employs a search population X, together with an elite archive A. Decisions on whether to add a solution $\mathbf{x}$ to A are based on the estimated noise-free objective values $\mathrm{est}(Y(\mathbf{x}))$, estimated from a number (possibly 1) of samples $Y(\mathbf{x}) = \{\mathbf{y}_i(\mathbf{x})\}_{i=1}^{n}$. In order to gain accurate estimates of the noise free objectives, but to avoid the overhead of reevaluating poor solutions, the algorithm repeatedly reevaluates solutions in A, the leading edge of the estimated Pareto front.

RTEA is outlined in Figure 5. On lines 1-2, r design vectors are sampled uniformly in $\mathcal{X}$ and each of these is evaluated a single time. We denote by $\mathbf{Y}$ the set of sets comprising the objective evaluations; each element of $\mathbf{Y}$ is the set of objective evaluations corresponding to a single solution, $\mathbf{x}$. After initialisation each element of $\mathbf{Y}$ has a single element corresponding to the single evaluation of a member of X. The non-dominated elite archive is initialised from the initial evaluations (line 3), and extracted from X, meaning that A is the estimated Pareto set, and X contains the solutions evaluated by the algorithm that are not in A. Also, for each member of X, a single solution which *dominates* it is recorded; this dominating element may be in X or in A.

After initialisation, the algorithm continues in an optimisation loop. During each iteration the algorithm proceeds by assessing a new solution in design space (as long as the exclusive refinement phase of the algorithm has not been reached). Two solutions are selected at random from the estimated Pareto set of the solutions evaluated so far (line 7). These solutions are then crossed over with probability p_{cross}, using simulated binary crossover [41], to create a single child (line 9). This child, $\mathbf{x}$, (or clone of an individual parent if no crossover

```
Require: $r$                                   Number of random locations to initially sample
Require: $g$                                   Total number of function evaluations
Require: $p_{cross}$                           Probability of crossover occurring
Require: $k$                                   Number of archive resamples per iteration
Require: $z$                                   Proportion of run to be spent solely refining the archive
 1: $X := \texttt{initial\_samples}(r)$          Generate $r$ random feasible solutions
 2: $\mathbf{Y} := \texttt{evaluate\_samples}(X)$          Evaluate members of the initial set just once each
 3: $\{A, \mathbf{F}\} := \texttt{get\_front}(X, \mathbf{Y})$    Extract estimated Pareto set and associated objective evaluations and track dominators
 4: $t := r$                                   Track number of evaluations taken thus far
 5: while $t < g$ : do
 6:    if $t < (1 - z)g$ then
 7:       $\{\mathbf{v}, \mathbf{u}\} := \texttt{sample\_leading\_edge}(A, \text{est}(\mathbf{F}))$          Sample from the estimated Pareto set
 8:       if $uniform\_rand() < p_{cross}$ then
 9:          $\mathbf{x} := \texttt{crossover}(\mathbf{v}, \mathbf{u})$          Create a single child via simulated binary crossover
10:       else
11:          $\mathbf{x} := \mathbf{v}$          If no crossover, then first parent is copied for later mutation
12:       end if
13:       $\mathbf{x}' := \texttt{vary}(\mathbf{x})$          Mutate $\mathbf{x}$
14:       $\mathbf{y}' := \texttt{evaluate}(\mathbf{x}')$          Evaluate child solution (single evaluation, so $\mathbf{y}'$ and $\text{est}(Y(\mathbf{x}'))$ are identical)
15:       $t := t + 1$
16:       if $\text{est}(\mathbf{F}) \not\prec \mathbf{y}(\mathbf{x}')$ :  then
17:          $\{A, \mathbf{F}, X, \mathbf{Y}\} := \texttt{update\_front}(A, \mathbf{F}, \mathbf{x}', \{\mathbf{y}'\}, X, \mathbf{Y})$          Update archive and search population
18:       else
19:          $\texttt{track\_dominator}(\texttt{A}, \mathbf{F}, \mathbf{x}', \{\mathbf{y}'\})$          Track a single dominating member of current $A$ for $\mathbf{x}'$
20:       end if
21:    end if
22:    for $k$ iterations do
23:       $\mathbf{x}'' := \text{argmin}\{|Y(\mathbf{x})| \text{ for } \mathbf{x} \in A\ \}$          Element with fewest samples
24:       $\{X_{\mathbf{x}''}, \mathbf{Y}_{\mathbf{x}''}\} := \texttt{remove\_tracking}(\mathbf{x}'', Y(\mathbf{x}''), X, \mathbf{Y})$          Get members which have $\mathbf{x}''$ as their dominator
25:       $\mathbf{y}'' := \texttt{evaluate}(\mathbf{x}'')$          Evaluate $\mathbf{x}''$ an extra time
26:       $Y(\mathbf{x}'') := Y(\mathbf{x}'') \cup \{\mathbf{y}''\}$          Update store of evaluations associated with $\mathbf{x}''$, thereby changing $\text{est}(Y(\mathbf{x}''))$
27:       $t := t + 1$
28:       $A := A \setminus \{\mathbf{x}''\}$          Remove $\mathbf{x}''$ from $A$
29:       $\mathbf{F} := \mathbf{F} \setminus \{Y(\mathbf{x}'')\}$          Remove evaluations of $\mathbf{x}''$ from $\mathbf{F}$
30:       $\{A, \mathbf{F}, X, \mathbf{Y}\} := \texttt{update\_front}(A, \mathbf{F}, \mathbf{x}'', Y(\mathbf{x}''), X, \mathbf{Y})$   If $\mathbf{x}''$ still on leading edge, put in $A$, otherwise put in $X$
31:       for each $\mathbf{x}$ and $Y(\mathbf{x})$ pair in $X_{\mathbf{x}''}$ and $\mathbf{Y}_{\mathbf{x}''}$ do
32:          $\{A, \mathbf{F}, X, \mathbf{Y}\} := \texttt{update\_front}(A, \mathbf{F}, \mathbf{x}, Y(\mathbf{x}), X, \mathbf{Y})$ See if previously dominated solutions should now be in $A$
33:       end for
34:    end for
35: end while
```

Figure 5. The rolling-tide evolutionary algorithm (RTEA).

occurs) is mutated, creating $\mathbf{x}'$ (line 13). Although we use simulated binary crossover and single element mutation here, other genetic operators could be used.

After its generation, $\mathbf{x}'$ is evaluated once and if it is not dominated by the current estimated Pareto set (based on comparing $\mathbf{y}'$ with $\text{est}(Y(\mathbf{x}))$ for $\mathbf{x} \in A$), it is added to the archive (lines 16 and 17), and any elements in A which are dominated by $\mathbf{x}'$ (based on their $\text{est}(Y(\mathbf{x}))$) are returned to X with $\mathbf{x}'$ recorded as their single tracked *dominator*. If however $\mathbf{x}'$ is dominated by the estimated Pareto front, then a single member of A is chosen as the single dominator for $\mathbf{x}'$ (line 19), from the members of A which dominate $\mathbf{x}'$. After this, the member of the elite archive with the fewest number of function evaluations, $\mathbf{x}''$, is selected and reevaluated an additional time (lines 23, 25 and 26). Prior to reevaluation, all solutions in X which have the selected archive member $\mathbf{x}''$ recorded as their single dominator are extracted (line 23); this set is denoted $X_{\mathbf{x}''}$.

Lines 28-30 ensure that if the reevaluation of $\mathbf{x}''$ means that the solution should no longer be in A, it is removed from A and placed in X. Likewise, lines 31 and 32 ensure that if solutions are no longer dominated by an element of A, they are transferred into A. This may be necessary because, for example, the reevaluated solution's new $\text{est}(Y(\mathbf{x}''))$ means that it now dominates or is dominated by other members of A. An illustration of possible movements of $\text{est}(Y(\mathbf{x}''))$ and implications for the management of A and X is shown in Figure 6. On line 32 elements which are not inserted into A

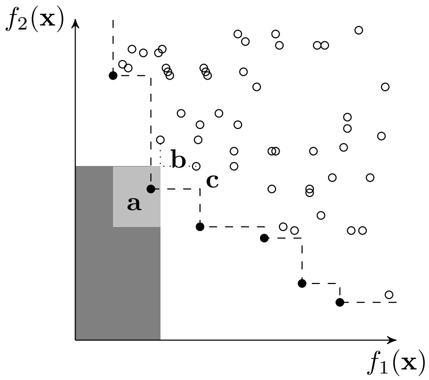

Figure 6. Potential effects when reevaluating a member of A. The estimated objective values $\mathrm{est}(Y(\mathbf{x}))$ of elements of A are shown with filled black circles and estimated objective values of elements of X are shown with unfilled circles. The (estimated) attainment surface is plotted with a dashed line. If on reevaluation, the solution marked $\mathbf{a}$ shifts in its $\mathrm{est}(Y(\mathbf{a}))$ position to any location in the light-grey region, A and X will be unchanged. If it moves to any location in the dark-grey region, the membership of A will be reduced because it now is estimated to dominate other solutions in A, but there will be no new entrants to A from X. Movement to any other location will cause $\mathbf{b}$ and/or $\mathbf{c}$ to become non-dominated by A and one or both of them enter A from X, as well as potentially causing the removal of other members of A (including $\mathbf{a}$ itself).

remain in X, and a dominating member of $A \cup X_{\mathbf{x}''}$ is set as their single tracked dominator.

It is the k reevaluations of leading edge solutions each iteration which gives the algorithm its *rolling tide* name. The effect is to constantly refine the estimate of the leading edge of solutions evaluated thus far. As the search progresses the proposal of new solutions drives the elite archive forward, but the reevaluations pull solutions away from the leading edge (or move them around), leaving others that had previously been dominated to take their place. This differs from e.g. [22], where elite members from the archive may be rejected on reevaluation, but previous solutions are not stored beyond the fixed archive size, and so cannot be pulled back in. The rolling tide approach also has the benefit that although we use a single initial evaluation for search points, the final estimated non-dominated archive tends to have many reevalutions per member and therefore higher confidence in their $\mathrm{est}(Y(\mathbf{x}))$ being a good estimate of the underlying true objective values.

Finally, RTEA incorporates a refinement phase, in which the search aspect of RTEA is switched off during the final stages of a run, so the algorithm focuses exclusively on increasing the accuracy of the archive. During the refinement phase lines 7-21 are not executed.

A. *Efficiently maintaining all evaluations*

Since the estimated objectives associated with solutions are uncertain RTEA keeps all solutions that have been evaluated because as solutions are re-evaluated their new estimated objectives may warrant including them in A. As pointed out above, re-evaluation of solutions means that solutions can move back and forth between X and A. Retaining a large number of solutions in X is potentially computationally expensive because, on the face of it, when a solution in A is reevaluated it has to be compared with all other solutions in A and all those which it previously dominated to discover whether it should still be a member of A. In practice however the computational overhead may be vastly reduced by maintaining a single reference for each member of X to another member in X or A which dominates it [42]. A member of X need only be compared to A again if its single tracked dominator is the member of A being reevaluated, because if the tracked dominator is not reevaluated, then the tracked dominator solution will *still* dominate the member of X at the current iteration.

In the worst case situation $|X_{\mathbf{x}''}|$ (the number of solutions which have a resampled member of A, $\mathbf{x}''$, noted as their single tracked dominator) could equal *all* the members of X that $\mathbf{x}''$ previously dominated, making the $|A| \times |X_{\mathbf{x}''}|$ dominance comparisons required computationally punitive. *However*, in practise $|X_{\mathbf{x}''}|$ averages between 3 or 4, as most often the noted dominator for a solution will end up residing in X (as search progresses) rather than A. See [42] for complexity derivations and empirical assessments on both synthetic data and optimiser runs.

VI. Empirical comparison of RTEA with noise-tolerant MOEAs

In this section we investigate the performance of RTEA and a selection of state-of-the-art noise-tolerant multi-objective optimisers, both for noise whose variance is constant and for noise whose variance varies during the optimisation and over space. The MOEAs compared are:

- The Bayesian (1+1)–ES using probabilistic dominance which we proposed in [6] (labelled BES here).
- The MOEA-RF optimiser of Goh & Tan [16], [17].
- The algorithm of Syberfeldt *et al.* [21] which is designed to cope with variable noise. We follow the practice of [21] and do not incorporate the optional surrogate which they advise for expensive real-world problems, but do not use in their evaluations on cheap-to-compute test problems. We label the algorithm MOP-EA here.
- The NMOE-AS optimiser of Park & Ryu [22].
- The RTEA optimiser.

As in Section III-B, results are presented on test problems UP1-10 from the CEC'09 test suite [38]. Each algorithm is run 30 times on each of the test problems, for a maximum of 300000 function evaluations. Algorithm parameters used are as suggested by the algorithm proposers [6], [16], [21], [22].[3]

[3] For MOEA-RF Goh & Tan recommend repeatedly evaluating a single point to estimate the noise width L at the start of an optimisation. The recommended number of samples is not provided in [16], [17], so here 100 samples are used as a reasonable but not excessive number. The values for the reliability threshold N_r and α reweighting term for the crowding distance used in NMOE-AS are not reported in [22], however the values of $N_r = 16$ and $\alpha = 3$ are used here, after personal correspondence with Dr Park.

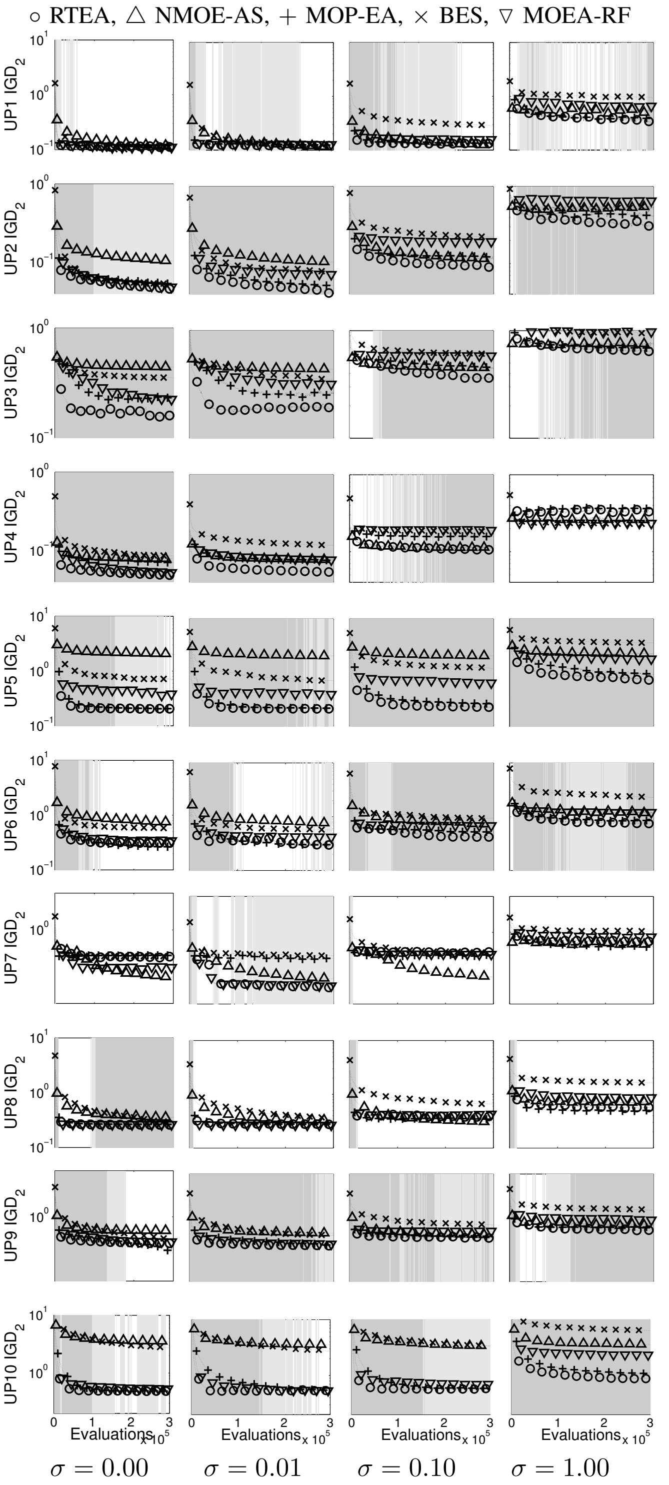

Figure 7. IGD_2 of archived solutions, reevaluated on noiseless cost function on UP1-10; 30 runs, plotted on a log scale. *Left to Right:* $\sigma = \{0.00, 0.01, 0.1, 1.0\}$. *Top to bottom:* UP1-10. A light grey background indicates when the median performance of RTEA is better than all of the other noise-tolerant algorithms, and a darker grey background denotes when it is *significantly* better.

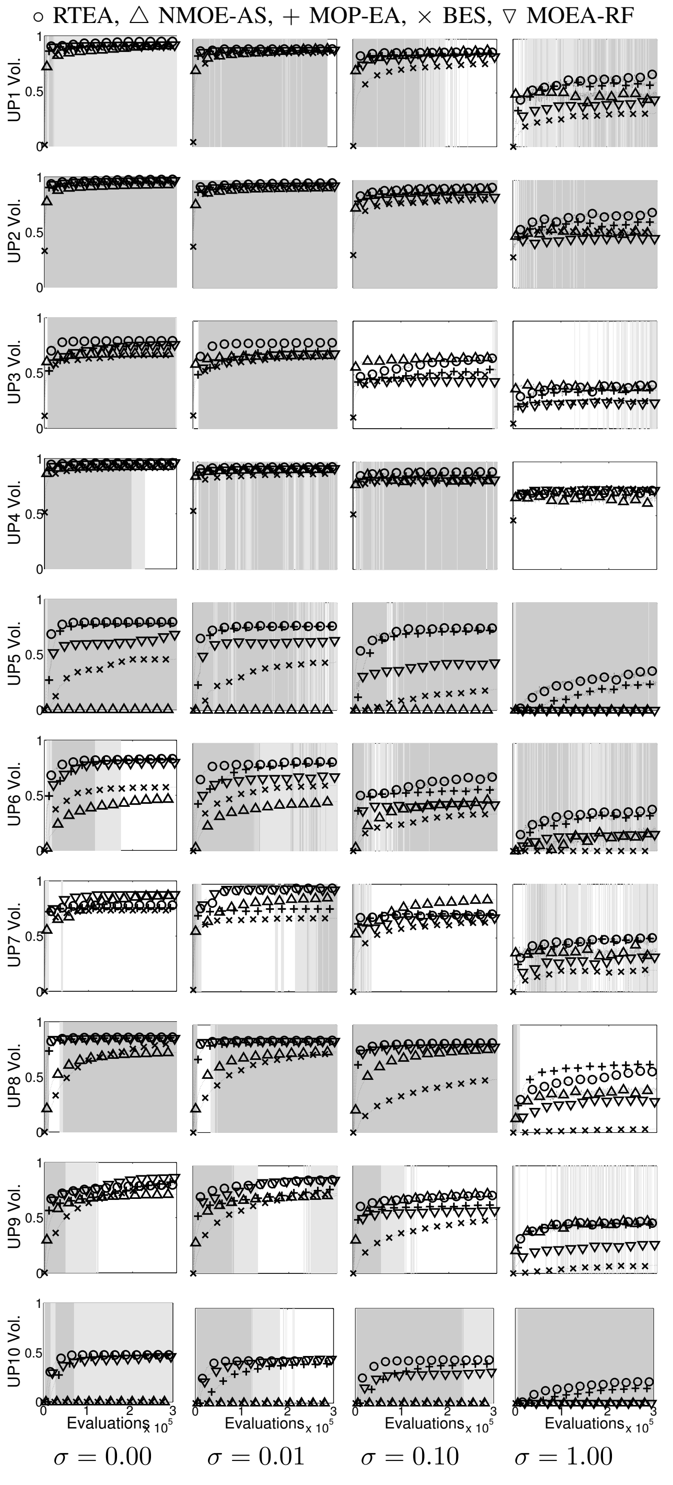

Figure 8. Hypervolume of archived solutions, reevaluated using the noiseless cost function on UP1-10; 30 runs. *Left to Right:* $\sigma = \{0.00, 0.01, 0.1, 1.0\}$. *Top to bottom:* UP1-10. Reference points are $(2, 2)$ and $(2, 2, 2)$ for two and three-objective problems respectively. A light grey background indicates when the median performance of RTEA is better than all of the other noise-tolerant algorithms, and a darker grey background denotes when it is *significantly* better.

Here RTEA is started with $r = 100$ initial random solutions, the mutation probability is set at $1/|\mathbf{x}|$, mutation width = 0.2, crossover probability = 0.8, crossover type = SBX, additional resamples per iteration $k = 1$, refinement phase $z = 5\%$ and $\text{est}(Y(\mathbf{x})) \equiv \text{mean}(Y(\mathbf{x}))$.

A. Static noise

In our first set of experiments we use static Gaussian noise in (3), as used in the experiments with standard MOEAs in Section III-B. We compare the performance of the algorithms for four noise magnitudes: $\sigma = \{0.00, 0.01, 0.10, 1.00\}$ and, as before, use the power mean IGD_2, the hypervolume and the NM measure to assess the performance of the algorithms.

The significance of the comparisons is assessed using the single-tailed non-parametric Mann-Whitney U test at the 5% level. In the plots we present a region with a light grey shading when the median result from RTEA is better than the other noise-tolerant algorithms, and darker grey when it is significantly outperforming all of them, as assessed with paired comparisons using the significance test.

Figure 7 shows the IGD_2 results and Figure 8 shows the hypervolume results. RTEA can be seen to be very competitive compared to the other noise-tolerant MOEAs when considering the hypervolume, rapidly outperforming the other algorithms across the majority of problems and noise levels, performing better than the other four noise-tolerant MOEAs across problems and noise levels 75% of the time. Assessment with the IGD_2 measure shows similar performance, with RTEA performing better than the other noise-tolerant MOEAs 71% of the time. On the NM measure, the NMOE-AS algorithm (which also resamples its archive members) tends to do better on the lower noise levels, with RTEA doing better on the higher noise levels, and both outperform the other noise-tolerant algorithms on this measure. Table I compares *all* the algorithms across UP1-10 for each noise level, and presents the percentage (averaged across the length of a run) for which the median performance of one algorithm is better (and significantly better) than all the others, averaged across the ten test problems, for each noise level.

It is interesting to note that the 'standard' MOEAs actually perform better than some of the noise-tolerant optimisers, not just in the noiseless case, but also at very low noise levels. In these situations the search/archive populations become slightly more diverse for standard MOEAs, but still hold 'good' solutions, meaning the noise-tolerant MOEAs only gain an advantage when the estimated Pareto set is well-converged. Also, many of the noise-tolerant MOEAs were previously assessed only on the ZDT test functions and it is possible that the more difficult UP1-10 problems may demand more sophisticated search and diversity management techniques than they provide. Finally, many all of the noise-tolerant MOEAs expend a substantial proportion of their allotted function evaluations *re*-evaluating a previously visited location, rather than searching for new samples, which may affect convergence speed. Nevertheless, RTEA is seen to perform better on average on IGD_2 when $\sigma \geq 0.01$, and hypervolume when $\sigma \geq 0.10$, although, as mentioned NMOE-AS does better on the NM measure for lower σ.

Figure 9 shows the median over 30 runs of the NM measure. For those algorithms which resample (RTEA, NMOE-AS, BES and MOP-EA) $\text{est}(Y(\mathbf{x}))$ is the mean of the samples at a location $\mathbf{x}$. We note that for MOEA-RF, which makes only a single evaluation at each design, $\text{est}(Y(\mathbf{x})) = \mathbf{y}$. NMOEA-AS and RTEA can be seen to *improve* the measure as the search continues. In contrast, for a number of problems the other algorithms become worse according to NM as search progresses. Note also the effect of the final 5% of the RTEA runs: when searching is discontinued and the archive further refined with resamples it is clear from Figure 9 that the

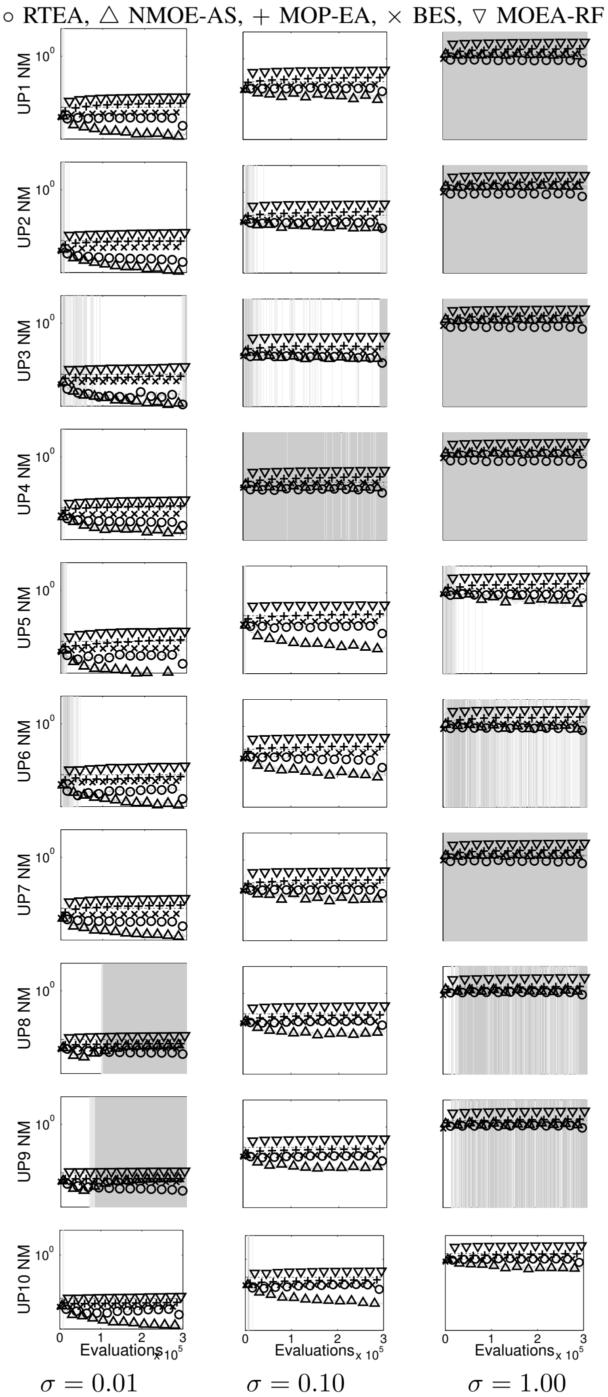

Figure 9. NM measure (plotted on the log scale) of archived solutions on noise injected UP1-10; 30 runs. *Left to Right:* $\sigma = \{0.01, 0.1, 1.0\}$. *Top to bottom:* UP1-10. A light grey background indicates when the median performance of RTEA is better than all of the other noise-tolerant algorithms, and a darker grey background denotes when it is *significantly* better.

Table I
PERCENTAGE OF TOTAL RUN LENGTH THAT THE MEDIAN PERFORMANCE OF EACH ALGORITHM WAS BETTER THAN THE OTHER EIGHT, AVERAGED OVER UP1-10, ON STATIC NOISE TYPES WITH $\sigma = \{0.00, 0.01, 0.10.1.00\}$. PERCENTAGE SIGNIFICANTLY BETTER GIVEN IN PARENTHESIS. BOLD INDICATES LARGEST PERCENTAGE. VALUES ROUNDED TO NEAREST WHOLE NUMBER.

	Hypervolume				IGD_2				NM			
σ	0.00	0.01	0.10	1.00	0.00	0.01	0.10	1.00	0.00	0.01	0.10	1.00
RTEA	17 (10)	32 (24)	**56** (**48**)	**53** (**37**)	10 (4)	**46** (**28**)	**73** (**51**)	**56** (**41**)	-	19 (16)	15(12)	**77** (**69**)
MOP-EA	0 (0)	0 (0)	1 (0)	12 (9)	8 (2)	0 (0)	6 (4)	18 (9)	-	0 (0)	0 (0)	0 (0)
BES	0 (0)	0 (0)	0 (0)	4 (0)	8 (0)	0 (0)	0 (0)	3 (0)	-	1 (1)	1 (1)	2 (1)
MOEA-RF	9 (1)	6 (0)	0 (0)	1 (0)	3 (0)	3 (0)	0 (0)	7 (2)	-	0 (0)	0 (0)	0 (0)
NMOE-AS	0 (0)	0 (0)	23 (4)	3 (0)	6 (0)	2 (1)	19 (10)	1 (0)	-	**79** (**77**)	**84** (**81**)	21 (18)
$\text{IBEA}_{\epsilon+}$	31 (6)	13 (2)	9 (2)	12 (2)	11 (0)	4 (0)	2 (0)	9 (2)	-	0 (0)	0 (0)	0 (0)
SPEA2	**38** (**23**)	**49** (**20**)	1 (0)	2 (0)	**53** (**50**)	45 (23)	1 (0)	6 (0)	-	0 (0)	0 (0)	0 (0)
NSGA-II	3 (0)	0 (0)	14 (9)	12 (10)	0 (0)	0 (0)	0 (0)	0 (0)	-	0 (0)	0 (0)	0 (0)
PAES	1 (1)	0 (0)	0 (0)	0 (0)	2 (1)	0 (0)	0 (0)	0 (0)	-	0 (0)	0 (0)	0 (0)

Table II
PERCENTAGE OF TOTAL RUN LENGTH THAT THE MEDIAN PERFORMANCE OF EACH ALGORITHM WAS BETTER THAN THE OTHER EIGHT, AVERAGED OVER UP1-10, ON VARIABLE NOISE TYPES. PERCENTAGE SIGNIFICANTLY BETTER GIVEN IN PARENTHESIS. BOLD INDICATES LARGEST PERCENTAGE. VALUES ROUNDED TO NEAREST WHOLE NUMBER.

	Hypervolume			IGD_2			NM		
Noise type	$\mathbf{y}$	$\mathbf{x}$	t	$\mathbf{y}$	$\mathbf{x}$	t	$\mathbf{y}$	$\mathbf{x}$	t
RTEA	**45** (**28**)	**54** (**40**)	**49** (**26**)	**62** (**45**)	**70** (**57**)	**60** (**38**)	12 (11)	31 (**31**)	4 (3)
MOP-EA	15 (3)	10 (9)	0 (0)	12 (3)	11 (9)	0 (0)	25 (20)	**34** (28)	4 (0)
BES	1 (0)	0 (0)	0 (0)	5 (0)	0 (0)	10 (10)	0 (0)	0 (0)	1 (0)
MOEA-RF	0 (0)	0 (0)	0 (0)	5 (1)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
NMOE-AS	33 (23)	35 (26)	14 (7)	2 (1)	8 (3)	0 (0)	6 (1)	14 (8)	0 (0)
$\text{IBEA}_{\epsilon+}$	4 (0)	0 (0)	10 (0)	10 (3)	0 (0)	6 (0)	0 (0)	0 (0)	0 (0)
SPEA2	1 (0)	0 (0)	17 (0)	1 (0)	0 (0)	24 (0)	**39** (**24**)	0 (0)	**75** (**69**)
NSGA-II	2 (0)	1 (0)	10 (9)	3 (0)	10 (10)	0 (0)	18 (0)	20 (20)	20 (20)
PAES	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)

NM measure decreases sharply in this final phase of the optimisation run. Nonetheless, even in the general search (during the first 95% of the optimisation) the discrepancy between the true objectives and the estimated objectives is smaller for RTEA than other algorithms, bar NMOEA-AS, due to its iterative re-estimation of the leading part of the front. These more accurate estimates of the front's location in turn allow faster convergence towards the true front.

B. Variable noise

In the previous experiments the noise was static. We now investigate the effect of different forms of noise which varies with objective space location or the duration of the optimisation. We modify the CEC'09 test functions to have noisy objectives, via (3), in three ways. First, the noise variance depends on the objective space location of the solution: $\epsilon_d \sim \mathcal{N}(0, f_d(\mathbf{x}))$. Second, the noise depends on the design space location of the solution: $\epsilon_d \sim \mathcal{N}\left(0, (0.1||\mathbf{x}||_1)^2\right)$. Of course, noise that varies with design space location can be regarded as a function of objective space location and *vice versa*. Nonetheless, the quality of the noise variation will be different and, in general, noise that varies as a function of design space is likely to vary more rapidly in objective space than noise that is a function of objective values. Finally, the noise is made temporally dependent noise: $\epsilon_d \sim \mathcal{N}(0, \omega_{d,t}^2)$, where $\omega_{d,t}$ is generated via a filtered random walk for each objective independently, with t denoting the time step of the walk. $\omega_{d,t}$ is generated according to the following nonlinear autoregressive process:

$$\omega_{d,t} = \begin{cases} 0.1, & \text{if } t = 1 \\ |\omega_{d,t-1} + \xi_d|, & \text{otherwise} \end{cases} \tag{11}$$

with $\xi_d \sim \mathcal{N}(0, 0.01^2)$. In the case of temporal noise, we generate a single set of 30 random walks using (11), and use these 30 for each algorithm so that each algorithm encounters the same temporal noise as its comparators.

The BES algorithm incorporates a term η which progressively down-weights the influence of historically observed noise in the Bayesian update scheme in order to cope with *temporally* varying noise [6]. We can also apply BES to the spatially-varying noise problems, because, since BES is a point-based optimiser, spatially varying noise is equivalent to temporally varying noise, although with high frequency variations. We use $\eta = 0.95$ here for the temporally varying noise and for the noise dependent on objective space location, and $\eta = 0.75$ for the noise dependent on design space location which changes more rapidly.

Since MOP-EA and NMOE-AS continuously re-estimate the noise by resampling, they can also be applied to situations where the noise varies spatially, as they effectively maintain a noise estimate at each design space location. Although a stationary, zero mean, Gaussian distributed model is no longer accurate, [21] still recommends its use when the noise is non-Gaussian, although the accuracy of the estimated confidence intervals will be affected. We therefore also apply it to the temporally varying noise problem. [22] adjusts the probability

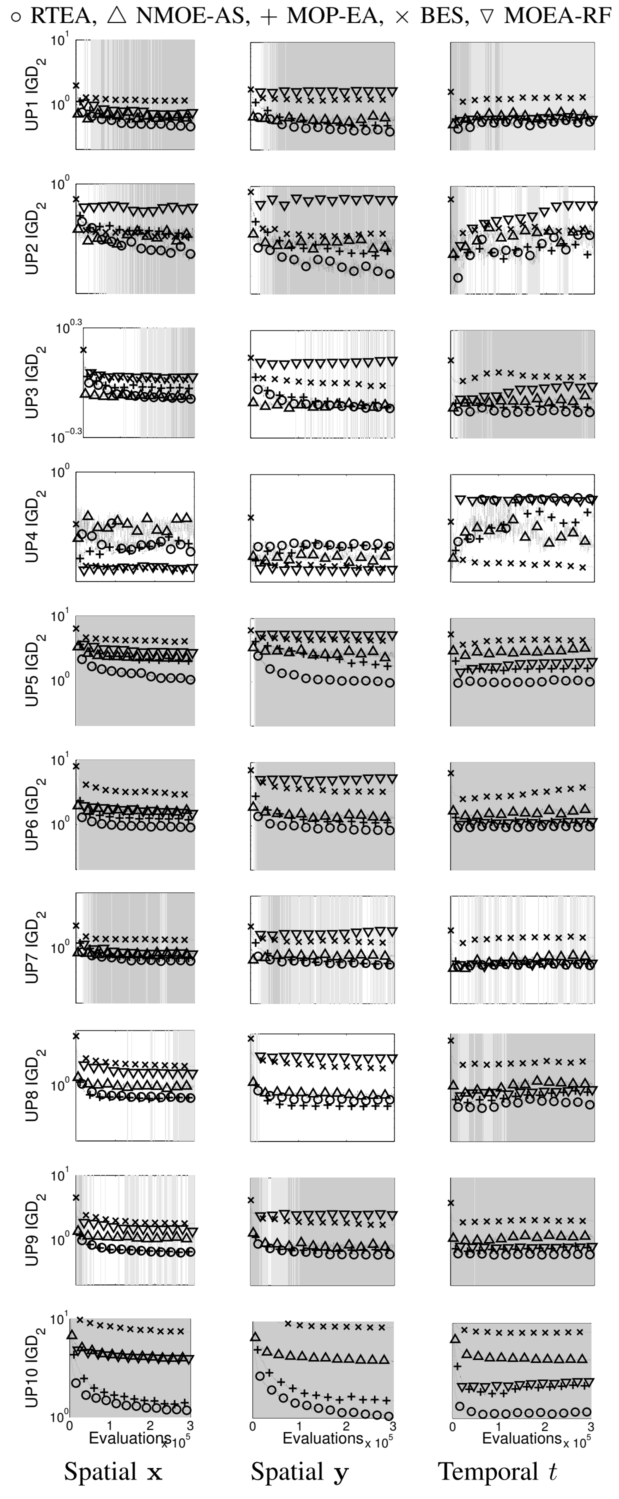

Figure 10. IGD_2 of archived solutions on varying noise problems, reevaluated on noiseless cost function on UP1-10; 30 runs, plotted on a log scale. *Left to Right:* Spatial $\mathbf{x}$, spatial $\mathbf{y}$, temporal. *Top to bottom:* UP1-10. A light grey background indicates when the median performance of RTEA is better than all of the other noise-tolerant algorithms, and a darker grey background denotes when it is *significantly* better.

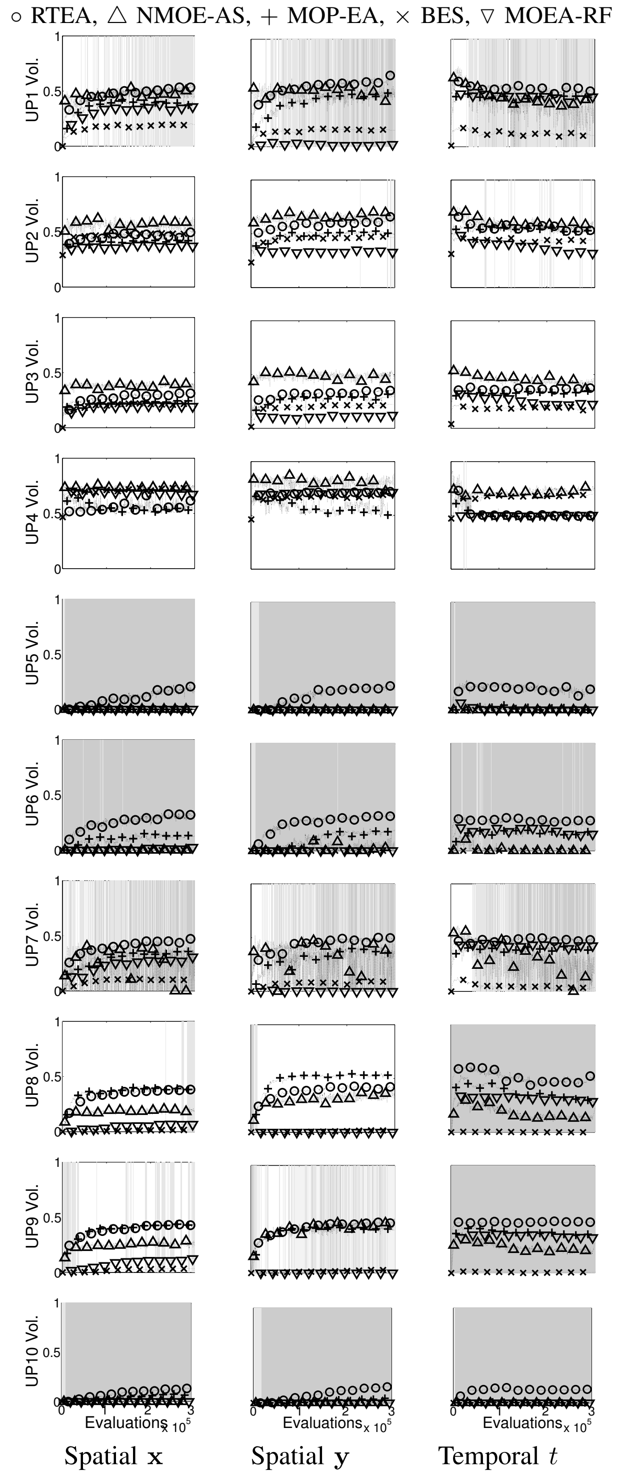

Figure 11. Hypervolume of archived solutions on varying noise problems, reevaluated on noiseless cost function on UP1-10; 30 runs. *Left to Right:* Spatial $\mathbf{x}$, spatial $\mathbf{y}$, temporal. *Top to bottom:* UP1-10. Reference points used in the hypervolume calculation are (2, 2) and (2, 2, 2), for two-objective and three-objective test problems respectively. A light grey background indicates when the median performance of RTEA is better than all of the other noise-tolerant algorithms, and a darker grey background denotes when it is *significantly* better.

of dominance calculation when the noise variance is assumed to vary at different locations.

Like a number of noise-tolerant optimisers, MOEA-RF is designed for situations with a fixed noise width and requires the noise variance to be specified as a parameter of the

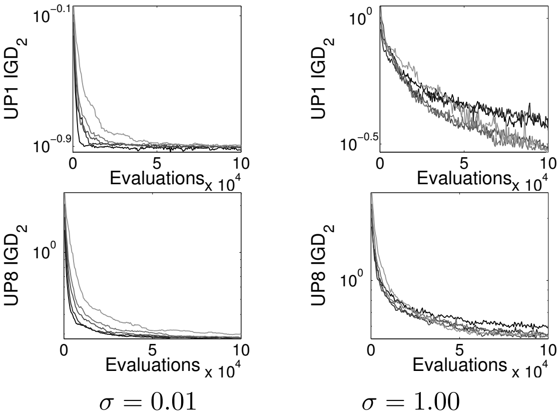

Figure 12. IGD_2 of RTEA with $k = \{1, 2, 3, 5, 10\}$. UP1, UP8, $\sigma = \{0.01, 1.00\}$.

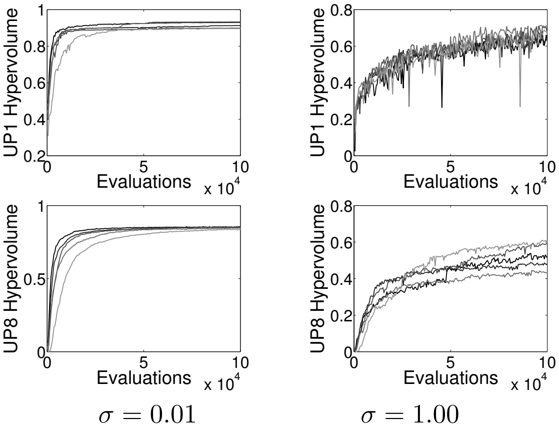

Figure 13. Hypervolume of RTEA with $k = \{1, 2, 3, 5, 10\}$. UP1, UP8, $\sigma = \{0.01, 1.00\}$.

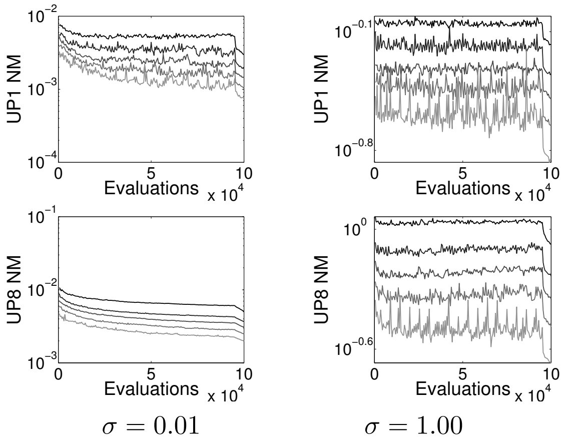

Figure 14. NM of RTEA with $k = \{1, 2, 3, 5, 10\}$. UP1, UP8, $\sigma = \{0.01, 1.00\}$.

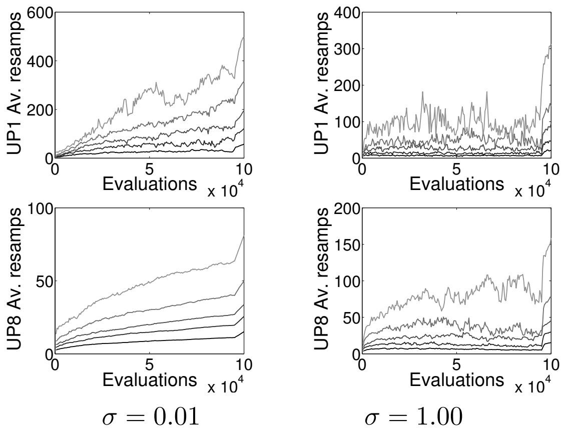

Figure 15. Average number of resamples of archive members in RTEA with $k = \{1, 2, 3, 5, 10\}$. UP1, UP8, $\sigma = \{0.01, 1.00\}$.

algorithm. Consequently we anticipate that the algorithm will perform relatively poorly on varying noise problems, however we include the results here for completeness.

Figures 10 and 11 show the results using the IGD$_2$ and hypervolume. A similar ordering of algorithms is seen as with the static noise, with RTEA performing better than the other four noise-tolerant MOEAs across problems and noise types on the hypervolume measure 58% of the time; and on the IGD$_2$ measure 73% of the time.

Table II compares *all* the algorithms across UP1-10 for each type of variable noise. When the noise varies according to $\mathbf{y}$ or $\mathbf{x}$ RTEA and NMOE-AS perform better than all other algorithms on the hypervolume measure, although RTEA tends to do much better on IGD$_2$ than NMOE-AS. When the noise varies over time NMOE-AS performs markedly less well, although it remains in second place. The NM measure is not well-correlated with the other two quality measures on variable noise, but this is unsurprising — for many of the problems the closer solutions are to the Pareto set (or sub regions thereof) the larger the noise intensity they experience, and therefore the higher the NM that would be expected.

C. Sensitivity analysis of RTEA

The lower the number of k resamples of archive members taken at each iteration, the more evaluations are available to RTEA for searching new locations (given a limited number of total function evaluations). But the archived solutions, from which the new solutions are evolved, will have estimated objective values of lower accuracy. Here we perform a sensitivity analysis of RTEA with various numbers of resamples taken of archive members at each iteration. We run RTEA 30 times for 100000 function evaluations on the two-objective CEC'09 problem UP1 and the three-objective CEC'09 problem UP8, with $k = \{1, 2, 3, 5, 10\}$. The corresponding median IGD$_2$, hypervolume and NM measure results of these experiments are shown in Figures 12-14, and the median number of resamples per archive member is shown in Figure 15, for $\sigma = \{0.01, 1.0\}$. The greyscale of the lines plotted correspond to the value of k, with $k = 1$ drawn with black and $k = 10$ with the lightest grey.

When the noise level is small, RTEA with lower k performs better than with higher k on IGD$_2$ and hypervolume. NM is always lower with higher k, irrespective of the noise level – this is as expected, because the larger k, the more resamples (on average) an archived solution will have (see Figure 15), and therefore the more accurate the $\texttt{est}(Y)$. For the large noise level smaller k performs better initially on IGD_2 and hypervolume, but is then rapidly overtaken by the RTEA algorithms using larger k. Based on these observations, there is potential in to developing a variant of RTEA in which k is adapted during the search, based on the noise properties encountered. However, as seen here, even with $k = 1$, RTEA performs competitively with other state-of-the-art approaches.

VII. Discussion

In this paper we have presented a novel algorithm for multi-objective optimisation with uncertain objectives, the rolling tide evolutionary algorithm (RTEA). RTEA can cope with noisy optimisation problems in which the noise characteristics may be fixed or variable. We make the minimal assumption that the noiseless objectives can be accurately estimated from a sufficiently large number of samples. The algorithm attempts to both harness the exploitation/exploration power of traditional EAs, whist simultaneously refining the accuracy of its active elite archive by constantly resampling the elite solutions, thus ensuring that 'high-quality' individuals feed into the evolutionary process. The approach also means that solutions are not 'lost' if another solution is thought to dominate them initially, but on reevaluation this is found to perform less well, thus allowing previously dominated solutions to be pulled into the archive at a later time step. If the algorithm is run with a single reevalution each iteration, roughly half of all function evaluations are at new locations and half are used in the reevaluation of previously evaluated locations, allowing RTEA to search to a much greater extent than most other noise-tolerant MOEAs.

We have shown that for even small values of noise the performance of standard MOEAs is significantly impaired, and we have also shown the ability of RTEA to cope well with noisy optimisation problems in comparison to a number of state-of-the-art noise-tolerant MOEAs. RTEA produces competitive results with state-of-the-art algorithms on a range of problem types, noise types and intensities. We find the next best competing algorithm is NMOE-AS, which like RTEA progressively resamples its archive (although without the ability to pull in previously dominated solutions to its archive). Progressive resampling therefore appears to be a very useful technique for noise-tolerant MOEAs

RTEA is simple to implement and also has the benefit of requiring relatively few parameters. It does rely on keeping all previously evaluated solutions (which has a storage overhead), however we have also discussed a method to ameliorate the cost associated with updating the elite archive in conjunction with these past solutions, based on tracking just a single dominating solution for each member of the non-elite population.

A potential extension of RTEA would be an automatic procedure for determining when to resample. Currently at each iteration a new location is evaluated and existing estimated elite solution(s) reevaluated. By tracking, for instance, the region dominated by the elite estimate over time the ratio of new location evaluations and resamples could be beneficially varied. When the elite front is advancing more new designs could be evaluated, whereas if the front is stalled or oscillating (due to reevaluations pulling the estimate back), then more resamples may be preferred. This would allow the algorithm to tune itself to different problem types and convergence stages.

Although RTEA has shown good performance on spatially varying noise, we believe there is still scope to improve performance on these types of problems further, by *learning* the spatially dependent characteristics of the noise during the search using, for instance, machine learning methods [10]. We look forward to the development of more sophisticated algorithms that can take advantage of learned characteristics to reduce the number of function evaluations required.

References

[1] I. Rechenberg, "Evolutionsstrategie – optimierung technischer systeme nach prinzipien der biologischen evolution," Ph.D. dissertation, Technical University of Berlin, Federal Republic of Germany, 1971.

[2] A. Thompson, "Hardware evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution," *Springer*, 1998.

[3] S. Harding and J. F. Miller, "Evolution *in materio*: initial experiments with liquid crystal," in *Proceedings of the NASA/DoD Workshop on Evolvable Hardware*. IEEE Computer Society, 2004, pp. 289–299.

[4] P. Stagge, "Averaging Efficiently in the Presence of Noise," in *Parallel Problem Solving from Nature*, A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds. Springer, 1998, pp. 188–200.

[5] D. Büche, P. Stoll, R. Dornberger, and P. Koumoutsakos, "Multiobjective evolutionary algorithm for optimization of noisy combustion processes," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 32, no. 4, pp. 460–473, 2002.

[6] J. Fieldsend and R. Everson, "Multi-objective optimisation in the presence of uncertainty," in *IEEE Congress on Evolutionary Computation*, 2005, pp. 243–250.

[7] M. Basseur and E. Zitzler, "A preliminary study on handling uncertainty in indicator-based multiobjective optimization," in *Applications of Evolutionary Computation. EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC*, ser. LNCS, vol. 3907. Springer, 2006, pp. 727–739.

[8] S. Rana, L. Whitley, and R. Cogswell, "Searching in the Presence of Noise," in *Parallel Problem Solving from Nature*. Springer Verlag, 1996, pp. 198–207.

[9] A. Di Pietro, L. While, and L. Barone, "Applying Evolutionary Algorithms to Problems with Noisy, Time-consuming Fitness Functions," in *IEEE Congress on Evolutionary Computation*, vol. 2. IEEE, 2004, pp. 1254–1261.

[10] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments-a survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.

[11] J. Horn and N. Nafpliotis, "Multiobjective Optimization Using the Niched Pareto Genetic Algorithm," Illinois Genetic Algorithms Laboratory (IlliGAL), University of Illinois at Urbana-Champaign, USA, Tech. Rep. 93005, July 1993.

[12] E. Hughes, "Constraint handling with uncertain and noisy multi-objective evolution," in *IEEE Congress on Evolutionary Computation*, 2001, pp. 963–970.

[13] J. Teich, “Pareto-front exploration with uncertain objectives,” in *Evolutionary Multi-Criterion Optimization*, ser. LNCS, E. Zitzler, K. Deb, L. Thiele, C. Coello Coello, and D. Corne, Eds., vol. 1993. Springer, 2001, pp. 314–328.

[14] E. Hughes, “Evolutionary multi-objective ranking with uncertainty and noise,” in *Evolutionary Multi-Criterion Optimization*, ser. LNCS, E. Zitzler, K. Deb, L. Thiele, C. Coello Coello, and D. Corne, Eds., vol. 1993. Springer, 2001, pp. 329–342.

[15] L. Bui, H. Abbass, and D. Essam, “Fitness Inheritance For Noisy Evolutionary Multi-Objective Optimization,” in *Proceeding of the Genetic and Evolutionary Computation Conference*, 2005, pp. 779–785.

[16] C.-K. Goh and K. Tan, “An investigation on noisy environments in evolutionary multiobjective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 3, pp. 354–381, 2007.

[17] ——, *Evolutionary Multi-objective Optimization in Uncertain Environments*. Springer, 2009.

[18] J. Knowles, D. Corne, and A. Reynolds, “Noisy multiobjective optimization on a budget of 250 evaluations,” in *Evolutionary Multi-Criterion Optimization*, M. Ehrgott, C. M. Fonseca, X. Gandibleux, J.-K. Hao, and M. Sevaux, Eds., 2009, pp. 36–50.

[19] F. Siegmund, “Sequential sampling in noisy multi-objective evolutionary optimization,” Master’s thesis, University of Skövde, School of Humanities and Informatics, Sweden, 2009.

[20] H. Eskandari and C. Geiger, “Evolutionary multiobjective optimization in noisy problem environments,” *Journal of Heuristics*, vol. 15, pp. 559–595, 2009.

[21] A. Syberfeldt, A. Ng, R. John, and P. Moore, “Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling,” *European Journal of Operational Research*, vol. 204, pp. 533–544, 2010.

[22] T. Park and K. Ryu, “Accumulative Sampling for Noisy Evolutionary Multi-Objective Optimization,” in *Proceeding of the Genetic and Evolutionary Computation Conference*, 2011, pp. 793–800.

[23] V. Shim, K. Tan, J. Chia, and A. Al Mamun, “Multi-objective Optimization with Estimation of Distribution Algorithm in a Noisy Environment,” *Evolutionary Computation*, vol. 21, no. 1, pp. 149–177, 2013.

[24] C. Villa, E. Lozinguez, and R. Labayrade, “Multi-objective Optimisation under Uncertain Objectives: Application to Engineering Design Problem,” in *Evolutionary Multi-Criterion Optimization*, ser. LNCS, R. Purshouse, P. Fleming, C. Fonseca, S. Greco, and J. Shaw, Eds., no. 7811. Springer, 2013, pp. 796–810.

[25] G. Schuëller and H. Jensen, “Computational methods in optimization considering uncertainties - An overview,” *Computer Methods in Applied Mechanics and Engineering*, vol. 198, pp. 2–13, 2008.

[26] K. Deb and H. Gupta, “Introducing Robustness in Multi-Objective Optimization,” *Evolutionary Computation*, vol. 14, no. 4, pp. 463–494, 2006.

[27] S. Yang, Y. Ong, and Y. Jin, *Evolutionary computation in dynamic and uncertain environments*. Springer, 2007.

[28] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, “A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II,” in *Parallel Problem Solving from Nature*. Springer, 2000, pp. 849–858.

[29] J. Knowles and D. Corne, “Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy,” *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[30] E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the Strength Pareto Evolutionary Algorithm,” Swiss Federal Institute of Technology Zurich (ETH), Tech. Rep. TIK-Report 103, May 2001.

[31] E. Zitzler and S. Künzli, “Indicator-Based Selection in Multiobjective Search,” in *Parallel Problem Solving from Nature*. Springer, 2004, pp. 832–842.

[32] J. Knowles, L. Thiele, and E. Zitzler, “A tutorial on the performance assessment of stochastic multiobjective optimizers,” Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland, Tech. Rep. 214, 2006.

[33] E. Zitzler and L. Thiele, “Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[34] T. Hanne, “On the convergence of multi objective evolutionary algorithms,” *European Journal of Operational Research*, vol. 117, pp. 553–564, 1999.

[35] J. Fieldsend, R. Everson, and S. Singh, “Using Unconstrained Elite Archives for Multi-Objective Optimisation,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, pp. 305–323, 2003.

[36] C. Coello, G. Lamont, and D. Van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems*, 5th ed. Springer, 2007.

[37] 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*, vol. 16, no. 4, pp. 504–522, 2012.

[38] Q. Zhang, A. Zhou, S. Zhao, P. Suganthan, W. Liu, and S. Tiwari, “Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition,” School of Computer Science and Electronic Engineering, University of Essex, UK, Tech. Rep. CES-487, April 2009.

[39] E. Zitzler, K. Deb, and L. Thiele, “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results,” *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[40] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca, “Performance Assessment of Multiobjective Optimizers: An Analysis and Review,” Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland, Tech. Rep. 139, Jun. 2002.

[41] K. Deb and R. Agrawal, “Simulated binary crossover for continuous search space,” *Complex Systems*, vol. 9, pp. 115–148, 1995.

[42] J. Fieldsend and R. Everson, “On the efficient maintenance and updating of Pareto solutions when assigned objectives values may change,” Computer Science, University of Exeter, UK, Tech. Rep., December 2013.

Jonathan E. Fieldsend (S’00-M’02) received the BA degree in Economics from the University of Durham, Durham, UK, in 1998, the MSc degree in Computational Intelligence at the University of Plymouth. Plymouth, UK in 1999 and a PhD in Computer Science from the University of Exeter, Exeter, UK, in 2003.

He is currently a Lecturer in Computer Science at the University of Exeter. His research interests include multi-objective optimisation, data visualisation, Bayesian methods and machine learning.

Richard M. Everson received a degree in Physics from Cambridge University, Cambridge, UK, and a PhD degree in Applied Mathematics from Leeds University, Leeds, UK.

He worked at Brown University, Providence, RI, and Yale University, New Haven, CT, on fluid mechanics and data analysis problems until moving to Rockefeller University, NY, to work on optical imaging and modelling of the visual cortex. He is a Professor of Machine Learning at the University of Exeter, Exeter, UK. His research interests lie in pattern recognition and machine learning especially applied to multi-objective optimisation and signal processing.