Constraint Handling With Uncertain and Noisy Multi-Objective Evolution

Evan J. Hughes
Department of Aerospace, Power and Sensors,
Cranfield University,
Royal Military College of Science,
Shrivenham, Swindon, England. SN6 8LA
ejhughes@iee.org

Abstract-

Many real world problems are constrained and have multiple objectives that must be satisfied. To compound the optimisation challenge, systems are often noisy or uncertain, leading to errors in the objective calculations. This paper develops theory to help reduce the effects of noise and uncertainty on constrained evolutionary optimisation processes. Experimental results are presented for generating Pareto surfaces with two different types of noise and also with constraints and designer preferences.

1 Introduction

Many practical problems have multiple objectives and are subject to constraints. Evolutionary methods have been shown [1] to be robust to noise in the optimisation process.

This robustness to errors has also been exploited by artificially adding noise to the objectives in an attempt to identify solutions that are robust to noise and uncertainty in the real system [2, 3, 4]. Noise is also often present when trying to optimise hardware systems such as in robotics. Noise or uncertainty in the objectives tend to slow evolution and reduce solution quality.

Attempts to reduce noise by repeating objective calculations and then averaging or combining results have been tried, but often with many realistic problems, the time to re-evaluate is prohibitive and often the number of samples used to average must be very small and therefore subject to considerable error. Most evolutionary algorithms to date have accepted these problems as the robustness of the algorithms allows small errors to be tolerated.

Therefore we may form two categories of problem:

- **Noisy:** Two successive evaluations of the same chromosome information return two different sets of objectives.
- **Uncertain:** Two successive evaluations of the same chromosome return the same objective values, but when comparing two different chromosomes, errors and approximations in the modelling may cause the objective values returned to classify the wrong solution as being superior.

Many engineering problems however may be both noisy and uncertain.

In the paper, section 2 establishes the theory for comparing two fitness measurements and section 3 details the new ranking process. Section 4 discusses how constraints and preferences may be applied, and section 5 gives the results of experiments to demonstrate the new algorithm. Section 6 concludes.

2 Comparing Objective Measurements

The fundamental operation in evolutionary algorithms is comparing two sets of objective measurements to see which solution is better. For simplicity, if the measurements are corrupted with zero mean Gaussian noise, we need to be able to assess the probability of the decision being correct.

One approach is to recognise that the difference between two Gaussian distributions is also Gaussian but with a mean value that is the difference between the means of the two distributions and a variance which is a sum of the two variances (Cramer's Theorem), i.e.,

$$N(\mu_a, \sigma_a^2) - N(\mu_b, \sigma_b^2) = N(\mu_a - \mu_b, \sigma_a^2 + \sigma_b^2) \quad (1)$$

If A dominates B in a maximisation sense, then the area under the resulting curve from zero to infinity will give the probability that the decision that A dominates B is correct. If we normalise B to give

$$\begin{aligned} N(\frac{\mu_a - \mu_b}{\sigma_b}, \frac{\sigma_a^2}{\sigma_b^2}) - N(0,1) &= N(\frac{\mu_a - \mu_b}{\sigma_b}, \frac{\sigma_a^2}{\sigma_b^2} + 1) \quad (2) \\ &= N(m, s^2 + 1) \quad (3) \end{aligned}$$

Where $m = \frac{(\mu_a - \mu_b)}{\sigma_b}$ and $s = \sigma_a / \sigma_b$. The probability of A dominating B in maximisation is then

$$\begin{aligned} P(A \text{ dom } B) &= \frac{1}{\sqrt{2\pi\,(s^2+1)}} \int_0^\infty e^{-\frac{(x-m)^2}{2(s^2+1)}} dx \quad (4) \\ &= \frac{1}{2} + \frac{1}{2}\,\text{erf}\left(\frac{m}{\sqrt{(2+2s^2)}}\right) \quad . \quad (5) \end{aligned}$$

Where $\text{erf}(u) = \frac{2}{\sqrt{\pi}} \int_0^u e^{-t^2} dt$.

2.1 Numerical Approximation of Probability

If we cannot afford to do multiple evaluations for each chromosome (often the case), we can choose a random chromosome before running the EA and perform multiple evaluations to estimate the noise standard deviation (and possibly noise distribution). If the noise statistics are known to be nonlinear, it may be advantageous to either re-estimate the statistics every few generations from an average chromosome, or even from the current population. When the same standard deviation is used for comparing two objective values, $\sigma_n = \sigma_a = \sigma_b$ therefore $s = 1$. Thus the probability is only determined by the value of m.

As the case of $s = 1$ is likely to be the most commonly used, we can tailor the equations specifically. Thus the probability of sample A dominating sample B in maximisation ($P(A > B)$) is

$$\begin{aligned} P(A > B) &= \frac{1}{2} + \frac{1}{2}\,\text{erf}\,(\frac{m}{2}) \\ &= \frac{1}{2} + \frac{1}{2}\,\text{erf}\,(\frac{A-B}{2\sigma_n})\ , \end{aligned} \qquad (6)$$

therefore if $A = 0$, $B = 5$ and $\sigma_n = 1$, $P(A > B) = 0$ as expected.

Unfortunately, the error function erf (x) is not easy to calculate quickly. Recognising that (5) is sigmoidal in shape, other standard sigmoidal curves have been fitted to give a good approximation to the curve, but allow the probability to be calculated quickly. Figure 1 shows the curve approximation and (7) & (8) show the equations. The results of the two different approximations are so similar to each other, they appear as a single line on the graph.

$$P(A > B) \approx \frac{1}{2}\,(1 + \tanh\frac{m}{0.8\sqrt{2+2s^2}}) \qquad (7)$$

$$P(A > B) \approx \frac{1}{1 + e^{-\frac{2.5\,m}{\sqrt{2+2s^2}}}} \qquad (8)$$

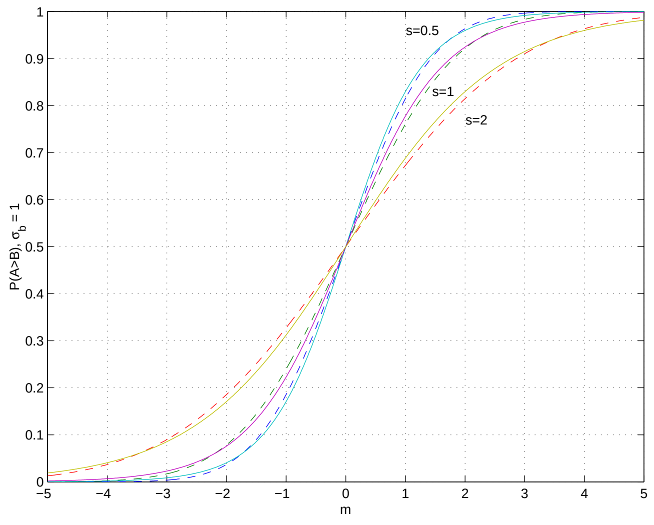

Figure 1: Approximation of $P(A > B)$ against m, dashed curve is (5)

The small errors in the approximation can be tolerated as a tradeoff for the speed gain. Further improvements in calculation speed may be obtained for certain problems by utilising the decomposition of $\tanh(A + B)$. This is detailed in [5].

Table 1: Ranks of example fitness values

Value	A	B	C	D	E	F	G
Rank	0	1	2	2	4	5	6

In practice, setting $\sigma_n = 0$ will give a divide by zero error, so this case must be trapped and with double precision numbers, $\sigma_n = 1E - 10$ is a good substitute.

If we have two, k objective, independent fitness measurements with corresponding objective values $A_1 \ldots A_k$, and $B_1 \ldots B_k$, the probabilities $P(A < B)$, $P(A > B)$, and non-domination or equivalence ($P(A \equiv B)$) are simply

$$\begin{aligned} P(A < B) &= \prod_{j=1}^{k} P(A_j < B_j) \\ P(A > B) &= \prod_{j=1}^{k} P(A_j > B_j) = \prod_{j=1}^{k}(1 - P(A_j < B_j)) \\ P(A \equiv B) &= 1 - P(A < B) - P(A > B)\ . \end{aligned} \qquad (9)$$

3 Probabilistic Ranking and Selection

3.1 Introduction

Ranking is often employed to prevent a superior solution dominating the early populations in the evolutionary process. The conventional ranking process, however, does not take the uncertainty in the measured fitness values into account. The following sections provide a fresh view of the ranking process and develop theory for multi-objective ranking of uncertain fitness measurements.

3.2 Single Objective Ranking

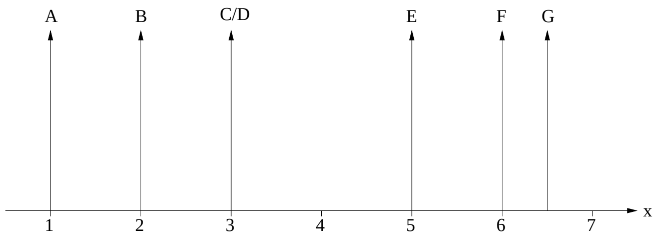

Figure 2: Fitness values to be ranked

Figure 2 shows seven fitness values to be ranked. If we are minimising, the best fitness value is the lowest. In the case shown, value *A* will get rank 0, and value *G* will be rank 6. Values *C* and *D* are equal and therefore should be assigned the same rank. We can assign rank values as shown in Table 1.

If we did not have a tie between *C* & *D*, we could use the linear selection equation (10) to calculate probabilities of selection, based on the ranked fitness, where n is the number of fitness values and R_i is the rank of individual i. The sum of the rank values on the denominator will sum to

$n(n-1)/2 = 21$ which is the sum of the arithmetic series zero to six, therefore the best individual will get a probability of selection of $2/n$ and the worst a probability of zero.

$$P(\text{select}_i) = \frac{(n-1) - R_i}{\sum_{j=1}^{n} R_j} = \frac{2((n-1) - R_i)}{n(n-1)} \quad (10)$$

If we use the rank values in Table 1 with both the tied fitness values being given the best 'untied' rank, we find that the sum of the ranks is no longer consistent, and in this case, $\sum_{j=1}^{n} R_j = 20$. Alternatively, as *C* & *D* are tied, it may be better to penalise them both a little and therefore take an average of the rank positions they could have shared, i.e., give them both a rank of 2.5. This would return the overall sum to be 21 and would be consistent, no matter how many fitness values share a rank. This is the method most used for ranking a vector of data.

We can view the ranking process as counting the number of fitnesses that dominate the fitness of interest [6]. If a fitness equal to the current one is encountered, then it is half dominating, and half dominated by the current fitness value. Therefore we can create the rank position numbers by this simple counting process. For example, *E* is dominated by *A*, *B*, *C* & *D* and therefore has a rank of 4. Value *C* is dominated by *A* & *B* but is tied with *D* and so gets a rank of 2.5.

Alternatively, we could consider the dominating / not dominating decision as being the *probability* that each fitness value dominates the value of interest. For example, if we consider fitness *C*, the probability that *A* dominates *C* is one. The probability that *G* dominates *C* is zero. The probability that *D* dominates *C*, from (5) with $m = 0$, is $P = 0.5$. Thus we can represent the rank position as the sum of probabilities of domination as shown in (11), where $P(F_j > F_i)$ is the probability that fitness value j dominates fitness value i.

$$R_i = \sum_{j=1}^{n} P(F_j > F_i)\bigg|_{i \neq j} \quad (11)$$

In (11), we have to be sure not to compare fitness F_i with itself. If we did, we would get an extra probability of 0.5 added to the sum. We can therefore include F_i in the sum, but subtract the effect of comparing the fitness with itself. This is shown in (12).

$$R_i = \sum_{j=1}^{n} P(F_j > F_i) - 0.5 \quad (12)$$

As (12) is based on probability, if the fitness values are uncertain, we can use (5) or the approximations (7) & (8) to calculate the probability of domination. For example, if fitness values *A* to *G* have a standard deviation of $\sigma_n = 1$, the rank positions (using (7)) compared to the no noise case are shown in Table 2.

With $\sigma_n = 0$, we have conventional ranking and the probabilities will range from $2/n$ to zero. If $\sigma_n = \infty$, all of the fitness values will be assigned the same rank, and will have a probability of selection of $1/n$.

Table 2: Ranks with uncertainty of $\sigma_n = 0$ and $\sigma_n = 1$

Value	Rank ($\sigma_n = 0$)	Rank ($\sigma_n = 1$)
A	0	0.38
B	1	1.27
C	2.5	2.31
D	2.5	2.31
E	4	4.17
F	5	5.07
G	6	5.49

3.3 Multi-Objective Ranking

With multiple objectives, we now have three possible outcomes from comparing the two fitness values: *A* dominates *B*, *A* is dominated by *B*, and *A* and *B* are non-dominated. If we apply the single objective ranking equation, we find that the total of the rank positions is no longer $n(n-1)/2$ as we now have to account for the non-domination. If we have no noise, for two fitness values, where *A* dominates *B*, $P(A > B) = 1$, $P(A < B) = 0$, and $P(A \equiv B) = 0$ Therefore when we sum the probabilities of domination, the contribution from this pair will be 1. If the fitness values are non-dominated, the corresponding probabilities are $P(A > B) = 0$, $P(A < B) = 0$, and $P(A \equiv B) = 1$. We have now lost the value 1 from the probability of domination calculations, therefore reducing the sum of ranks total. This state will be the same when we compare *A* to *B* and when we compare *B* to *A*, therefore if we sum the total probability of non-domination, this will give us twice what was lost from the rank calculations.

If we consider the ranking case for a single dimension, if *A* and *B* are identical, we cannot choose between them and so add in 0.5 to the sum. With non-domination, we also have the situation where we cannot choose between objectives and should therefore add 0.5 to the sum as required. In the case of uncertain measurements, we can multiply the value of 0.5 by the probability of non-domination, and still subtract off 0.5 to allow for comparing the individual with itself, thereby maintaining the sum of the rank positions as $n(n-1)/2$. Thus we can add the non-domination term into (12). The rank calculation for multi-objective ranking is shown in (13), where n is the number of fitness measurements.

$$R_i = \sum_{j=1}^{n} P(F_j > F_i) + \frac{1}{2}\sum_{j=1}^{n} P(F_j \equiv F_i) - 0.5 \quad (13)$$

This *probabilistic ranking* equation allows chromosomes to be selected based on uncertain multi-objective fitness measurements. For the objectives shown in Fig. 3, we can calculate the rankings in order to minimise the fitness values. Table 3 shows ranks (R) for no noise, and one standard deviation noise.

In the example, we see that *A* is non-dominated with *B*, *C*, *D*, & *E* and therefore gets a rank of 2. Fitness *B* is non-

Table 3: Ranks with uncertainty of $\sigma_n = 0$ and $\sigma_n = 1$

Value	R ($\sigma_n = 0$)	R ($\sigma_n = 1$)
A	2	2.27
B	1.75	1.65
C	1.5	1.42
D	1.5	1.92
E	3.25	3.22
F	5.0	4.53

dominated with *A, C, & D* but shares an objective value with *E*, thus being half dominating and half non-dominated with *E*, the rank of *B* is 1.5 from the three non-dominated points and 0.25 from *E*, giving a total of 1.75. We also see that each of the columns of Table 3 sums to 15 $(= n(n-1)/2)$ as expected. The ranking process is $O(n^2)$, as are many of the other ranking methods [6, 7].

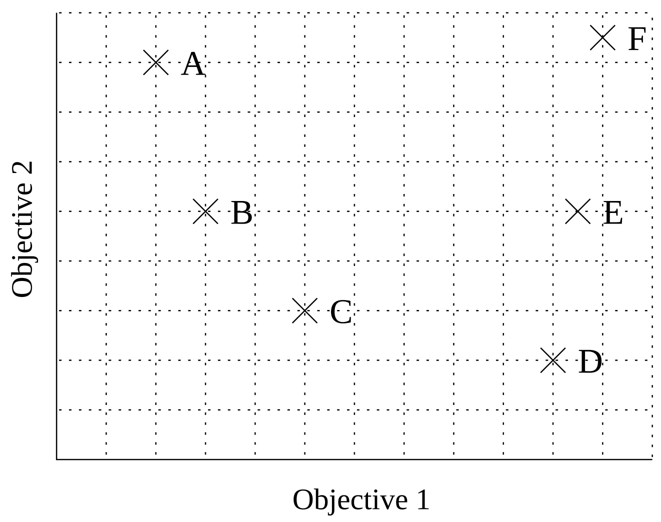

Figure 3: Multiple fitness values to be ranked

In the general noisy scenario, we see that the proximity of other fitness values, even if only close on one objective, can influence how the rank is assigned. Measurements such as *C* which are relatively well spaced out on all objectives are ranked more highly than other fitness values that are uncertain. With no noise, the basic ranking by just counting how many points dominate each fitness measurement described by Fonseca and Flemming [6] is very similar, but does not allow for the non-dominated cases. The sum of the rank values will not be consistent if non-dominated solutions are present, causing a bias towards non-dominated solutions over other solutions. The ranking used by Srinivnas and Deb [7] is based on 'layers' of non-dominated solutions and has no consistency with regards to how many layers, or ranks, are produced, therefore making calculating selection probabilities awkward.

It is interesting to note that if we require an objective to be maximised, setting σ_n negative will cause the probabilities to be calculated for maximisation. Setting σ_n negative has the same effect as negating the fitness values (the conventional way of converting from minimisation to maximisation). Therefore both minimisation and maximisation objectives may be handled easily by just setting the sign of the corresponding value of σ_n appropriately.

A comparison of the effects of noise on MOGA [6], NSGA [7], and these new ranking equations is given in [8]. These results show that NSGA performs badly in the presence of noise, whereas MOGA and the new equations are much better. By increasing the values of σ_n appropriately for the objectives, the new technique can give more stable results than both NSGA and MOGA.

4 Designer Preference and Constraints

4.1 Introduction

For optimising real systems, allowing the designer to have interactive control over the evolutionary process is paramount. The designer will often want to confine the region of evolution in order to investigate an interesting region in detail and may also want to mark some objectives as being a higher priority. A simple approach to defining regions of interest is to allow the designer to specify a limit on each objective. Solutions with a corresponding value worse than the limit would be penalised. Problem parameters are also often constrained. With a single objective it is straight forward to add a *penalty function* that adds to the objective value if the parameters are out of bounds. In a multiple objective case, as the relative scalings between the objectives is not known, applying a penalty is less straight forward. The designer may also want to focus attention on certain solutions, in the case of a noisy function, the objective values will be time dependent. For a discussion of other Pareto preference techniques see [9].

4.2 Parameter Constraints and Objective Limits

The parameters defining potential problem solutions often have regions within which they must be constrained. Deb [10] provides a discussion on the different methods used to apply constraints in EA's. If possible, the constraints should be handled by the genotypic / phenotypic description to help prevent the production of fatal solutions. To describe the constraint, we can define a function $G(\chi_i)$ as being unity if the chromosome χ_i is wholly within a constrained region (assuming no noise or uncertainty) and zero if a constraint is violated. If the individual constraints are formulated as $g_j(\chi_i) \leq 0$, we can define an error metric as $\epsilon_{ji} = g_j(\chi_i)$ and use (6) with the constraint noise standard deviation σ_c to give

$$G_j(\chi_i) \quad = \quad \frac{1}{2} + \frac{1}{2}\,\mathrm{erf}\,(\frac{-\epsilon_{ji}}{2\sigma_c}) \ . \qquad (14)$$

Multiple constraints may be combined either by forming the product (15) of the u constraints, or by taking the geometrical mean (16). It is prudent not to constrain the problem too highly in the early generations if possible to allow the evolutionary process to work with the greatest number of feasible solutions possible.

$$\begin{aligned} G &= \prod_{j=1}^{u} G_j & (15) \\ G &= \sqrt[u]{G_1 G_2 \cdots G_u} & (16) \end{aligned}$$

Limits on the objectives may be applied by treating the k limits as forming a point Z in the k dimensional objective space. The probability of each individual dominating this point can be calculated ($P(F_i > Z)$) and this probability can then be multiplied with the parameter constraint value $G(\chi_i)$ to give $C(\chi_i)$.

We can apply the constraint easily to the previously developed ranking process by applying the logic of: if both chromosomes meet all constraints, the dominance probabilities are unchanged; If both violate the constraints, they are classed as being non-dominated; if one violates constraints and the other does not, the violating chromosome is classed as being dominated.

Equation 17 shows the logic expressed in a form suitable for noisy systems.

$$\begin{aligned} P_c(A > B) &= P(A > B)C(A)C(B) + C(A)(1 - C(B)) \\ P_c(A < B) &= P(A < B)C(A)C(B) + (1 - C(A))C(B) \\ P_c(A \equiv B) &= 1 - P_c(A > B) - P_c(A < B) \qquad (17) \end{aligned}$$

The probabilities after the constraints have been applied may be used directly in the ranking calculation shown in (13).

With this method of applying the constraints, the sum of the ranks is preserved, compared to an alternative technique where the reversed rank of individual i is multiplied by $C(\chi_i)$ to give $C(\chi_x)((n-1) - R_i)$, i.e., the constraints are applied to the ranked values. Thus a solution with good objective values that violates constraints will get a low probability of selection. With this alternative approach the sum of the ranks may not be $n(n-1)/2$ anymore. With either technique, by modifying the rank position, objective scaling differences are no longer a problem. This approach allows the designer to specify limits on the evolution interactively as the population evolves. Sharing may also be applied and the niche count used to reduce $C(\chi_i)$ accordingly, i.e. $C_s(\chi_i) = C(\chi_i)/s$, where s is the niche count for the individual. This is detailed in [5].

4.3 Priority

If we have an objective with a priority of zero, it should play no part whatsoever in the ranking process. Equation 18 shows equations to allow the priorities ρ_j for each objective j to be integrated into the ranking process. The priorities ρ_j lie in the interval [0,1] with 1 being the highest priority and zero making the objective play no part in the ranking. In (18), the factor h will be zero if all of the priorities are zero. This will force all solutions to be non-dominated and so have an equal probability of selection.

$$\begin{aligned} h &= \left(1 - \prod_{j=1}^{k}(1-\rho_j)\right) \\ P_p(A > B) &= h \prod_{j=1}^{k} \left(P_c(A_j > B_j)\rho_j + (1-\rho_j)\right) \\ P_p(A < B) &= h \prod_{j=1}^{k} \left(P_c(A_j < B_j)\rho_j + (1-\rho_j)\right) \quad (18) \end{aligned}$$

As the priority calculation is performed as part of the Pareto ranking process, the consistency in the sum of the ranks is maintained. Equation 18 can be used in place of (9) in calculating the domination probabilities. This elegant integrated approach to priority and constraint gives full control to the designer.

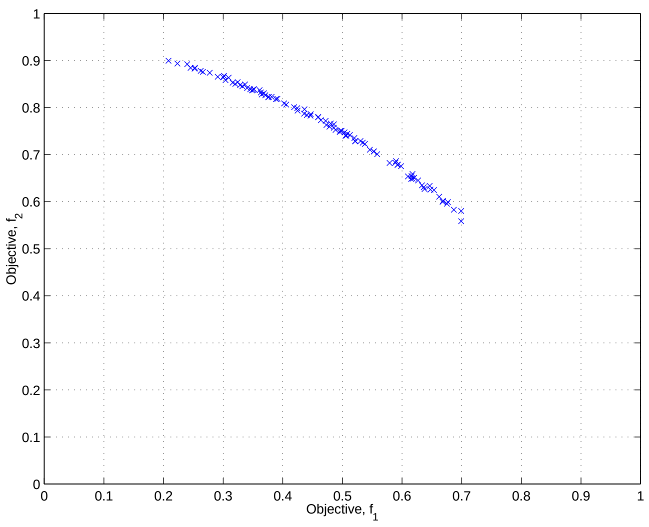

Figure 4: MOP2 objective space. No noise, Limits [0.7 0.9], No preference

5 Experimental Demonstrations

5.1 Introduction

The following graphs demonstrate the effectiveness of the ranking equations under the conditions of:

- Limits on objectives.
- Objective priority.
- Parameter constraints.

Noise and uncertainty can be split into two broad categories relating to noise that occurs within the process (Type A) and measurement noise (Type B):

- **Type A Noise:** Noise is applied to the chromosome *before* the objective function is calculated, i.e. $\mathcal{O} = F(\chi + N)$.

- **Type B Noise:** Noise is applied to the objective function *after* calculation, i.e. $\mathcal{O} = F(\chi) + N$.

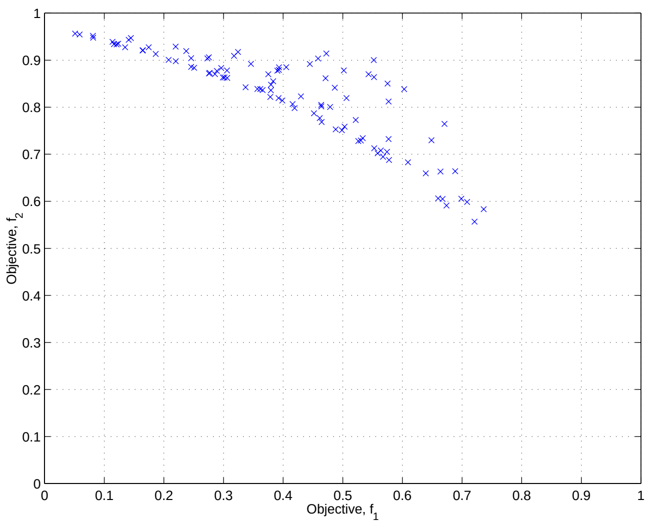

Figure 5: MOP2 objective space. Noise $\sigma = 0.1$, Type A, $\sigma_n = 0.1$, Limits [0.7 0.9], No preference

Both types of noise are of interest and often the observed noise will be a combination of type A and B. A range of test objectives were developed for the trials. Table 4 lists the objective functions used, with either type A or type B noise as appropriate.

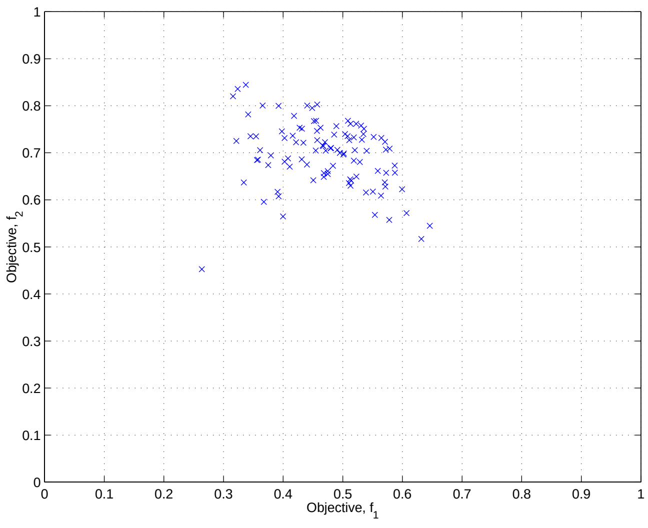

Figure 6: MOP2 objective space. Noise $\sigma = 0.1$, Type B, $\sigma_n = 0.1$, Limits [0.7 0.9], No preference

The evolutionary algorithm used was a simple structure with selection, crossover, and mutation. A population of 100 individuals was used with chromosomes consisting of two real-valued genes with values lying in the range [0,1]. Stochastic universal sampling was used to select individuals for breeding and then intermediate crossover at a rate of 70% and uniformly distributed mutation at a rate of 10% were applied to generate new individuals. The best 70% were inserted back into the population. The plots shown were all taken after 50 generations.

5.2 Objective Limits

Figure 4 shows the effect of applying the limits [0.7 0.9] to objectives f_1 and f_2 respectively with $\sigma_n = 0$ used to give a rapid transition from constrained to unconstrained. The algorithm has identified the Pareto set within the constrained region easily.

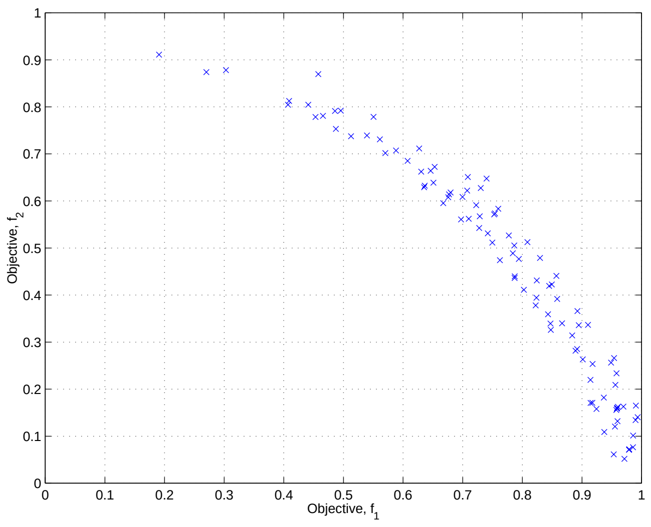

Figure 7: MOP2 objective space. No noise, No limits, Preference [0.9 1.0], Sharing 0.005 on objectives

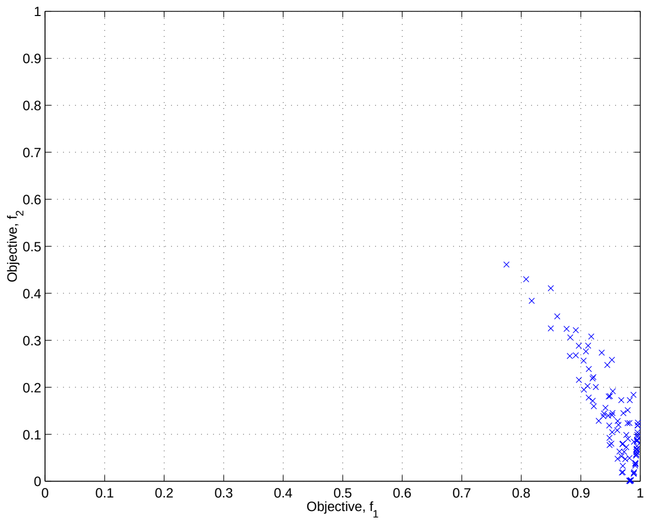

Figure 8: MOP2 objective space. No noise, No limits, Preference [0.5 1.0], Sharing 0.005 on objectives

Figure 5 shows how the ranking process copes with significant type A noise. The chromosome values have been per-

Table 4: Objective Functions

Objective	Definition	Input
MOP2 [11]	$$f_1 = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right)$$ $$f_2 = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \qquad (19)$$	$x_i = 4\,\chi(i) - 2$
CON1	$$f_1 = 1.0$$ $$f_2 = 1.0$$ $$0.01 \geq (x-0.5)^2 + (y-0.5)^2 \qquad (20)$$	$x = \chi(1)$ $y = \chi(2)$
CON2 [11]	$$f_1 = x$$ $$f_2 = y$$ $$0 \geq -(x)^2 - (y)^2 + 1 + 0.1\cos\left(16\arctan\left(\frac{x}{y}\right)\right)$$ $$0.5 \geq (x-0.5)^2 + (y-0.5)^2 \qquad (21)$$	$x = \pi\,\chi(1)$ $y = \pi\,\chi(2)$

turbed, leading to a smaller region in the chromosome space being responsible for the spread of objective values seen. Despite the noise, the search is still focussed in the required region.

Figure 6 shows the effect of type B noise. Here the objective value itself has been perturbed. Again the chromosomes occupy a smaller region, with the perturbed objective values still being focussed. By focusing the objectives, the spread of chromosomes is also reduced. With noisy objectives, if the objectives are constrained too much, there may be no single chromosome that will always have a perturbed solution within the constrained region. This can cause problems with genetic drift and loss of diversity within the population.

5.3 Objective Preference

Often with multiple objectives, not all the objectives are of equal interest to the designer. For example, in a cost / performance tradeoff, if very high volumes are to be manufactured, the cost is often paramount.

Figure 7 shows the function MOP2 without noise or objective limits but with the priorities [0.9 1.0] specified for objectives f_1 and f_2 respectively. As the objectives are minimised, preferred solutions are better on f_2 and therefore will be worse on f_1 and so will tend to lie towards the bottom right of the plot. A small amount of sharing has been applied to reduce genetic drift. This has caused the Pareto front to smear as the sharing is applied to all individuals, rather than within Pareto layers. It is clear that f_2 is dominating the results as indicated by the preference vector.

Figure 8 shows the effect of a preference vector [0.5 1.0]. Here f_1 is penalised further. If the vector [0 1.0] was used, only f_2 would have any influence on the ranking process. The priority vector allows priority information to be used within the Pareto ranking process by indicating the relative merit of each objective. In this approach to priority control, if all the elements of the priority vector are unity, a full Pareto surface can be generated.

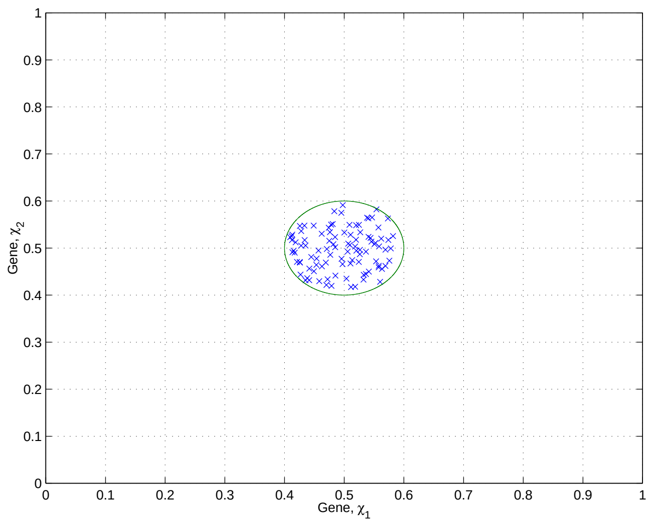

Figure 9: CONS1 chromosome space

5.4 Chromosome Constraints

Figure 9 shows the chromosome locations for the problem CONS1, along with the boundary of the constrained region. Both objective values always equal unity and therefore only the constraints have any effect. A constraint spread of $\sigma_c = 0$ was used to cause a hard transistion. It is clear that the algorithm quickly converges to the constrained region and demonstrates the effectiveness of this integrated approach over penalty methods. In problems of this type, it would be advisable to use sharing on the chromosome positions to try to reduce the effects of genetic drift.

Figure 10 shows the non-dominated boundary of the CONS2 function. Again hard constraints were generated. The discontinuous objective surface can be seen clearly. The handling of the constraints as part of the ranking process treats solutions that do not satisfy constraints as non-dominated, and so share rank positions. This reduces the rank value and effective selective pressure of the individuals, allowing the solutions that satisfy the constraints to dominate.

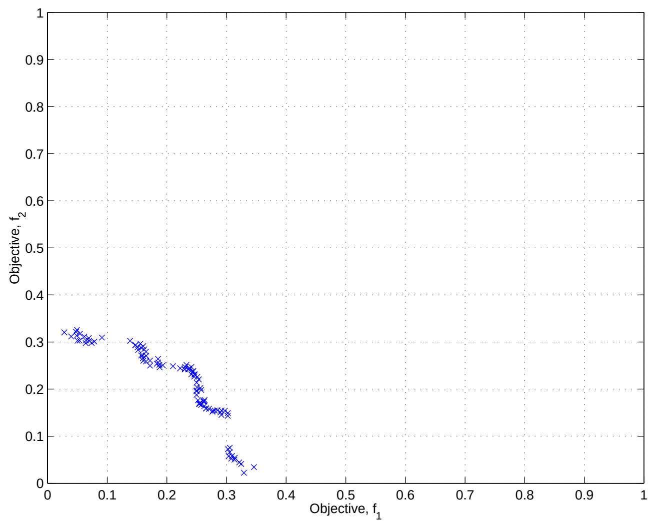

Figure 10: CONS2 Objective space

6 Conclusions

The integrated ranking, constraint, and priority equations that have been developed form a first step towards evolutionary algorithms that can address the problems of noisy objective functions directly. By integrating the constraints and priorities into the ranking, the rank values maintain their consistency and allow selection probabilities to be calculated easily.

The new ranking, constraint, and preference equations are simple functions, unlike many existing ranking processes that are based on logical decisions and are difficult to manipulate mathematically. This may help in the analysis of algorithm operation in the future. By reducing the effects of the noise on the rank positions, the evolutionary process is more stable and with the inclusion of constraints and preferences, allows the designer full interactive control over the evolutionary process.

Acknowledgements

The author would like to acknowledge the use of the Department of Aerospace, Power, and Sensors DEC Alpha Beowulf cluster for this research.

Bibliography

[1] T. W. Then and E. K. Chong, "Genetic algorithms in noisy environment," in *IEEE International Symposium on Intelligent Control*, (Columbus, Ohio), pp. 225–230, 16-18 August 1994.

[2] A. Thompson, "Evolutionary techniques for fault tolerance," in *Control '96, UKACC International Conference on*, vol. 1, pp. 693–698, IEE, 2-5 September 1996. Conf. Publ. No. 427.

[3] T. Bäck and U. Hammel, "Evolution strategies applied to perturbed objective functions," in *IEEE World Congress on Computational Intelligence*, vol. 1, pp. 40–45, IEEE, 1994.

[4] S. Tsutsui and A. Ghosh, "Genetic algorithms with a robust searching scheme," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 201–8, Sept. 1997.

[5] E. J. Hughes, "Multi-objective probabilistic selection evolutionary algorithm," Tech. Rep. DAPS/EJH/56/2000, Dept. Aerospace, Power, & Sensors, Cranfield University, RMCS, Shrivenham, UK, SN6 8LA, Sept. 2000.

[6] C. M. Fonseca and P. J. Flemming, "Multiobjective genetic algorithms made easy: Selection, sharing and mating restriction," in *GALESIA 95*, pp. 45–52, 12-14 September 1995. IEE Conf. Pub. No. 414.

[7] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1995.

[8] E. J. Hughes, "Evolutionary multi-objective ranking with uncertainty and noise," in *Evolutionary Multicriterion Optimization, EMO'01*, (Zurich, Switzerland), Springer-Verlag, 7 –9 March 2001. To Appear.

[9] D. Cvetković and I. C. Parmee, "Genetic algorithm-based multi-objective optimisation and conceptual engineering design," in *Congress on Evolutionary Computation - CEC99*, vol. 1, (Washington D.C., USA), pp. 29–36, IEEE, July 1999.

[10] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, pp. 311–338, June 2000.

[11] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Tech. Rep. TR-98-03, Air Force Institute of Technology, 1 Dec 1998.