Many Objective Optimisation: Direct Objective Boundary Identification

Evan J. Hughes

Department of Informatics and Sensors,
Cranfield University, DCMT, Shrivenham, UK. SN6 8LA
ejhughes@theiet.org

Abstract. This paper describes and demonstrates a new and highly innovative technique that identifies an approximation of the entire bounding surface of the feasible objective region directly, including deep concavities, disconnected regions and the edges of interior holes in the feasible areas. The Pareto front is a subset of the surface of the objective boundary and can be extracted easily. Importantly, if the entire objective boundary is known, breaks and discontinuities in the Pareto front may be identified using automated methods; even with high objective dimensionality. This paper describes a proof-of-principle evolutionary algorithm that implements the new and unique Direct Objective Boundary Identification (DOBI) method.

1 Introduction

The objective boundary is the 'outside hypersurface' of the hypervolume of the feasible objective region in objective space. In its full form, it encompasses both maximisation and minimisation of the objectives as demonstrated in Fig. 1a. The leading edge of the objective boundary that is heading towards a utopia point of interest is the *objective front*. Deep concavities in the objective front may be dominated, but are still viable solutions. The Pareto front is the sub-set of non-dominated solutions, given a criteria of minimisation or maximisation for each objective. If a solution is identified as belonging to the objective front and is a minimal/maximal solution as desired, but is not part of the Pareto front, then the solution must lie in a discontinuity of the Pareto front. Thus regions which are true discontinuities in the Pareto front can be identified, rather than just not knowing if solutions exist, but have not been identified by the Pareto-based optimiser.

Many real engineering problems [1] require 4 or more objectives and optimisation and visualisation of the results becomes difficult. Previous work has shown that identification of spot solutions on the objective front can yield useful information about the structure of the Pareto front [2]. Additionally, any optimisation algorithms that are intended for use in many dimensions must be capable of producing useful front approximations as the problem dimensionality increases [3–5].

Regions of the objective space that have no feasible solutions associated are also of interest. An algorithm that is attempting to identify the full objective boundary should be capable of identifying the boundary of disconnected valid objective regions and also identify the boundaries of 'holes' within feasible objective regions.

This paper describes a new and very unique algorithm that is designed to identify the objective boundary directly for many-objective problems. The approximation of the Pareto front can then be extracted from the results and analysed. The prime focus of this research has been to see if a practical algorithm for direct identification of the objective boundary could be demonstrated. The described algorithm can identify the boundaries of convex and concave regions, disjoint regions and also identify the boundaries of large 'holes' within feasible spaces. The algorithm uses a normalisation process to allow the search to be conducted easily in high-dimensional objective spaces and the paper provides a theoretical basis for key elements of algorithm tuning.

2 Objective Boundary Definition

The *objective boundary* is the set of points that form the boundary of the feasible objective region. The *objective boundary set* is the corresponding points in decision space. In objective space, points could be considered as *interior*, where they are surrounded on all sides by other feasible points, or *exterior* which have at least one direction in which they have no immediate neighbouring feasible solutions. Mathematically, the set of *exterior* solutions $\mathcal{E}$ is a subset of the entire feasible objective set $\mathcal{Q}$, $\mathcal{E} \subseteq \mathcal{Q}$, and is defined formally in (1);

$$\mathcal{E} = \{\exists \vec{n} : \vec{\mathcal{E}} + \delta t \vec{n} \notin \mathcal{Q}\} \tag{1}$$

ie. for each member of the set $\mathcal{E}$ (the objective vector denoted as $\vec{\mathcal{E}}$), there exists at least one direction $\vec{n}$, which when examined over a small distance, δt, there are no solutions in $\mathcal{Q}$ that form part of the feasible objective region. The definition is demonstrated graphically in Fig. 1a where the directions in which no immediate neighbours exist have been indicated.

3 Neighbourhood Assessment

In reality, the objective region is sampled by the optimisation process, rather than being a continuous set around each point of interest. There are many possible approaches to identifying which of the objective-space sample points are surrounded and therefore interior. Common methods for reconstructing convoluted surfaces may be considered, such as Delaunay Triangulation and level sets [6]. Delaunay Triangulation fundamentally is the basis of many of the reconstruction methods and forms a connected net between the observed valid points in the feasible region such that for any simplex in the net (triangle in 2D, tetrahedron in 3D etc.) there are no points within the circumhypersphere of the simplex. The objective surface will form a subset of the faces of the Delaunay triangulation. Unfortunately the computation of the Delaunay triangulation with high numbers of objective dimensions (≥ 10) becomes very time consuming. For computationally and theoretical simplicity, and also to provide a processing time that is linear in the number of objectives, an alternative to hypersphere and Delaunay based methods has been sought.

The concept of the new algorithm described in this paper is to assess the quality of each solution by examining the largest empty *hypercone* (or an approximation to

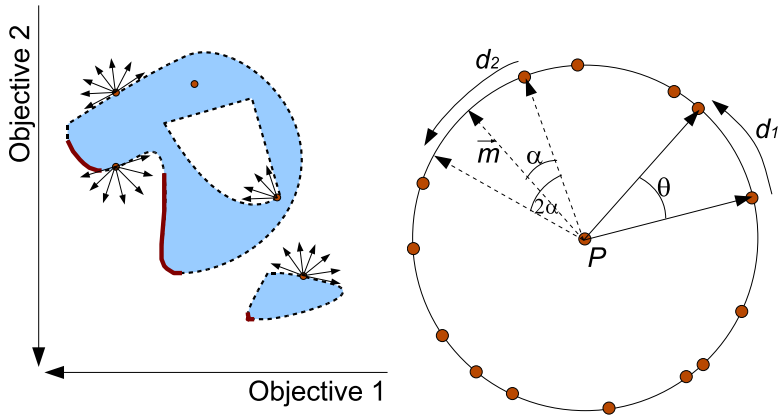

Fig. 1. Left figure shows diagram of feasible objective solutions, showing the boundary that forms the objective front (dashed), and also the Pareto front when the objectives are minimised (Solid). The objective region shown has two disjoint feasible regions, one of which has a 'hole'. The figure shows a selection of feasible solutions and the directions in which there are no close neighbouring solutions. The right figure shows a diagram of ideal conditions where neighbours are distributed uniformly and densely on the surface of a hypersphere. Arc length d_1 and nearest-neighbour angle θ will be distributed exponentially. The angle α is between a test direction, $\vec{m}$, and its nearest neighbour and α will also be distributed exponentially, but with half the mean of the distribution of θ

the largest hypercone) that can be projected from the current point of interest (cone *vertex*), out between the other valid points. The centre vector (*axis*) of the hypercone is an approximation of the vector $\vec{n}$ in (1) and the directionality of the axis can be used to assess how relevant the point under consideration is (for minimisation/ maximisation). The nearer to the objective boundary that the point of interest lies, the greater the apex angle of the largest empty hypercone becomes, i.e. a surrounded point can only fit a very narrow cone between the other points identified in the objective region.

The definition in (1) does not provide an implicit description of an optimal solution, i.e. an algorithm based on (1) would both minimise and maximise all objectives in one run of the algorithm! It is sensible in practice to restrict the direction of vector $\vec{n}$ in (1) to always point with a component in the direction of an ideal utopia solution, for example if all objectives are to be minimised, then it would be sensible to constrain all elements of the vector $\vec{n}$ to be negative, pointing to the 'lower left' corner of the optimisation hyperspace. If some objectives are to be maximised, the preferred search direction can easily be constrained accordingly.

Unfortunately finding the largest projected hypercone through a set of N-dimensional points is itself not trivial and the calculation of an exact solution will rival the processing required for the Delaunay triangulation of the set of points. Alternatively the identification of the largest cone angle could be treated as an optimisation problem.

If a random unit-length vector $\vec{m}$ is chosen and projected from a point within the objective space that is being assessed, then the closest point in angle from the remainder

of the set of objective points would form one vector which tracks the edge of an empty hypercone (a generatrix of the hypercone). If a random search is conducted of cone axis vectors $\vec{m}$, eventually a vector $\vec{m}$ would be identified which is the axis of the largest empty hypercone. Given that the probability of a random vector landing in a hypercone is highest for the hypercone of interest (i.e. by definition the largest empty hypercone has the largest solid angle), a simple random search could be employed.

In a 2D problem, if we consider a set of a large number of points distributed uniformly along a circular trajectory with a point-of-interest, P, at the cluster centre, we may analyse the behaviour of the random search theoretically. Figure 1b shows a typical configuration. If the distribution of the points are uniform along the circular trajectory, then the arc-length between neighbouring points, and therefore the cone angles between them will be distributed following an exponential distribution. Moreover, if we place a test-point at a location on the circle selected uniformly at random, the segment length and therefore the angle to the nearest neighbour solution will also be distributed exponentially [7].

The exponential behaviour for nearest neighbour distances does however require two assumptions to be satisfied:

1. The point density is sufficiently high so that the probability of observing the upper limit on the angle $\theta = 2\pi$ is very small,
2. the distribution of the points on the circle is from a uniform distribution.

The exponential distribution of the nearest neighbours is described by

$$f(x) = \lambda e^{-\lambda x} \tag{2}$$

$$F(x) = 1 - e^{-\lambda x}, \quad x \geq 0; \tag{3}$$

where $f(x)$ is the probability density function with mean $1/\lambda$ and F(x) is the cumulative distribution function that describes the probability of an observation from the distribution being less than or equal to x. For the nearest neighbour distribution the angle between neighbours will be distributed with $\lambda_\theta = N_l/2\pi$, giving a mean angle of $\bar{\theta} = 2\pi/N_l$; where N_l is the number of points on the circular trajectory ($N_l = 14$ in Fig. 1b). For the angle α between a random test vector $\vec{m}$ (distributed uniformly within the circle) and its nearest neighbour, the distribution of α will also be exponential, but with a mean that is half of $\bar{\theta}$; i.e. $\lambda_\alpha = N_l/\pi$. Although the theoretical analysis is for points lying on a circle, as a uniform distribution of points on the surface of a hypersphere may be generated by creating a vector with each axis drawn from the normal distribution N(0,1) and then normalised to unit length [8], the theory will also hold for the *angles* between a normally distributed cluster of points too, and in practice the angles between points uniformly distributed in a cartesian region [9].

In practice, a cluster of 7+ points will provide a practical lower limit to producing the anticipated exponential behaviour (assumption 1) as there is less than a 1 in 1000 chance of the exponential distribution ideally producing values greater than the maximum $\theta = 2\pi$ limit (calculated from the cumulative distribution in (3)). If a Gaussian mutation scheme is used within an evolutionary algorithm, often the localised distribution of neighbouring solutions is sufficiently uniform to satisfy assumption 2.

If we perform a random search by generating a series of test vectors, $\vec{m}$ and then taking the maximum nearest neighbour angle, then we can determine the resultant probability density distribution of the maximum observed angle by transforming (2) and (3) with the order-statistic formula [10]

$$f_{N_s,z}(x) = \frac{N_s!}{(z-1)!(N_s-z)!} F(x)^{z-1}(1-F(x))^{N_s-z} f(x); \tag{4}$$

where N_s is the number of random samples taken and z is the order statistic of interest. In this work we are interested in the maximum value and therefore $z = N_s$. Combining (2) and (3) with (4) yields (5)

$$f_{N_s}(x) = N_s \lambda \sum_{k=0}^{N_s-1} \frac{(N_s-1)!}{k!(N_s-1-k)!}(-1)^k e^{-\lambda(k+1)x} \tag{5}$$

$$F_{N_s}(x) = N_s \sum_{k=0}^{N_s-1} \frac{(N_s-1)!}{k!(N_s-1-k)!} \frac{(-1)^k}{k+1} \left(1 - e^{-\lambda(k+1)x}\right) \tag{6}$$

where $f_{N_s}(x)$ is the probability density distribution formed from taking the maximum of N_s angles and $F_{N_s}(x)$ is the cumulative probability distribution.

Thus a point can be assessed as being *interior* by:

1. Generating a random direction vector $\vec{m}$ and identifying the nearest neighbour in angle space. The nearest neighbour is found by taking the dot/inner product between $\vec{m}$ and the difference vector between the point under consideration, P and a set of N_l points $\{Q_j : j \in [1, N_l]\}$ which form a local neighbourhood around the point P. The dot product (if $\vec{m}$ and the difference vector are unit length) yields the cosine of the angle between the vectors. The smallest angle (largest cosine) is the nearest neighbour;
2. The process repeated for a different random unit length vector $\vec{m}$ and the largest overall observed neighbour angle (smallest cosine) and corresponding vector $\vec{m}$ are recorded.

The method is described mathematically as

$$\alpha = \cos^{-1}\left(\min_{i=1}^{N_s}\left(\max_{j=1}^{N_l}\left(\frac{\vec{m}_i \cdot (Q_j - P)}{|\vec{m}_i||(Q_j - P)|}\right)\right)\right). \tag{7}$$

The resultant angle α can then be considered as being associated with interior or exterior points by recognising that the cumulative probability distribution $F_{N_s}(\alpha)$ in (6) can be used to determine a threshold level for the observed angle. If the value of α needed to give $F_{N_s}(\alpha) = 0.95$ is found and used as a threshold, then there is a 95% probability that a point that is truly an interior point will be correctly classified as interior. The method can be verified experimentally and is accurate as long as the assumptions of sufficient point density and uniform distribution are observed (software that performs the experimental verification of the CDF is available at [11]).

Unfortunately the ideal case of being able to assess explicitly whether a point is exterior is not practical, as the probability density distribution of exterior points will

vary depending on the degree of convexity of the objective front local to the point and is unknown and not trivial to observe. Practically, the threshold that can be calculated for interior point acceptance is sufficient to create a useful algorithm. If a higher threshold level (e.g. 99%) is set, the algorithm is more aggressive at classifying points as interior.

4 Extension to Many-Objectives

As the dimensionality of the objective space is increased, the apex angles of the largest empty hypercones that can be fitted between a set of points also tend to increase. Thus instead of specifying the limit on the smallest apex-angle before solution cropping, an alternative strategy has been employed that allows the apex angle to be normalised into a common space, removing any issues associated with objective space dimensionality.

If a hypercone is projected through a hypersphere, the region of the hypersphere surface that is contained within the cone is a hyperspherical cap. The normalisation is obtained by transforming the observed cone angles into the ratio of the area of the hyperspherical cap to the total hypersphere area. The key observation of this transformation is that even in very high dimensional spaces, the distribution of the hyperarea ratios between nearest neighbours still follows the exponential distribution theory for nearest neighbours that was observed in the two-dimensional case. For example, in two dimensions, in Fig. 1b, if the circle is unit radius, the arc length between the two neighbours is the hyperspherical cap and d_1 may be transformed into a ratio $R = d_1/2\pi$.

Equations 8 and 9 provide the description of the hypersphere and hyperspherical cap areas [12] for a hypersphere of radius r, and (12) describes the ratio of the areas, based on the axis-to-generatrix cone angle α as shown in fig. 1b; where α would be used to calculate the ratio of the arc d_2 to the total circumference. It is clear that to calculate (12), the processing requirement is linear in the number of objectives, n.

$$A_s = \begin{cases} \frac{r^{n-1} n \pi^{n/2}}{(n/2)!} & \text{n is even,} \\ \frac{r^{n-1} n \pi^{(n-1)/2} 2^{n+1} (\frac{n+1}{2})!}{(n+1)!} & \text{n is odd,} \end{cases} \tag{8}$$

$$A_c = \begin{cases} \frac{r^{n-1} n \pi^{(n-2)/2}}{(n/2)!} p & \text{n is even,} \\ \frac{r^{n-1} n \pi^{(n-1)/2} 2^{n} (\frac{n+1}{2})!}{(n+1)!} q & \text{n is odd,} \end{cases} \tag{9}$$

$$p = \begin{cases} \alpha & n = 2, \\ \alpha - \cos(\alpha) \sum_{j=0}^{\frac{n-4}{2}} \frac{2^{2j} (j!)^2}{(2j+1)!} \sin(\alpha)^{2j+1} & n \geq 4 \end{cases} \tag{10}$$

$$q = 1 - \cos(\alpha) \sum_{j=0}^{\frac{n-3}{2}} \frac{(2j)!}{2^{2j} (j!)^2} \sin(\alpha)^{2j} \tag{11}$$

$$R = \frac{A_c}{A_s} = \begin{cases} \frac{1}{\pi} p & \text{n is even,} \\ \frac{1}{2} q & \text{n is odd,} \end{cases} \tag{12}$$

The largest angle observed by the random sampling process can be converted to a ratio R using (12) and then the ratio compared against the limit calculated from (6)

(with x representing ratio not angles) to test whether the point is interior or not. The configuration for many-objectives is the same as for the 2D case with $\lambda = N_l$ forming the shape parameter of the exponential distribution and N_s being the number of random samples used to estimate the largest cone angle (and therefore ratio). In practice, conditions of $N_s/N_l < 1/2$ and $N_s > 7$ are sufficient to ensure that the assumption that the distribution is exponential is valid. A local neighbourhood is best formed by selecting the N_l nearest neighbours using Euclidian distance in the objective space. If the local neighbourhood is small (e.g. in range 25 to 100) then the assumption that the points used for comparison follow a uniform distribution in the single dimension *ratio* space is also usually satisfied. The ratio R can also be used in the fitness assignment process to determine solution quality, with points having larger ratios being more 'interesting'.

5 Solution Spreading and Front Maintenance

As with most multi/many objective optimisation algorithms, some means of spreading the solutions evenly across the objective boundary is needed. Naturally, the solutions will tend to prefer convex regions if the ratio R alone is used as a selection criteria. For demonstration purposes with the proof-of-principle algorithm, a simple sharing mechanism has been employed. The sharing mechanism is to calculate the total weighted Euclidean distance to the N_l neighbouring solutions. The Euclidean distance to the N_l neighbours is needed in order to normalise the difference vectors for angle and ratio calculations, therefore the sharing calculation overhead is little more.

The sharing function is defined in (13), where $F_s(i)$ is the shared fitness of solution i, T is the set of N_l local solutions (including solution P), R_i is the transformed ratio for solution i, σ_s is the standard deviation of the sharing function and d_{ij} is the Euclidian distance between solutions i and j. As solution i will be compared with itself, the denominator of (13) will always be at least unity.

$$F_s(i) = \frac{R_i}{\sum_{j \in T} \exp(-\frac{d_{ij}^2}{2\sigma_s^2})} \tag{13}$$

The distance d_{ij} may be calculated in either objective or decision space, however for the proof-of-principle algorithm, only objective space sharing has been applied. For sensible sharing distances to be generated, it is advised to scale each of the objectives appropriately; in real engineering problems, this scaling information is often available.

All of the solutions that have been generated by the evolutionary algorithm so far are sorted so that the feasible solutions with the highest shared fitness are at the start of the list, and then any constrained solutions are appended after the list of sorted feasible solutions. The constrained solutions are sorted based upon their worst degree of constraint violation, i.e. each solution has a vector of constraint violations, with negative values indicating a violated constraint. The more negative the violations, the further the solution is from the constraint boundary; the minimum value of the constraint vector is used for solution sorting.

6 Evolutionary Algorithm Structure

A mutation only, evolutionary process has been employed, with an incremental structure. Each generation, the 100 top solutions are chosen from the sorted list of all solutions evaluated. These solutions are mutated with a normal distribution in each dimension. The standard deviation of the mutation is initially $\frac{1}{8}^{th}$ of the range of each decision variable, and the standard deviation is reduced by 10% each generation (i.e. multiplied by a factor of 0.9). To estimate the hypersphere ratio, a random search size of $N_s = 10$, local sample size of $N_l = 25$ and $\sigma_s = 0.001$ have been applied.

To reduce the computational effort, the hypercap ratios of most of the interior and a proportion of the crowded solutions are not evaluated each generation. Points that are considered as interior are not removed completely; they are marked as constrained but given a 50% probability of being selected for ratio re-calculation on the next iteration. If the point is still considered interior then its probability of re-calculation is reduced by a factor of 50% to 25% etc. Heavily crowded points are also marked as constrained but given a probability of 50% of being reconsidered on the next algorithm iteration.

By maintaining all solutions generated so far as possible candidates for future mutations, the algorithm does slow with increasing numbers of generations. However by not forgetting where previous solutions have been generated, the algorithm builds a good approximation to the objective front in few generations. The algorithm is best suited to initial coarse exploration of unknown problems to identify the underlying objective space structure and identify regions which warrant more focussed investigation with Many-Objective algorithms such as MSOPS-II [2].

7 Example Behaviour Results

As a demonstration of the behaviour of the algorithm, the Tanaka objective function has been used (detailed in [2]) but with an added constrained region of a small circle centred at (0.3 0.2) in the plots. Two runs, both for 50 generations have been conducted. The first run has no restriction on the direction of the boundary search and has identified the entire objective boundary, while for the second run, a preferred direction corresponding to minimisation has been imposed by restricting the components of $\vec{m}$ in (7) to be negative. Figure 2 shows the locations of the evaluated and classified solutions for the two conditions.

Figure 3 shows a comparison of the DOBI algorithm to other state-of-the-art methods. The test function has a Pareto front formed by a constraint boundary comprising the unit hypershpere (and hence is concave). The distance from the origin to identified objective vectors have been grouped to form histograms; an ideal case would have all solutions with a distance of unity. The results show clearly that the DOBI algorithm can be competitive in low dimensions and is superior for many-objective problems.

Rigourous statistical verification of the prototype algorithm behaviour is at present on-going; there are no existing metrics that allow the performance of the algorithm to be examined, and no other algorithms are capable of developing the entire objective boundary. For example the closest contender MSOPS-II can only form a limited objective front; it is not capable of identifying interior regions without a-priori knowledge of their location.

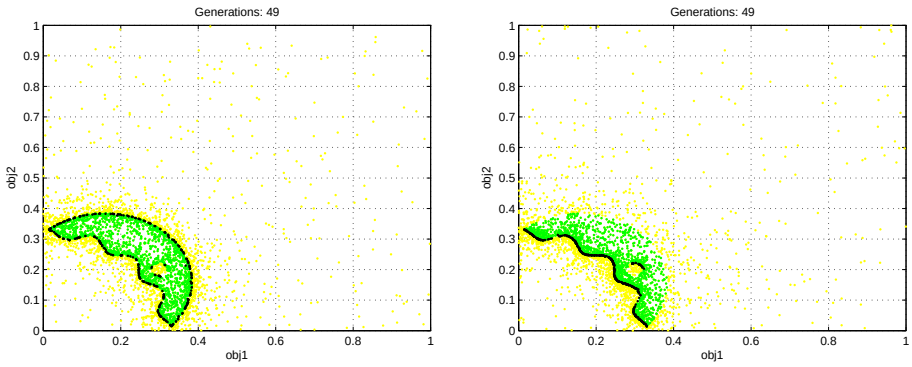

Fig. 2. Output of the prototype DOBI algorithm for the modified Tanaka function. The left plot shows the behaviour when the boundary search is unrestricted and the entire objective boundary is identified. The right figure shows the output when a restriction of minimisation is placed on the algorithm. In both plots, the convex/ concave regions of the objective front and the hole in the feasible region have been identified correctly. The light points are constrained solutions, mid grey are classed as interior and the black points are classed as exterior.

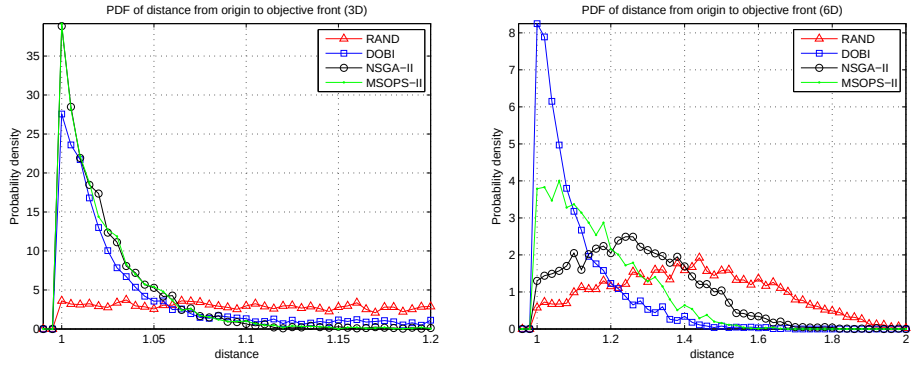

Fig. 3. Histograms of Pareto front distance for 5000 solution points from multiple runs on a unit hypersphere function of NSGA-II, MSOPS-II, DOBI (with restricted boundary search) and a random search . The left plot shows the behaviour for the 3-objective configuration; MSOPS-II and NSGA-II provide indistinguishable results with the DOBI algorithm a close 3^{rd}. The right plot shows the performance with 6 objectives. The DOBI algorithm is clearly superior, with the performance of NSGA-II falling as anticipated.

8 Conclusions

This paper has introduced a new and unique concept for an algorithm that is capable of identifying the entire boundary of the feasible objective space, even with internal holes, in many objective dimensions. The theory behind the exponential distribution of hyperspherical cap ratios has been introduced and used to construct a prototype evolutionary algorithm capable of unparalleled exploration behaviour.

By using a small random search and a normalisation into hyperspherical cap ratio space within the evolutionary algorithm, the time complexity of the search scales linearly with the number of objective dimensions, unlike Delaunay-based methods which become impractical for even moderate dimensionality. Prototype algorithm software that will re-produce all of the results in this paper is available for academic use at [11].

References

1. Hughes, E.J.: Radar waveform optimisation as a many-objective application benchmark. In: Evolutionary Multi-Criterion Optimization. Volume LNCS 4403/2007., Springer Berlin (2007) 700–714
2. Hughes, E.J.: MSOPS-II: A general-purpose many-objective optimiser. In: Congress on Evolutionary Computation CEC 2007, Singapore, IEEE (September 2007) 3944–3951
3. Purshouse, R.C.: On the Evolutionary Optimisation of Many Objectives. PhD thesis, Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK (September 2003)
4. Hughes, E.J.: Evolutionary many-objective optimisation: Many once or one many? In: IEEE Congress on Evolutionary Computation, 2005. Volume 1., IEEE (September 2005) 222–227
5. Knowles, C., Deb, eds.: Multiobjective Problem Solving from Nature: From Concepts to Applications. Natural Computing Series, Springer Berlin (2008)
6. Zhao, H., Osher, S., Fedkiw, R.: Fast surface reconstruction using the level set method. In: Proceedings of IEEE Workshop on Variational and Level Set Methods in Computer Vision (VLSM 2001). (July 2001)
7. Jiřina, M.: Nearest neighbour distance statistics estimation. Technical report, Institute of Computer Science: Academy of Sciences of the Czech Republic (October 2002) available ftp://ftp.cs.cas.cz/pub/reports/v878-02.pdf.
8. Muller, M.E.: A note on a method for generating points uniformly on n-dimensional spheres. In: Communications of the ACM. Volume 2., ACM (April 1959) 19–20
9. Hicks, J.S., Wheeling, R.F.: An efficient method for generating uniformly distributed points on the surface of an n-dimensional sphere. In: Communications of the ACM. Volume 2., ACM (April 1959) 17–19
10. Rose, C., Smith, M.D.: Mathematical Statistics with Mathematica. Springer (2002)
11. Hughes, E.J.: Software resources http://code.evanhughes.org.
12. Jacquelin, J.: Le problème de l'hyperchèvre. Quadrature (49) (July-September 2003) 6–12 ISSN 1142-2785; available http://www.maths-express.com/articles/hyperchevre.pdf.