Proceedings of IDETC/CIE 2008
ASME 2008 International Design Engineering Technical Conferences &
Computers and Information in Engineering Conference
August 3-6, 2008, Brooklyn, New York, USA

DETC2008/DAC-49684

IMPROVING THE PERFORMANCE OF VISUAL STEERING COMMANDS FOR MULTI-DIMENSIONAL TRADE SPACE EXPLORATION

Christopher D. Congdon[1]**, Daniel E. Carlsen**[2]**, and Timothy W. Simpson**[3*]
The Pennsylvania State University
University Park, PA 16802 USA

Jay D. Martin[4]
The Applied Research Laboratory
State College, PA 16804 USA

ABSTRACT

Designers perform many tasks when developing new products and systems, and making decisions may be among the most important of these tasks. The trade space exploration process advocated in this work provides a visual and intuitive approach for formulating and solving single- and multi-objective optimization problems to support design decision-making. In this paper, we introduce an advanced sampling method to improve the performance of the visual steering commands that have been developed to explore and navigate the trade space. This method combines speciation and crowding operations used within the Differential Evolution (DE) algorithm to generate new samples near the region of interest. The accuracy and diversity of the resulting samples are compared against simple Monte Carlo sampling as well as the current implementation of the visual steering commands using a suite of test problems and an engineering application. The proposed method substantially increases the efficiency and effectiveness of the sampling process while maintaining diversity within the trade space.

1 INTRODUCTION

Designers perform many tasks when developing new products and systems, and making decisions may be among the most important of these tasks, given the impact that these decisions ultimately have on the product's or system's cost, performance, time-to-market, etc. These decisions typically involve tradeoffs between competing or conflicting objectives, and many designers employ optimization-based approaches and techniques to try and help them resolve these tradeoffs.

Unfortunately, most designers do not really know their preferences when they start this process [1], or perhaps more importantly understand the implications of their preferences until they have been able to evaluate some preliminary design alternatives to form "realistic expectations of what is possible". In fact, Balling [1] has noted that the traditional optimization-based design process of "1) formulate the design problem, 2) obtain/develop analysis models, and 3) execute an optimization algorithm" often leaves designers unsatisfied with their results.

Consequently, we are investigating ways to help designers formulate and solve single- and multi-objective optimization problems in a more visual and intuitive manner. This process, which we refer to as *trade space exploration*, is an embodiment of the Design by Shopping paradigm advocated by Balling [1]: designers want to be able to "shop" for the best design, to gain intuition about trades, to see what is feasible and what is not, and to learn about their alternatives first before making a decision. Our trade space exploration process combines a multidimensional data visualization tool – the Applied Research Laboratory's Trade Space Visualizer, or ATSV [2] – along with visual steering commands [3,4] to put designers "back-in-the-loop" when performing design optimization. For example, designers can now sample new designs near any point or region of interest within the trade space by placing one or more *attractors* directly within the visualization interface.

In this paper we introduce a new attractor-based sampling strategy to bias the sampling near the point or in the region of interest in either the design space or the performance space while simultaneously maintaining diversity throughout the remainder of the trade space. The next section reviews related work, including the current implementation of our visual steering commands. In Section 3, we present an advanced sampling method to improve on the performance of attractor-based samplers. Section 4 describes a parameter robustness study of the proposed method, and Section 5 compares its performance on a set of test problems. Section 6 summarizes the key findings from this work and identifies future work.

[1] Graduate Research Assistant, Industrial & Manufacturing Engineering. Student Member ASME
[2] Graduate Research Assistant, Mechanical & Nuclear Engineering
[3] Professor of Mechanical and Industrial Engineering. Member ASME. **Corresponding Author**. 314D Leonhard Building, Phone/fax: (814)863-7136/4745. Email: tws8@psu.edu
[4] Research Associate, Product and Process Division. Member ASME

2 RELATED WORK

Interactive optimization-based methods fall mainly into the area of *computational steering* whereby users (e.g., designers) interact with a simulation model during the optimization process to help "steer" the search process toward an optimal solution, as supposed to relying on the algorithm to locate an optima based solely upon the user's initial preference. The designer observes a visual representation of the optimization process and then uses intuition, heuristics, or some other method to adjust the design space to move toward something that may not have been intuitive at the beginning of the simulation. For instance, Wright, et al. [5] applied computational steering methods to the geometric and material design of glass for a furnace. Kesavadas and Sudhir [6] created large-scale manufacturing "simulations on the fly" by allowing users to make quick changes and continue with the simulation. Messac and Chen [7] proposed an interactive visualization method based on Physical Programming [8], where the progress of the optimization is visualized - but not steered - throughout the design process, not just at the beginning and end. Likewise, Visual Design Steering and Graph Morphing [9-11] allow users to stop and redirect the optimization process to improve the solution; however, their visualization capabilities are limited to 2-D and 3-D representations of constraints and objectives.

As problem dimensionally increases, however, methods and tools are needed to help designers explore the trade space more effectively, formulate their preferences and identify the best design. Toward this end, visual steering commands [4] have been created to help designers navigate large, multi-dimensional trade spaces. We generally classify these visual steering commands as attractors, repellers, and spreaders based on their functionality: attractors/repellers bias sampling toward/away from the point or region of interest, while a spreader attempts to sample new designs uniformly over the entire region of interest. The remainder of this paper will focus specifically on improving the performance of attractors; however, we anticipate that our approach will be generalizable, enabling similar improvements for repellers and spreaders.

Sampling the output/resulting trade space with attractors can be considered as a special case of the inverse problem [12]. In the physical sciences, an inverse problem is typically defined to solve for the unobservable values that can be attributed to a set of observable values [13]. Inglese [14] solves an inverse problem that detects unobservable corrosion based upon measurements at an accessible location. In engineering design, inverse models are often created to allow designers to identify the necessary design parameters to yield a specific set of specifications. For instance, Moreau et al. [15] present a method to determine the initial temperature distribution necessary to post-creep glass dimensions to specification. Lu et al. [16] present a Backward Mapping Methodology for Design Synthesis that breaks the design space into feature-based sub-regions and then fits linear approximations between the each sub-region and each performance variable. This decomposition process allowed the methodology to be applied to non-linear and many-to-one (non-invertible) mappings. Barton et al. [12,17] present a forward-inverse metamodeling technique that attempts to minimize the number of functional evaluations required to generate an adequate representation of both the forward and inverse models. The goal of their approach is to optimize the selection of design points so that an optimal inverse metamodel can be generated, but their research focuses only on invertible problems with a single performance variable.

Finally, Stump et al. [4] introduced an attractor-based sampler controlled by an evolutionary algorithm known as Differential Evolution (DE) [18]. Their strategy utilizes the distance to the attractor as the fitness metric in DE to drive samples toward the attractor. Stated more formally, the designer creates an attractor vector φ when s/he places an attractor in the trade space. This specifies the location to bias exploration (i.e., the generation of new sample points) within a region of the trade space. The resulting objective function for the attractor is:

$$\begin{aligned}&\text{Minimize } \left|\mathbf{Y}-\boldsymbol{\varphi}\right| \\ &\text{s.t. } a_i \le \mathbf{X} \le b_i\end{aligned} \quad , \qquad (1)$$

where **X** is a *n*-dimensional design vector with bounds $[a_i,b_i]$ and **Y** is a *m*-dimensional performance vector, which combines to form the $(n+m)$-dimensional trade vector **Z**. φ is the *k*-dimensional attractor vector where $k \le (n+m)$.

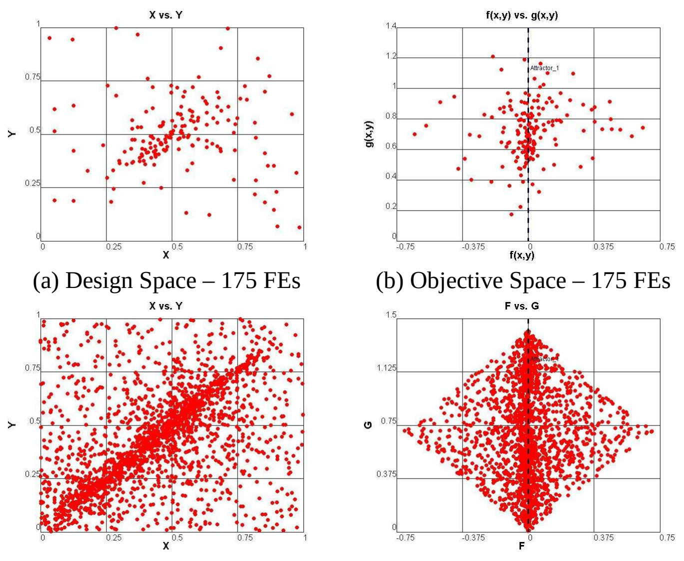
Figure 1 demonstrates the performance of this DE-based attractor for the example

$$\begin{aligned}f(x,y) &= x\cos(45°) - y\sin(45°) \\ g(x,y) &= x\sin(45°) + y\cos(45°)\end{aligned} \quad , \qquad (2)$$

where x and y vary between [0,1] with an attractor φ placed at $f(x,y) = 0$. As a result, Eq. (1) becomes:

$$\text{Minimize } |f(x,y) - 0| \qquad (3)$$

This process effectively "paints" the trade space around the attractor, allowing preference to be specified only in a limited subset of the trade space while randomly exploring the unspecified objective(s). The current implementation of the DE algorithm [4] scales in both attractor and trade vector size, but it is designed to solve single-objective problems, i.e., it will converge to the first point that reaches the attractor. Figure 1a and Figure 1b show the performance of the resulting *line attractor* in the design space and objective space, respectively, after covering to the attractor in 175 function evaluations (FEs). Figure 1c and Figure 1d show the same results after 2000 FEs, which represents 10 independent runs of the DE algorithm. These subsequent runs do not incorporate any information learned previously, which limits the effectiveness of this approach. The modifications proposed in the next section seek to overcome this limitation by employing techniques from multimodal optimization. These techniques allow the algorithm to paint the trade space around the attractor in a single run, which will improve the algorithm's efficiency and effectiveness.



(a) Design Space – 175 FEs (b) Objective Space – 175 FEs

(c) Design Space – 2000 FEs (d) Objective Space – 2000 FEs

Figure 1. DE-based Attractor Example (Attractor at f(*x*,*y*)=0)

3 PROPOSED METHOD

We introduce a new method for Advanced Sampling by Differential Evolution (ASDE) to improve the performance of the DE-based attractor. The ASDE biases new samples toward the user-specified attractor while continuing to explore dimensions for which no preference has been specified. The proposed ASDE algorithm combines the traditional DE algorithm with a speciation strategy from Li [19] and a crowding strategy from Thomsen [20], tailored for implementation within our visualization software. The remaining sections describe the key operators and discuss strategies for setting parameters for trade space exploration.

3.1 Mutation and Mating

There are many DE mutation schemes in the literature. The strategy DE/*rand*/1/*exp* is popular for multi-objective optimization problems [19,20] and is utilized for this research. We refer the reader to [21] for a detailed comparison of this strategy to other mutation strategies. This strategy works by selecting 3 random parents and recombining them in various fashions in order to generate a child. Equation (4) demonstrates this approach, with a mutated vector $\mathbf{V}_{i,\theta}$ [21]:

$$\mathbf{V}_{i,\theta} = \mathbf{X}_{r_0,\theta} + F \times \left(\mathbf{X}_{r_1,\theta} - \mathbf{X}_{r_2,\theta} \right), \tag{4}$$

where F $\in$ (0,1) is a user-defined scale factor and $\mathbf{X}_{r_i,\theta}$ is the randomly selected individual from the current species θ where $\mathbf{X}_{i,\theta}$ where $i \in \{1,\ldots,NP\}$.

After a mutated vector $\mathbf{V}_{i,\theta}$ is generated, it is mated with a parent population member $\mathbf{X}_{i,\theta}$ through a cross-over strategy. A starting point *i* is selected and the probability of cross-over is the user-defined parameter, *CR*. If cross-over is successful, then the *i*th gene of the mutated vector $\mathbf{V}_{i,\theta}$ is swapped with the *i*th gene of parent vector, $\mathbf{X}_{i,\theta}$. Then, *i* is indexed, and the process is repeated until the cross-over is unsuccessful or the end of the gene is reached, at which point the mating process stops.

After cross-over, the feasibility of the generated child must be assessed. Mezura-Montes et al. [22] and Lampinen et al. [23] present three selection criteria rules for handling selection between infeasible solutions. For this problem, we want to make sure that no sample is "wasted" due to infeasibility; therefore, a simple repair algorithm was developed to guarantee feasibility of all children. If a proposed child is infeasible, then each infeasible dimension is made feasible by multiplying the distance from $\mathbf{X}_{r_o,g}$ in Eqn. (4) to the boundary by a uniform U[0,1] random number and adding this distance to $\mathbf{X}_{r_o,g}$. This ensures feasibility and attempts to keep points from collecting on the feasible boundary.

3.2 Speciation

A technique to locate multiple optima is *speciation* used by Li in the development of his species-based DE (SDE) [19]. Speciation is a niching mechanism whereby the algorithm's population is grouped based on Euclidean distance in the design space. In DE, speciation keeps the randomly chosen members used for mating and mutation similar to those within the species. This effectively limits the algorithm's step-size to the neighborhood of each individual species, reducing selection pressure for the global population.

At the beginning of each iteration, the entire population, of size μ_g, where:

$$\mu_g = \sum_{i=1}^{NS} NP_i\,, \tag{6}$$

NS is the number of species and NP_i is the population of species *i*, is sorted based on fitness, and the fittest member is selected to be a species seed. The next individual's distance to the seed is checked, and if it falls within a user-specified radius r_θ from the seed, then it is considered to be part of that species; otherwise, it is set to be another species seed. This is repeated for all NP_i individuals in the population.

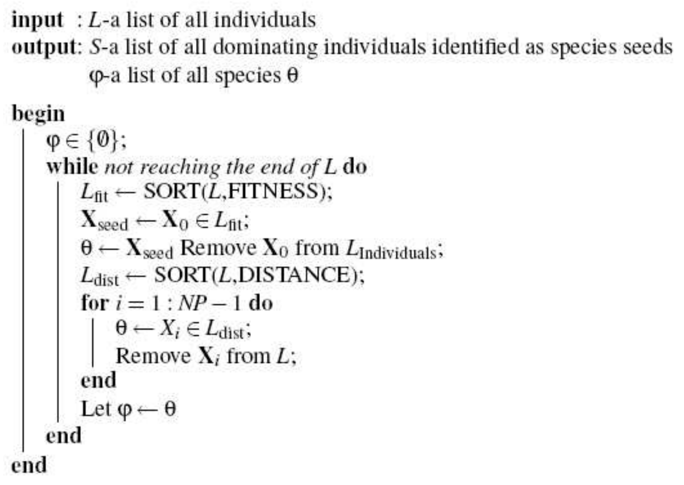
Once the species are established, a check is done to ensure that each species has at least β individuals. In DE, β is usually set to a number greater than or equal to 3 because DE requires 3 or more individuals within each species to implement its mutation and cross-over operators. If there are not enough individuals, then new individuals are randomly created within the species radius until at least β individuals exist.

Adding individuals to the species could potentially create a population larger than μ_g, but if this is the case, only μ_g total individuals are selected to be species seeds and members of a species. Another potential problem is that some species may converge more quickly than others, which may yield redundant individuals. To combat this problem, whenever an individual is created it is checked against the other individuals within the current species. If it is a copy, then it is replaced by a new random individual [19].

Specifying the radius r_θ requires problem-specific knowledge that can impact the algorithm's performance; moreover, it can be difficult to interpret from a user perspective. If r_θ is too small, then each population member will become its own seed, generating a total of β total species. Each species will then need to be increased to *NS* members to ensure that the DE strategy can perform. Li [19] suggests randomly generating these extra members, which can lead to a random-walk condition as (β-1) randomly-generated individuals compete with the species seed (worst-case). This leads to a total of (β-1)*NP* randomly-generated individuals in a given population.

The ASDE algorithm eliminates the radius, r_θ, parameter and replaces it with the species parameter, *NS*, which is more intuitive to the user, since it directly corresponds to the desired amount of diversity in the search space. By specifying *NS* and *NP*, the user is generating a total of μ_g parents in each generation, where $\mu_g = NP \times NS$. This also eliminates the need for the β parameter since all species will have *NP* members.

For ASDE, the defining member of each species is its species seed. The species seed will have the highest fitness of a given species, and the membership of each individual is determined solely on that individual's location with respect to the species seed. Figure 2 describes the layout of the ASDE algorithm for generating species.



```
input  : L-a list of all individuals
output: S-a list of all dominating individuals identified as species seeds
        φ-a list of all species θ
begin
    φ ∈ {∅};
    while not reaching the end of L do
        $L_{\text{fit}} \leftarrow$ SORT(L,FITNESS);
        $\mathbf{X}_{\text{seed}} \leftarrow \mathbf{X}_0 \in L_{\text{fit}}$;
        $\theta \leftarrow \mathbf{X}_{\text{seed}}$ Remove $\mathbf{X}_0$ from $L_{\text{Individuals}}$;
        $L_{\text{dist}} \leftarrow$ SORT(L,DISTANCE);
        for $i = 1 : NP - 1$ do
            $\theta \leftarrow X_i \in L_{\text{dist}}$;
            Remove $\mathbf{X}_i$ from L;
        end
        Let φ ← θ
    end
end
```

Figure 2. ASDE Algorithm for Generating Species

This algorithm begins with a empty set of species γ. It takes *L* as an input, which is the collection of surviving individuals from all species in the previous generation. *L* is sorted by fitness and placed in L_{fit}. The individual with highest fitness, $\mathbf{X}_o$, is removed from *L* and becomes the first species seed, $\mathbf{X}_{\text{seed}}$. L_{dist} is defined as *L*, sorted in increasing Euclidean distance to $\mathbf{X}_{\text{seed}}$. At this time, the first *NP*-1 elements of L_{dist} are inserted into species θ, while being removed from L. Species θ is added to the set of species γ, L_{fit} is resorted from the remaining members of *L*, and the process is repeated until there are *NS* total species in γ, each with *NP* individuals.

3.3 Crowding

Another multimodal optimization technique that can be used to maintain diversity in the population is *crowding* [20]. Crowding encourages exploration of the trade space, increasing the probability of converging to multiple optima. It achieves this by only allowing a newly generated offspring $\mathbf{X}_o$ to compete with the individual $\mathbf{X}_i$ that is most similar to it. Thomsen [20] defines similarity by Euclidean distance in the design space, but a performance space-based implementation can also be employed. This offspring will then replace $\mathbf{X}_i$ in the next generation if it has a better fitness. The pool of individuals that $\mathbf{X}_i$ is chosen from is a random subset (crowd) of the population with a crowd size set by the crowding factor (*CF*); usually taken to be a small number such as 2 or 3.

Due to the *CF* usually being set to a low number, crowding has been known to experience a problem called replacement error, where the offspring replaces an individual that is not similar to it [20]. ASDE combats this by setting *CF* equal to the population size, *NP*, guaranteeing that it replaces the most similar individual in the population. This increases runtime, but this is usually insignificant when compared to time needed for fitness evaluation.

3.4 Complexity

The computational complexity of the resulting ASDE algorithm is determined by the number of sorting operations and the total population size. For each generation, there is one sort by fitness (L_{fit}) and *NS* sorts by distance (L_{dist}). For each L_{dist} calculation, a total of (*NP*-1) distance calculations are required. This gives a computational complexity for speciation of $\vartheta(NS(NP\text{-}1))$, which is similar to the results found by Li [19], but with a fixed number of species, the run-time is more easily controlled by specifying the parameters *NP* and *NS*.

Meanwhile, each new individual must be compared to *NP* individuals in order to determine its closest neighbor for the crowding operator. This requires a total of $NS \times NP(NP\text{-}1)$ comparisons for each generation. The overall computational complexity of this algorithm is $\vartheta(\mu_g(NP\text{-}1))$, which is smaller than Thomsen's $\vartheta(\mu_g^2)$ for crowding [20], but is larger than Li's $\vartheta(\mu_g \times NS)$ for speciation [19] since typically $NS \leq NP$.

4 ASDE PARAMETER ROBUSTNESS STUDY

4.1 Experimental Set Up

A robustness study was conducted to understand the sensitivity of parameter selection and tuning of the proposed algorithm for ASDE. The factors (and levels) of interest are:

- *NP*: Species Population Size (10,15,20,25)
- *NS*: Number of Species (5,10)

The scale factor *F* and the cross-over ratio *CR* were set to 0.5 and 0.9, respectively, based on Price's guidelines [24] and testing by both Li [19] and Thomsen [20]. The testing range for *NP* and *NS* were selected in order to gain insight into the best makeup of a generation for a given number of functional evaluations (i.e., a generation is defined by *NP* x *NS*). Each test executes 5000 functional evaluations, which correspondingly ranges to 20 (for *NP*=25 and *NS*=10) to 100 (for *NP* =10 and *NS* = 5) total generations.

For each run, the effect of attractor location is addressed by varying the placement of the attractor at four different locations in the performance space between [0,1] (normalized by the maximum). While a *k*-dimensional attractor can consist of any combination of design and performance variables, this study will focus on performance-space attractors because design-space attractors are problem invariant, and trivial to implement. Of special interest is the algorithm's performance when the attractor is located at the boundary of performance feasibility, since all observations will be strictly greater/less than the specified attractor. In addition, three seeds are selected for random number generation for each run, with the results being averaged to account for randomness in the algorithm.

An initial set of 8 functions from the Walking Fish Group (WFG) test suite [25] were selected to be representative of multi-objective engineering design problems. The WFG suite is completely scalable in both number of allowable design and performance variables. Design variables are allocated as either position or distance variables. Position variables define a Pareto Front; their modification generates additional points about the same front. Distance variables move the Pareto Front; their modification either dominates or is dominated by the original point. By increasing the number of position parameters, the number of unique designs that map to a single performance vector increases, creating a many-to-one Pareto mapping, allowing for multiple design vectors to correspond to a single performance vector. Table 1 summarizes the features of the 8 WFG test problems used in this first study. Note that two problem sizes are considered, representing "small" (5 inputs, 2 outputs) and "medium" (15 inputs, 2 outputs) engineering design problems. Half of the problems are unimodal while the other half have multiple local optima that can trap an algorithm into false Pareto Fronts. Finally, the problems are also evenly divided between one-to-one mappings and many-to-one mappings. Executing each combination of *NP* and *NS* on each test problem yields a total of 768 runs (8 test problems x 8 combinations x 4 attractor locations x 3 random seeds).

4.2 Performance Metrics

For this parameter robustness study, the performance of each parameter combination is measured for its (1) *accuracy* – the ability to generate points near the specified attractor, and (2) *diversity* – the ability to spread points about the non-attracted variables in the design space. In a traditional multi-objective optimization algorithm, accuracy is often measured based on the distance of the generated set of non-dominated points to a known Pareto-optimal set [26,27]. Since optimization is not the intention of the ASDE algorithm, we use an average distance metric, Υ, to measure the average distance of all *n* designs to the user-specified attractor:

$$\Upsilon = \sum_{i=1}^{n} \left[\frac{|\mathbf{Y} - \boldsymbol{\varphi}|}{n} \right], \tag{7}$$

where $\mathbf{Y}$ and $\boldsymbol{\varphi}$ are the normalized [0,1] values used to ensure each dimension in the performance space is equally weighted.

The diversity metric, *H*, is based on Shannon's Entropy metric [28], which measures how evenly spaced the points are throughout the region of interest:

$$H = -\sum_{i=1}^{N} \rho_i \ln \rho_i \ , \tag{8}$$

where ρ_i represents the density of the i^{th} of *N* bins. For this experiment, the number of bins is taken as $N = n^{1/2}$.

Variations of this metric are commonly used for testing the diversity of multi-objective optimization algorithms [26,27]. For attractor-based sampling, the goal is to measure the spread of all samples about the attractor; therefore, the diversity metric *H* is calculated for each non-attracted design variable individually and then averaged for each test problem and normalized by ln(*N*), which represents the greatest possible diversity for a given number of bins.

Table 1. Features of WFG Test Problems for Robustness Study

Problem	Function	Mode	Size	Mapping
1	WFG8	Unimodal	[5,2]	One-to-One
2	WFG8	Unimodal	[5,2]	Many-to-One
3	WFG8	Unimodal	[15,2]	One-to-One
4	WFG8	Unimodal	[15,2]	Many-to-One
5	WFG9	Multimodal	[5,2]	One-to-One
6	WFG9	Multimodal	[5,2]	Many-to-One
7	WFG9	Multimodal	[15,2]	One-to-One
8	WFG9	Multimodal	[15,2]	Many-to-One

4.3 Results of ASDE Parameter Robustness Study

Assuming that the given problem is a black-box model, then the only information known *a priori* is the size of the design and performance vectors, and the intended steering command (in this case, a 1-D attractor placed in the performance space). Other information, such as the modality and mapping of the problem will seldom be known, at least not until many samples have been taken [28]. Therefore, the analysis was completed by problem size, and the effect of the tuning parameters for attractor location, modality and mapping was determined.

The results are plotted in the Appendix. The effects of *NP* and *NS* on the accuracy and diversity are plotted separately in Figure A for the "small" (*n* = 5) and "medium" (*n* = 15) WFG test problems. Regardless of problem size, *NP* = 10 and *NS* = 5 perform the best, and these settings dominate all other settings with respect to the accuracy metric. This corresponds to an exploiting algorithm setting, where a small population drives the sampling quickly to the attractor. While accuracy was worst when the attractor was near the boundaries, these settings still performed the best, relative to the other settings. For the diversity metric, *NP* = 25 and *NS* = 10 performed the best across all problem types, which corresponds to an exploring setting. This result is intuitive, since there is more diversity in the population as *NP* and *NS* increase; hence, there is more exploration of the trade space. However, the difference between average diversity of settings [*NP*,*NS*] = [25,10] and settings [*NP*,*NS*] = [10,5] is only 3%, while the difference of accuracy at these settings is 25%. Therefore, based on this analysis, [*NP*,*NS*] = [10,5] are recommended.

5 DETAILED ASSESSMENT OF ASDE

Having identified robust parameter settings for the ASDE algorithm, we can now test its performance on different problems and compare it against other sampling methods. We use the accuracy and diversity metrics from Section 4.2 to assess the performance of ASDE in each case.

5.1 Test Problems for Detailed Assessment Study

The ASDE algorithm is now tested on all 9 functions in the WFG test suite [25] with levels:

- Problem Size: [5,2],[15,2]
- Mapping: Many-to-One, One-to-One

As in the parameter robustness study, the scale factor *F* and the cross-over ratio *CR* were set to 0.5 and 0.9, respectively, and the values of *NP* and *NS* were set to 10 and 5, respectively based on the results of the robustness study. For the Many-to_One problems, all but 2 inputs are designated as position variables. For the One-to-One problems, 1 input is designated as a position variable. Each test is run for 5000 functional evaluations, and the effect of attractor location is addressed by positioning the attractor at 4 different locations between [0,1] (normalized by the maximum), and three seeds were selected for random number generation for each run, with the results being averaged. These results are compared to the baseline algorithm of simple Monte Carlo sampling over the trade space. The results are discussed and analyzed in Section 5.2.

An engineering application is also tested (see Table 2). This application is a vehicle configuration model that was developed previously to evaluate the technical feasibility of new vehicle concepts [29] and to demonstrate the visual steering commands initially proposed by Stump et al. [4]. The model consists of 11 inputs (ten continuous and one discrete) and 7 outputs (total constraint violation, mass, and five objectives). The ten continuous inputs define vehicle geometric variables and are normalized to [0,1] against the baseline model to protect the proprietary nature of the data; H defines the powertrain and can take one of six options: [1,2,3,4,5,6]. Feasible designs are those with no constraint violation (ConVio=0), and the preference for each objective is indicated in the table. In order to compare results to previous ones, we used the same attractor location: [Obj1,Obj2]=[0.847,1.127]. Results are discussed in Section 5.3.

Table 2: Vehicle Problem Definition

Model Inputs		
Variable	**Lower Bound**	**Upper Bound**
A	0	1
B	0	1
C	0	1
D	0	1
E	0	1
F	0	1
G	0	1
H	1,2,3,4,5, or 6	
I	0	1
J	0	1
K	0	1
Model Outputs		
ConVio	0 → feasible	> 0 → infeasible
Mass	Baseline = 1	Defines weight class
Obj1	Baseline = 1	Smaller is better
Obj2	Baseline = 1	Larger is better
Obj3	Baseline = 1	Larger is better
Obj4	Baseline = 1	Larger is better
Obj5	Baseline = 1	Larger is better

5.2 Performance of ASDE on WFG Test Problems

Figure 3 summarizes the percent change of ASDE for the WFG test problems as compared to simple Monte Carlo sampling. As expected, ASDE significantly outperforms MC sampling with respect to the accuracy metric, but at the cost of diversity. However, the median loss of diversity is only 9% while the median gain in accuracy is 65%. The One-to-One problems had a higher accuracy improvement, with a mean gain of 72% vs 60% for Many-to-One problems. ASDE's performance is relatively invariant to mapping, but the MC sampling performs about 16% worse with the Many-to-One problems. This can be attributed to a significant bias caused by the Many-to-One mapping which decreased the probability of randomly sampling near the placed attractors. There was no significant difference in diversity with respect to problem mapping for either ASDE or simple MC sampling.

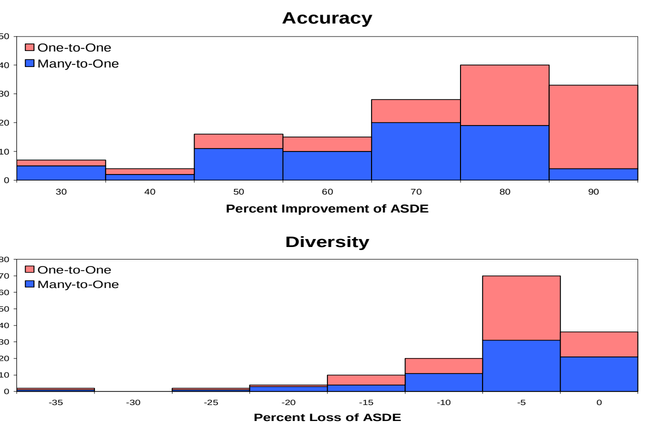
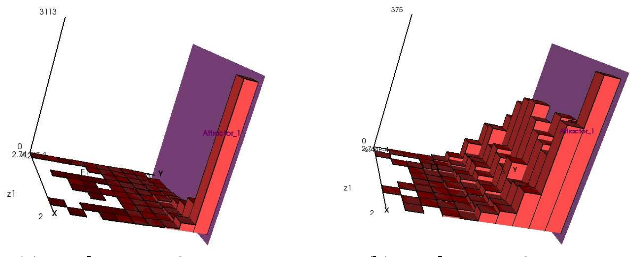


Figure 3. Percent Difference in Accuracy (top) and Diversity (bottom) of ASDE compared to Monte Carlo Sampling for All WFG Test Problems

Figure 4 demonstrates the effect of tuning-parameter selection on ASDE's diversity. It shows a 2-D histogram of design variable **Z1** vs. performance variable **F1**. The test function that performed the worst, where the diversity loss was nearly 40%, was WFG1 when the attractor was placed at [φ: **F1**=2.75] which was the upper bound of feasibility for **F1** (see Figure 4a). This is a circumstance where increasing exploration through tuning would be beneficial. For example, if this test was run with tuning parameters [*NP*,*NS*] = [25,10], then the diversity is improved as seen in Figure 4b. Under this circumstance, by increasing ASDE's exploration capability through tuning, the gain in accuracy (as compared to MC sampling) decreases from 70% to 36%, but the reduction in diversity is only 4% (vs. 7%) of the MC.

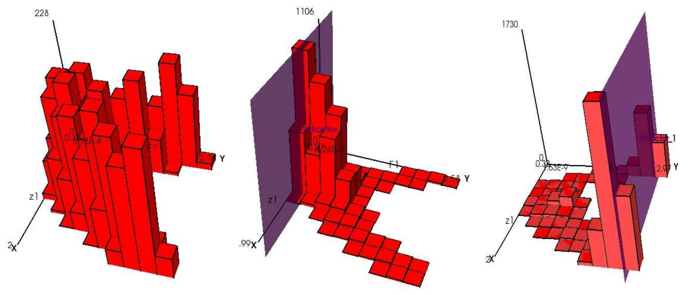


(a) Exploitation Strategy (b) Exploration Strategy

Figure 4. Effect of Tuning Parameter Selection on WFG1 with an Attractor Placed Outside the Feasible Region [φ: F1=3.0**]**

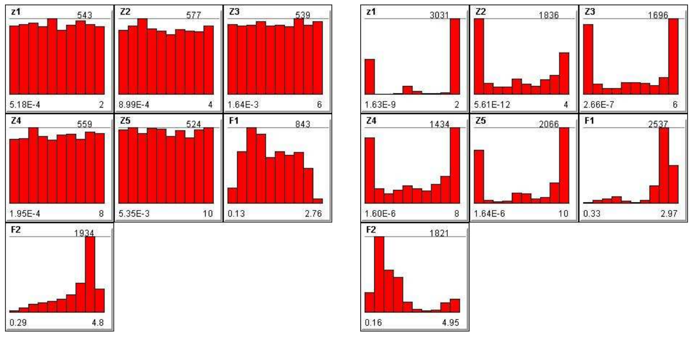
Figure 5 visually demonstrates the ability of ASDE to bias sampling to various attractor locations for WFG4 with [*n*=5] and a many-to-one mapping. Note that the attractor location is indicated by the vertical plane shown in each plot, and the height of each column indicates the number of samples within this particular region of the two-dimensional histogram. The axes represent design variable **Z1** and performance variable **F1**. While Monte Carlo sampling is effective at spreading points across **F1** (see Figure 5a), ASDE can bias sampling toward specific regions of interest. For instance, Figure 5e shows that ASDE can effectively sample two regions simultaneously by adjusting the sampling of each design variable in **X**.

A comparison of the final distribution of each design variable in **X** is shown in Figure 6. The speed of ASDE in locating an attractor can be seen in a time-series of 2-D histograms in Figure 7. Figure 7a shows the initial exploration of ASDE, as the search space is sampled broadly in order to locate points near the attractor. By 1000 FEs, the algorithm has already begun biasing samples towards the attractor. There is no discernable change in the sampling from the 1000 to 2000 FEs interval (see Figure 7b) to the 4000 to 5000 FEs interval, (see Figure 7c). By 2000 FEs, ASDE has exploited the region around the attractor, and relatively few samples are generated away from it. However, there is a cluster of points generated near [Z1=1.0], which maintains active sampling throughout the entire time-series, even though it is not near the attractor of interest. This demonstrates the balance of exploration and exploitation, as a species is located in that region every generation in an effort to find new and potentially better solutions. As *NS* is decreased, the effect of these "rouge" species decreases, at the risk of completely missing alternate regions of interest along the attractor.



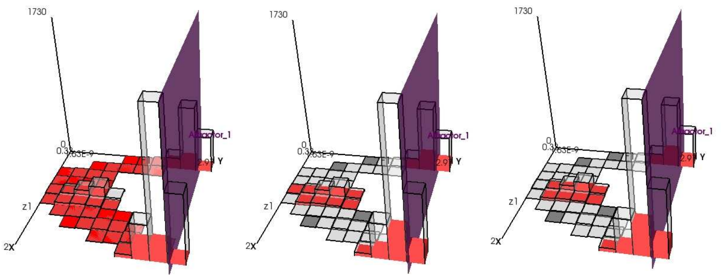
(a) Simple Monte Carlo Sampling (b) ASDE - Attractor: [φ: F1=0.25] (c) ASDE - Attractor: [φ: F1=2.75]

Figure 5. Examples of Attractor Placement and Resulting Sample Distribution for WFG4 (*n*=5, many-to-one mapping)



(a) Simple Monte Carlo Sampling (b) ASDE - Attractor: [φ: F1=2.75]

Figure 6. Histogram of Sample Distributions for All Dimensions for WFG4 (n=5, many-to-one mapping)

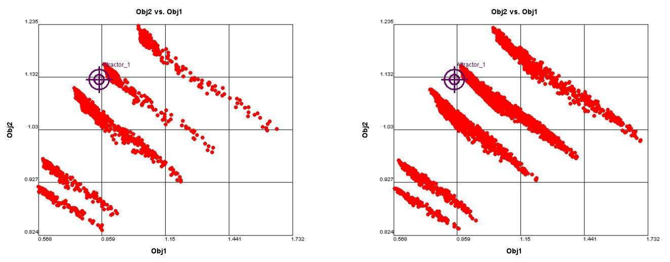


(a) 0 to 1000 FEs (b) 1000 to 2000 FEs (c) 4000 to 5000 FEs

Figure 7. Evolution of Sample Points by Number of Function Evaluations (FEs) for WFG4 (*n*=5, many-to-one mapping) and Attractor at [φ:F1=2.75].

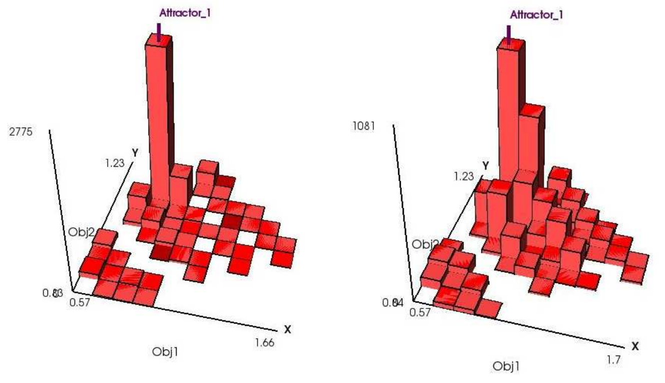
5.3 ASDE on Engineering Application

Five trials are run using the original DE-based point sampler [4], and five are run using ASDE for the vehicle configuration model. Each trial is allowed to run for 5000 function evaluations by setting the Population Limit for ASDE to 5000 and looping the DE-based point sampler until it uses 5000 function evaluations. Each of the five ASDE trials and each DE-based point sampler trial are very similar to one another, and Figure 8 and Figure 9 are representative of the results from these five runs. Using the default settings for the DE-based point sampler and [*NP*, *NS*] = [10,5] for ASDE yields more diversity in DE-based point sampler results but the ASDE results are more accurate, i.e., closer on average to the user-specified attractor.



(a) ASDE Sampler - Run 1 (b) DE-based Point Sampler - Run 2

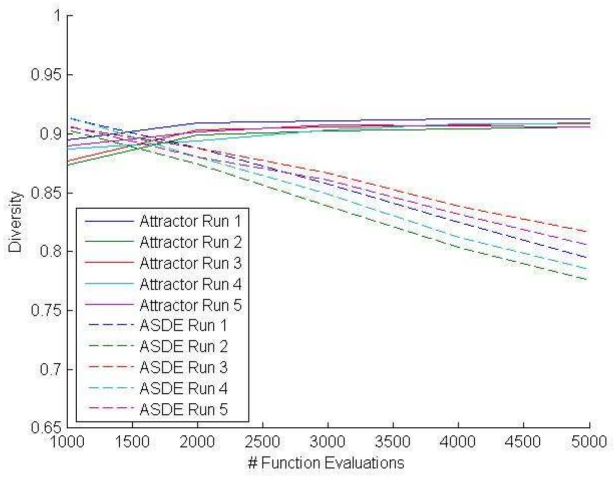
Figure 8. Representative Scatter Plots of ASDE Sampler and DE-based Point Sampler



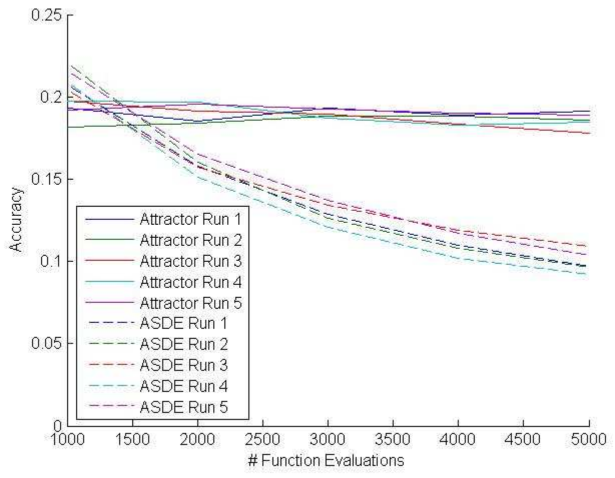
(a) ASDE Sampler - Run 1 (b) DE-based Point Sampler - Run 2

Figure 9. Corresponding 2D Histograms of ASDE Sampler and DE-based Point Sampler

Figure 10 shows how the accuracy and diversity metrics evolve with the number of function evaluations for each trial. For all of the trials, both metrics start off close to one another for the two samplers; however, the metrics for the DE-based point sampler remain nearly constant during all 5000 function evaluations. Meanwhile, the ASDE sampler becomes more accurate as the number of function evaluations increase but at the sacrifice of diversity. This is likely happening because each time the DE-based point sampler converges to the attractor it restarts with no memory of its previous search. The ASDE sampler, on the other hand, takes longer to find the attractor, but once it does, it continues to find points near the attractor. As discussed in Section 5.2, the performance of ASDE can be modified by adjusting [*NP*, *NS*] to meet the user's needs, tailoring the algorithm to the specific problem as needed.



(a) Diversity vs. Functional Evaluations



(b) Accuracy vs. Functional Evaluations

Figure 10. Evolution of Diversity and Accuracy of ASDE and DE-based Point Sampler for Each Run

6 CLOSING REMARKS AND FUTURE WORK

The proposed ASDE algorithm extends DE through modified speciation and crowding to explore a trade space more efficiently and effectively. This is in contrast to the traditional DE strategy, which seeks to converge to a single optimum (single objective) or Pareto set (multi-objective). In doing so, it provides useful information to designers, allowing them to form their preferences as they learn about what is realistically feasible versus forcing them to state their preferences *a priori*.

A parameter robustness study was performed to determine the effect of parameter selection on the performance of the ASDE algorithm. The study found that under most conditions a single parameter setting provides the best accuracy with minimal loss in diversity. A detailed study compared the performance of ASDE to simple Monte Carlo sampling using the WFG test suite [25] and found that ASDE was effective in accurately locating the attractor with minimal diversity loss.

Finally, ASDE was compared to a DE-based point sampler [4] on an existing vehicle configuration model. ASDE was able to outperform the DE-based point sampler in terms of accuracy by continuing to explore regions near the attractor without repeatedly restarting the algorithm after convergence. This restarting gave the DE-based point sampler an advantage in terms of diversity, resulting from the randomness with which new populations are created each time the algorithm restarts.

ASDE does not currently employ a species convergence strategy to ensure that new and unique points are continually generated. For future work, a strategy to monitor and address species pre-convergence is needed. In addition, methods to modify *NP* and *NS* "on the fly" should be investigated to allow designers to maintain and fine-tune the extent to which ASDE explores the trade space versus exploits new knowledge as it is gained. Additional studies using design-space attractors, combined design/performance space attractors, and "large" test problems with high dimensionality (e.g., 50+ inputs) and more than two objectives should be evaluated to further assess the robustness of the ASDE algorithm. Finally, research into seeding ASDE with promising designs selected by the user will provide the user with more direct control of the search process and could improve the algorithm's efficiency and effectiveness.

ACKNOWLEDGEMENTS

This work has been supported by the National Science Foundation under Grant No. CMMI-0620948. Any opinions, findings, and conclusions or recommendations presented in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Balling, R., 1999. "Design by Shopping: A New Paradigm?," *Proceedings of the Third World Congress of Structural and Multidisciplinary Optimization*, Buffalo, NY, University at Buffalo, Vol. 1, pp. 295-297.

[2] Stump, G. M., Yukish, M., Simpson, T. W., and Harris, E. N., 2003. "Design Space Visualization and Its Application to a Design by Shopping Paradigm," *ASME Design Engineering Technical Conferences*, Chicago, IL, ASME, Paper No. DETC2003/DAC-48785.

[3] Yukish, M., Stump, G. M. and Lego, S., 2007. "Visual Steering and Trade Space Exploration," *IEEE Aerospace Conference*, Big Sky, MT, IEEE, IEEE-2007-1188.

[4] Stump, G. M., Lego, S., Yukish, M. A., Simpson, T. W., and Donndelinger, J. A., 2007. "Visual Steering Commands for Trade Space Exploration: User-Guided Sampling with Example," *ASME Design Engineering Technical Conferences*, Las Vegas, NV, ASME, Paper No. DETC2007/DAC-34684.

[5] Wright, H., Brodlie, K., and David, T., 2000. "Navigating High-Dimensional Spaces to Support Design Steering," *Proceedings of IEEE Visualization 2000*, IEEE Computer Society Press, pp. 291–296.

[6] Kesavadas, T., and Sudhir, A., 2000. "Computational Steering in Simulation of Manufacturing Systems," *IEEE International Conference on Robotics and Automation*, IEEE, Vol. 3, pp. 2654–2658.

[7] Messac, A., and Chen, X., 2000. "Visualizing the Optimization Process in Real-Time Using Physical Programming," *Engineering Optimization*, **32**(6), pp. 721–747.

[8] Messac, A., 1996. "Physical Programming- Effective Optimization for Computational Design," *AIAA Journal*, **34**(1), pp. 149–158.

[9] Winer, E. H., and Bloebaum, C. L., 2001. "Visual Design Steering for Optimization Solution Improvement," *Structural and Multidisciplinary Optimization*, **22**(3), pp. 219–229.

[10] Winer, E. H., and Bloebaum, C. L., 2002. "Development of Visual Design Steering as an Aid in Large-Scale Multidisciplinary Design Optimization. Part I: Method Development," *Structural and Multidisciplinary Optimization*, **23**(6), pp. 412–424.

[11] Winer, E. H., and Bloebaum, C. L., 2002. "Development of Visual Design Steering as an Aid in Large-Scale Multidisciplinary Design Optimization. Part II: Method Validation," *Structural and Multidisciplinary Optimization*, **23**(6), pp. 425–435.

[12] Barton, R. R., 2005. "Issues in Development of Simultaneous Forward-Inverse Metamodels," *37th Conference on Winter Simulation*, M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds., Orlando, Fl: IEEE, pp. 209–217.

[13] Tarantola, A., 2005, *Inverse Problem Theory and Methods for Model Parameter Estimation*, SIAM, Philadelphia, PA.

[14] Inglese, G., 1997. "An Inverse Problem in Corrosion Detection," *Inverse Problems*, **13**(4), pp. 977–994.

[15] Moreau, P., Lochegnies, D., and Oudin, J., 1998. "An Inverse Method for Prediction of the Required Temperature Distribution In The Creep Forming Process," *Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, **212**(1), pp. 7–11.

[16] Lu, S., Bukkapatnam, S., Ge, P., and Wang, N., 1999. "Backward Mapping Methodology For Design Synthesis," *ASME Design Engineering Technical Conferences*, Las Vegas, NV, ASME, Paper No. DETC1999/DTM-8766.

[17] Barton, R. R., Meckesheimer, M., and Simpson, T. W., 2001. "Experimental Design Issues for Simultaneous Fitting of Forward and Inverse Metamodels," *4th St. Petersburg Workshop on Simulation*, St. Petersburg, Russia, pp. 69–76.

[18] Storn, R., and Price, K., 1995. *Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Space*, Tech. Rep. TR-95-012, ICSI, Berkeley, CA.

[19] Li, X., 2005. "Efficient Differential Evoluation Using Speciation for Multimodal Functional Optimization," *2005 Conference on Genetic and Evolutionary Computation*, Washington, DC, ACM, pp 873–880.

[20] Thomsen, R., 2004. "Multimodal Optimization Using Crowding-Based Differential Evolution," *Proceedings of the 2004 Congress on Evolutionary Computation*, IEEE, Vol. 2, pp. 1382–1389.

[21] Price, K., Storn, R., and Lampinen, J., 2005, *Differential Evolution. A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag Berlin Heidelberg, New York.

[22] Mezura-Montes, E., Coello, C., and Tun-Morales, E., 2004. "Simple Feasibility Rules and Differential Evolution for Constrained Optimization," *Proceedings of the Third Mexican International Conference on Artificial Intelligence*, G Arroyo-Figueroa, L.E. Sucar, and H. Sossa, eds, Mexico City, Mexico, Springer, pp. 707–716.

[23] Lampinen, J., 2002. "A Constraint Handling Approach for The Differential Evolution Algorithm," *Congress on Evolutionary Computation 2002*, Honolulu, HI, IEEE, Vol. 2, pp. 1468–1473.

[24] Price, K. V., 1999. "An Introduction to Differential Evolution," In *New Ideas in Optimization*. McGraw-Hill Ltd., UK, pp. 79–108.

[25] Huband, S., Hingston, P., Barone, L., and While, L., 2006. "A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit," *IEEE Transactions on Evolutionary Computation*, **10**(5), pp. 477–506.

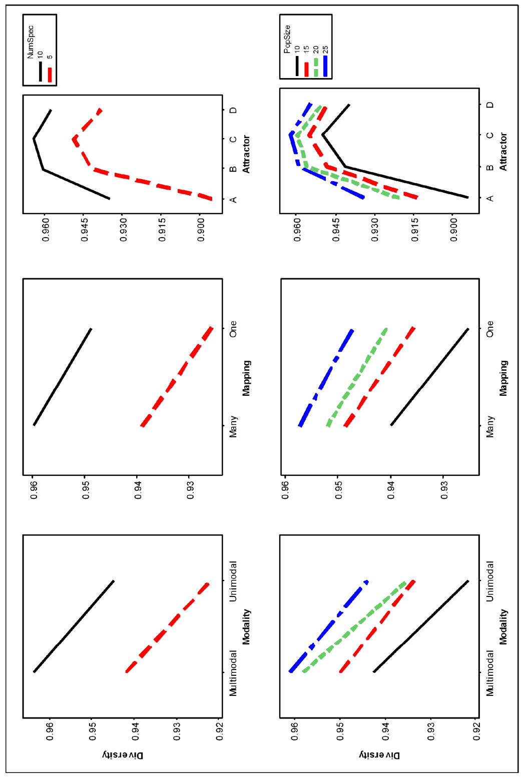
[26] Deb, K., and S., J., 2004. *Running Performance Metrics for Evolutionary Multi-Objective Optimization*, Tech. Rep. 2002004, Indian Institute of Technology, Kanpur, India.

[27] Farhang-Mehr, A., and Azarm, S., 2002. "Diversity Assessment of Pareto Optimal Solution Sets: An Entropy Approach," *Congress on Evolutionary Computation*, Honolulu, HI, IEEE, Vol. 1, pp. 723–728.

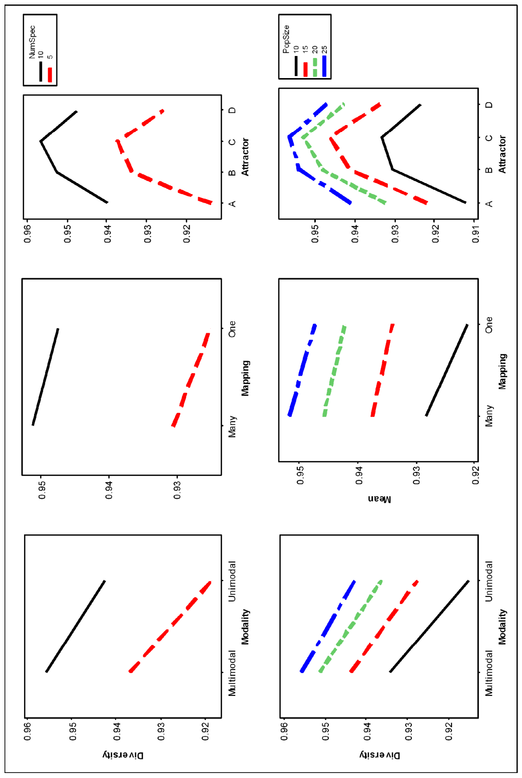
[28] Shannon, C. E., 1948. "A Mathematical Theory of Communication," *The Bell Systems Technical Journal*, **27**, pp. 379–423, 623–656.

[29] Donndelinger, J., Ferguson, S., and Lewis, K., 2006. "Exploring Mass Trade-Offs In Preliminary Vehicle Design Using Pareto Sets," *11th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA. AIAA-2006-7056.

Diversity Metric

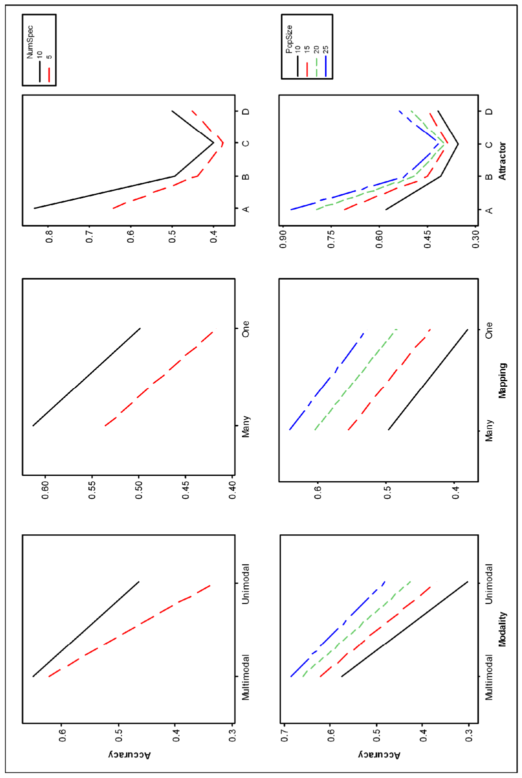


(a) “Small” problems (n=5)

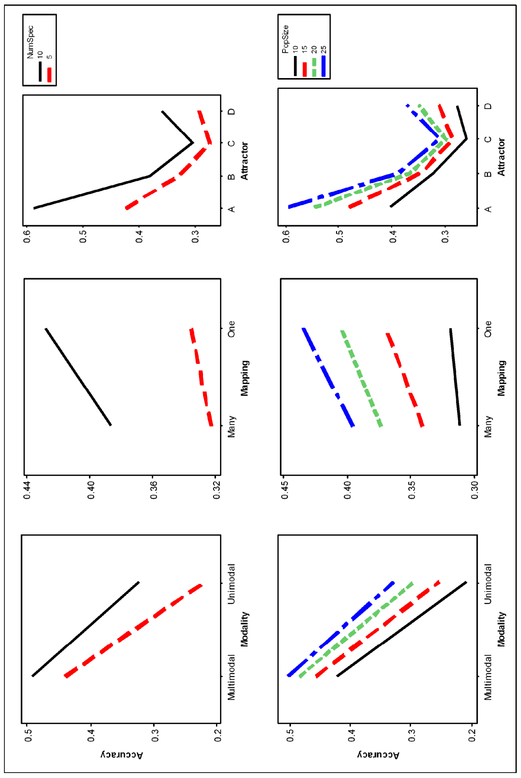


(b) “Medium” problems (n=15)

Accuracy Metric



(a) “Small” problems (n=5)



(a) “Medium” problems (n=15)

Figure A. Impact of ASDE Tuning Parameters (NS in top row, NP in bottom)