AIAA 2001-1550

A Bell-Curve Based Algorithm for Mixed Continuous and Discrete Structural Optimization

Rex K. Kincaid
College of William and Mary,
Williamsburg, VA 23187-8795

Michael Weber
21351 Ridgetop Circle, Suite 400, Dulles, VA 20166

Jaroslaw Sobieszczanski-Sobieski
NASA Langley Research Center, Hampton, VA 23681,
AIAA member

42st AIAA Structures,
Structural Dynamics and Materials
April 16–20, 2001/Seattle, WA

A Bell-Curve Based Algorithm for Mixed Continuous and Discrete Structural Optimization

Rex K. Kincaid*
College of William and Mary, Williamsburg, VA 23187-8795
Michael Weber†
21351 Ridgetop Circle, Suite 400, Dulles, VA 20166
Jaroslaw Sobieszczanski-Sobieski‡
NASA Langley Research Center, Hampton, VA 23681, AIAA member

An evolutionary based strategy utilizing two normal distributions to generate children is developed to solve mixed integer nonlinear programming problems. This Bell-Curve Based (BCB) evolutionary algorithm is similar in spirit to $(\mu + \mu)$ evolutionary strategies and evolutionary programs but with fewer parameters to adjust and no mechanism for self adaptation. First, a new version of BCB to solve purely discrete optimization problems is described and its performance tested against a tabu search code for an actuator placement problem. Next, the performance of a combined version of discrete and continuous BCB is tested on 2-dimensional shape problems and on a minimum weight hub design problem. In the latter case the discrete portion is the choice of the underlying beam shape (I, triangular, circular, rectangular, or U).

1. Introduction.

Evolutionary methods are exceedingly popular with practitioners of many fields; more so than perhaps any optimization tool in existence. Historically Genetic Algorithms (GAs) led the way in practitioner popularity (Reeves 1997). However, in the last ten years Evolutionary Strategies (ESs) and Evolutionary Programs (EPs) have gained a significant foothold (Glover 1998). One partial explanation for this shift is the interest in using GAs to solve continuous optimization problems. The typical GA relies upon a cumbersome binary representation of the design variables. An ES or EP, however, works directly with the real-valued design variables. For detailed references on evolutionary methods in general and ES or EP in specific see Back (1996) and Dasgupta and Michalewicz (1997). We call our evolutionary algorithm BCB (bell curve based) since it is based upon two normal distributions.

BCB for continuous optimization, first presented in Sobieszczanski-Sobieski et al. (1998), is similar in spirit to ESs and EPs but has fewer parameters to adjust. A new generation in BCB is selected exactly the same as a $(\mu + \lambda)$-ES with $\lambda = \mu$. That is, the best μ individuals out of μ parents plus λ children are selected for the next generation. Thus, fit individuals may continue from one generation to the next. The recombination and mutation mechanisms are illustrated in Figure 1. Consider the line through two n-dimensional parent vectors $\vec{P}_1$ and $\vec{P}_2$ selected for mating. First, determine the weighted mean $\vec{M}$ of these two vectors where the weights are given by the fitness of each parent. Next, sample from a normal distribution $N(0,\sigma_m)$ to establish point $\vec{B} = \vec{M} + |\vec{P}_2 - \vec{P}_1| * N(0, \sigma_m)$. Note that $\vec{B}$ must lie on the line through P_1 and P_2 but may lie outside the line segment $\overline{P_1P_2}$. Next, we generate a radius r for an $n-1$ dimensional hypersphere centered on $\vec{B}$. The radius is a realization from a $N(0, \sigma_r)$. Typically $(\sigma_r >> \sigma_m)$. Finally the child $\vec{C}$ is selected by sampling uniformly on the surface of the $n-1$ dimensional hypersphere. Hence, there are two parameters σ_r and σ_m in addition to the traditional parameters of population size and number of generations. The tails of the two normal distributions, $N(0, \sigma_m)$ and $N(0, \sigma_r)$, provide opportunity for additional, low probability mutations.

Sobieszczanski-Sobieski et al. (1998) presented two mating schemes that were tested for a continuous hub design problem. Mating scheme 1 chose two parents from a roulette wheel in which the sector sizes were determined by a fitness value equal to the sum of the

*Department of Mathematics, The first and second authors gratefully acknowledge the support of of NASA-Langley Research Center–NAG-1-2077

†Cigital, Inc.

‡Computational AeroSciences and Multidisciplinary Research Coordinator

weight of the structure (the objective value) and the maximum constraint violation. Mating scheme 2 selected one parent from a roulette wheel based on objective values and a second parent from a roulette wheel based on maximum constraint violation. Method 2 was shown to be superior and we use this mating scheme in our BCB experiments in this manuscript. The quality of solutions generated in Sobieszczanski-Sobieski et al. (1998) were verified by comparing BCB solutions to ones generated by CONMIN (Vanderplaats 1973), a standard nonlinear programming package. No attempt was made in Sobieszczanski-Sobieski et al. (1998) to analyze the sensitivity of the BCB parameters. Kincaid et al. (2000a) provide an investigation into BCB parameter selection. Two results are of particular import to our work here. They found that it is crucial to scale the range of values for the decision variables so that the scales are roughly equivalent. They also found that, in general, $\sigma_r >> \sigma_m$ is best and that $\sigma_r = 4.0$ and $\sigma_m = 1.0$ are reasonable default values. Kincaid et al. (2000b) compares the performance of BCB against two competing evolutionary programs. In addition, a hybrid scheme is examined in which BCB's role is to identify high quality basins of attraction before an efficient local search takes over. Two local search schemes were tested—a pattern search and a quasi-newton search. Although the standard BCB performed well the hybrid approach proved superior.

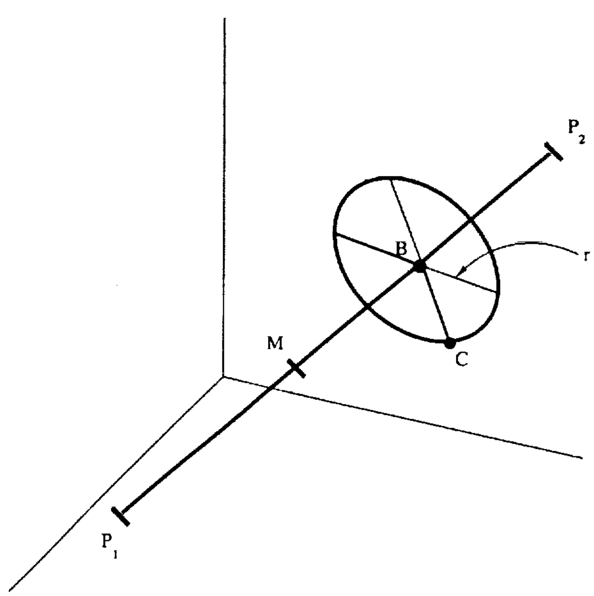

Figure 1. BCB Geometrical Construct in 3D Space

2. BCB for discrete optimization problems

BCB was designed to find high quality solutions to continuous optimization problems. In order to solve mixed continuous and discrete variable optimization BCB must also be able to optimize purely discrete problems as well. Our first experiment in this regard was to simply round continuous BCB solutions to the nearest discrete value (see Kincaid et al. 2000) for a 2-member hub design problem. This approach is known to work fairly well for quasi-discrete variables, that is, the variables have a continuous physical interpretation but may be implemented only by choosing from a discrete set of options. The thickness of commercially available sheet metal is an example. In contrast, the variables may be truly discrete. Choosing between propellers or jet engines in an aircraft design optimization problem is an example. The rounding approach is totally inapplicable for truly discrete variables. Consequently, in the truly discrete setting there is no obvious analog to a line segment connecting parents. Defining an underlying geometry via a lattice is possible but our approach is to define an artificial neighborhood structure.

The notion of a neighborhood we use rests on a definition of distance between objects in a finite set. This idea is not new. In particular, Kelly et al. (1994) use a similar measure in their diversification strategies in tabu search. To the best of our knowledge there is no reference to this in the evolutionary strategy literature. To be generic, consider an example of three objects, each possessing three attributes. Figure 2b provides such an example in which the attributes are letters chosen from a list of six letters (Figure 2a). Initially we assume that the order of the attributes in the object does not matter. We will remove this assumption later. Now, consider object β and ask the question: "How many letters must be changed to make β identical to α?". By inspection, the answer is three. The answer to the same question for γ is two. Finally to change γ to β we need change only one letter. Obviously, the above questions may be reversed without changing the answers. That is, the transformation of α to γ also requires two letter changes. We now place α, β, and γ on a numerical axis (Figure 2c). Choosing α arbitrarily as a reference at 0, then the locations of γ and β are 2 and 3 respectively. That is, each of these objects is placed at the number of attribute changes required to transform it to the reference object (α in this case).

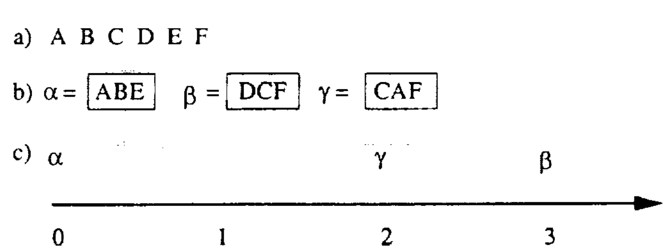

Figure 2. Distance between members of a discrete set

A definition of a *distance* between objects that are identified by attributes chosen from a finite set emerges from the example in Figure 2.

- The *distance* between two objects characterized by discrete attributes drawn from the same set is equal to the number of attributes that must be changed in one object so that it is identical to the second object in terms of its attributes.

The distance defined above might also be interpreted as the dissimilarity between two objects. In this interpretation, the null distance corresponds to a complete similarity (no dissimilarity) while the distance equal to the total number of attributes in the object (three in Figure 2) signifies a complete dissimilarity (no similarity). Note that the properties of the similarity and dissimilarity are mutually complimentary–their measures add to 1.

Now consider the case when the order of attributes in the object does matter. To illustrate the consequence, consider a transformation of γ to α. In addition to replacing the letters C and F with B and E respectively, one needs to change to position of A in γ from position 2 to position 1. Counting the two letter replacements and the position exchange results in γ to α distance of 3. Although this describes correctly how to construct the transformation of γ into α we will be interested in sampling among all the objects at a prescribed distance e between the two objects.

The tacit assumption in these examples is that the number of attribute positions in each object are the same. Although this need not be the case in general, it will be true for the problems we study. Given the aforementioned distance definition we can now formally describe a discrete version of BCB. We focus on the case when the positions of the attributes do not matter since that is the only type of discrete optimization problems we solve in this manuscript. The symbols P_1, P_2, and C (two parents and the resulting child) correspond to α, β, and γ respectively in the previous discussion. Let k denote the number of attribute positions in P_1 and P_2, let n denote the number of possible attribute values ($n > k$), and let r denote the number of attributes that are dissimilar between P_1 and P_2.

- Step 1. Place P_1 and P_2 on a numerical axis ranging from 0 to the number of dissimilar components between P_1 and P_2. Without loss of generality, let P_1 serve as the reference point at 0 on this numerical axis.

- Step 2. Create a truncated discrete normal distribution on the axis described in step 1. The distribution is centered on a point M whose location may be at the midpoint between P_1 and P_2 or it may be shifted toward the fittest parent. The distribution is truncated at P_1 and P_2. The parameter σ_m must be chosen by the user.

- Step 3. Let e denote the sample value chosen from the distribution in step 2. Place the child B at a distance e from P_1 on the numerical axis between P_1 and P_2.

- Step 4. Reorder the attributes of P_1 and P_2 so that the $k - r$ attributes in common appear in the last $k - r$ attribute positions. Assign the remaining r attributes of P_1 and P_2 randomly to the first r attribute positions respectively within P_1 and P_2. Label these reordered parents $\bar{P}_1$ and $\bar{P}_2$. Construct a child B of P_1 and P_2 as follows. First, set $B = \bar{P}_2$. Then re-assign the attributes contained in the first $r - e$ positions of P_1 to the first $r - e$ attribute postitions of B.

- Step 5. Construct the final child C from B by randomly mutating B. That is, with a small probability of occurrence, p, randomly (uniformly) replace any attribute of B with any of the allowed attribute values.

As our first experiment with discrete BCB we consider a actuator location problem to dampen the noise in the interior of a cylinder. In this example an m by n matrix contains the relevant data. The m rows denote the noise measurements taken at m microphones in the interior of a cylinder for a single frequency disturbance (200 Hz.). The n columns correspond to the available sites for actuator locations. The actuator locations are on the surface of a cylinder and their purpose is to produce an anti-noise signal to cancel out the noise produced by the single frequency noise source. Only k actuators (number of attribute positions) may be selected from the $n = 102$ possible locations (attributes). (Order does not matter in this example.) We compare the performance of discrete BCB against a tabu search code taken from Kincaid et al. (1997). The performance measure is the decibel (db) reduction in the noise level. We note that the db scale is logarithmic. Table 0 summarizes a comparison of our discrete BCB versus tabu search for an actuator placement problem on a cylinder. As can be seen from Table 0 BCB does as well or better than tabu search for all k and with far fewer solutions examined for $k > 4$. Buoyed by the success of the discrete version of BCB we proceed to examine an approach that combines continuous BCB with discrete BCB.

We note that many discrete optimization problems can be modeled as selecting k out of n columns (attributes) for a given m by n matrix. In these types of problems a performance measure is given that maps the entries in the appropriate m by k submatrix to a single number. Given our previous distance measure for discrete objects in which order does not matter two parent solutions $\vec{P}_1$ and $\vec{P}_2$ are neighbors if it is possible to exchange one of the k indices in $\vec{P}_1$ with one of the $n-k$ indices not in $\vec{P}_1$ and arrive at $\vec{P}_2$. That is, $\vec{P}_1$ and $\vec{P}_2$ must have $k-1$ indices in common.

k	Best BCB (db)	# Solns. Exam.	Best Tabu (db)	# Solns. Exam.
4	-14.2	20,000	-14.2	23,000
8	-19.5	20,000	-19.5	45,000
14	-23.7	20,000	-23.7	62,000
16	-25.2	20,000	-25.2	83,000
20	-27.0	20,000	-26.3	100,000
32	-33.2	20,000	-32.2	116,000

Table 0. Actuator Selection: BCB versus tabu search (k out of 102)

3. Description of mixed BCB.

In this description it is convenient to let the discrete variables represent a discrete set of design elements (e.g. beam cross-section types) while the continuous variables represent the dimensions of the design elements (e.g. thickness, height etc. of the beams). Suppose that our structure has b beams, and that there are t possible beam types. Further suppose that the cross-section of each beam type can be described by n continuous variables We have examined two basic approaches to handling the discrete and continuous variables. The first approach is a bi-level method. That is, the main objective is to choose an optimal set of design elements (beam types), while the secondary objective is to optimize the dimensions of the design elements. In this approach a solution is represented as $(y_1, y_2, \ldots, y_b)$ where y_i is in the range $[1, t]$. Thus, the value of y_i indicates which beam type we have chosen for beam t. Recombination is done by discrete BCB.

Evaluation of the fitness of a solution requires a secondary optimization. If continuous BCB is the optimizer of choice then a population of solutions of the form

$$(x_{11}, \ldots, x_{1n}; x_{21}, \ldots, x_{2n}; \ldots; x_{t1}, \ldots, x_{tm})$$

is constructed, where x_{ij} represents the jth cross-section decision variable of the ith beam type. The continuous version of BCB would be applied to this population, and when BCB terminates, the fitness of the best solution would be associated with the original $(y_1, y_2, \ldots, y_b)$.

The second approach is at one level. That is, we treat the discrete and continuous variables as a single collection of decision variables. As before suppose that our design structure has b beams, and that there are t possible beam types. Further suppose that the cross-section of each beam type can be described by n continuous variables. Then a 1-level solution is a vector $\vec{x}$ that is a concatenation of b vectors of the form

$$(x_{11}, \ldots, x_{1n}; x_{21}, \ldots, x_{2n}; \ldots; x_{t1}, \ldots, x_{tm})$$

with one such vector for each beam. Here, x_{ij} represents the jth cross-section decision variable of the ith beam type. Thus, a solution will describe not just b beams, but $b*t$ beams. A second vector $(y_1, y_2, \ldots, y_b)$ indicates the chosen set of beam types, where y_i is in the integral range of $[1, t]$. Hence, the value of y_i indicates which beam type we have chosen for beam i. Given two parents of the form described above, continuous BCB recombines the $\vec{x}$'s and discrete BCB recombines the $\vec{y}$'s.

We have tested both the 1-level and 2-level approach on the problems described in the next section. The 2-level approach required an order of magnitude longer runtime and was less consistent in identifying high quality solutions. Hence, in the next section we report only on the computational results for the 1-level approach. This does not mean that the 2-level approach is without merit. There are at least two ways in which this is so. First, as the number of discrete decision variables increases the runtime increases more quickly for the 1-level than for the 2-level approach. Second, the 2-level approach allows for the possibility of using a different solver for each level.

4. Mixed continuous and discrete: 2-dimensional shape problems.

Throughout this section we report solely on the 1-level BCB solution approach. We begin with a 2-dimensional mixed continuous and discrete optimization problem in which 5 shapes must be selected so that the total perimeter of the 5 shapes minus the maximum difference in perimeter between any pair is maximized. There are 3 choices for the 5 shapes–a circle, a square or a right isosceles triangle. The continuous variable is the radius (r) for the circle, the length of the base (b) for the triangle and the length of a side (s) for a square. We have two additional constraints. The total area must be less than 100 units and at most 4 shapes may be triangles. Note that the perimeter to area ratios are: 2/r for a circle, 4/s for a square, and $(4+2\sqrt{2})$/b for a triangle. Thus, triangles contribute more to the perimeter per unit area and circles contribute the least. The optimal set of shapes then for our constrained problem will be four triangles and one square. It is straight forward to determine the optimal value of the triangle base b and the side s of the square. Hence, the optimal solution and objective value (103.346) is known a priori. Table 1 shows that the correct shapes are determined first. The optimization of the values for b and s take longer to compute.

Next is a related problem whose optimal solution is more difficult for BCB to find. We still wish to select 5 shapes from a list of three (circle, triangle, and square), but now we order the five positions 1 through 5. The goal is to choose 5 shapes (possibly with repeats) and the dimensions of those five chosen shapes to minimize the total perimeter. Each chosen shape's perimeter is then weighted by its position in the list: the first shape in the list is weighted by 5, the second by 4, the third by 3, and so on. The continuous variable is, as before, the radius (r) for the circle, the length of a side (s) for the square and the length of the base (b) for a right isosceles triangle. The constraints, however, are different than before. The total area must be greater than 100 units and each continuous variable has a lower bound of 1.0 and an upper bound of 10.0. The optimal set of shapes is 4 triangles and 1 circle with an objective value of 82.9. The circle has the smallest perimeter to area ratio (2/r) and is chosen for position 5 with r just large enough so that 100 - (sum of the 4 triangles area) is satisfied. Small triangles are chosen for the remaining 4 shapes since its perimeter to area ratio is largest. Table 2 summarizes the computational results. We see that the convergence to the optimal shapes proceeds at roughly the same rate as for the optimal values of b and s. We conjecture that BCB has a more difficult time selecting the optimal set of shapes because a solution of 4 squares and 1 circle is a nearby local optimum with an objective value of about 90.7.

In an attempt to understand how BCB scales with respect to the number of discrete variables we solve the position dependent shape problem with the addition of regular polygons with 5, 6, 8, and 10 sides. Increasing the number of possible shapes exhibits a linear relationship with run time in Table 3. We note, however, that the mean solution quality degrades as does the frequency of identification of the optimal shapes as the number of possible shapes increases.

# Sols	Mean Best Obj	Min Best Obj	Freq. of Opt. Set of Shapes
2000	72.4	77.841	1/15
3000	97.9	100.541	6/15
4000	102.2	102.469	10/15
6000	103.3	103.332	12/15
8000	102.7	103.320	14/15
10000	102.9	103.328	15/15
20000	103.3	103.346	15/15

Table 1. Solution quality: popsize varies while numgens is constant

# Sols	Mean Best Obj	Min Best Obj	Freq. of Opt. Set of Shapes
2000	72.4	77.841	1/15
2000	92.3	87.4	0/15
4000	92.4	83.7	2/15
6000	88.3	82.9	4/15
8000	87.3	82.9	2/15
10000	86.5	82.9	3/15
20000	83.7	82.9	8/15
50000	83.8	82.9	10/15

Table 2. Solution quality: popsize varies while numgens is constant

# Possible Shapes	Mean Best	Min Best	Freq. Opt. Set Shapes	Time
3	83.7	82.9	10/15	28.4
4	87.7	82.9	4/15	34.5
5	90.6	82.9	4/15	40.4
6	89.9	82.9	4/15	46.3
7	94.3	82.9	2/15	52.5

Table 3. Computing time as a function of discrete variables (15 reps, 20000 soln.)

Table 4 records the number of solutions needed to roughly keep the mean best objective and the frequency of the optimal set of shapes roughly the same as the number of possible shapes. In Table 4 below we use # as an abbreviation for 'Design Number'. This same abbreviation appears in Tables 6 and 8. Here it is harder to determine the exact relationship between the number of shapes and run time. It appears to have a big jump from 3 to 4, hit a plateau for 4, 5, and 6, and then make another jump form 6 to 7.

#	Mean Best	Min Best	Freq. Opt. Set Shapes	Time	# Soln.
3	83.7	82.9	10/15	28.4	20,000
4	83.2	82.9	11/15	138.3	80,000
5	83.2	82.9	11/15	162.7	80,000
6	83.9	82.9	8/15	187.2	80,000
7	83.4	82.9	11/15	427.3	160,000

Table 4. Computing time as a function of discrete variables (15 reps.)

5. Test Case Descriptions.

Our standard test problem in Sobieszczanski-Sobieski et al. (1998), Kincaid et al. (2000a) and Kincaid (2000b) has been a minimum weight (volume) design of a hub structure also found in Balling and Sobieszczanski-Sobieski (1994). Each member of the hub is an I-beam rigidly attached to the hub and to the wall. The beam cross-sectional dimensions are the design variables, and the constraint functions reflect the material allowable stress as well as overall and local buckling. The top and bottom flanges of the I-beam are not necessarily of the same dimensions. Hence, the cross-section of each I-beam requires six design variables. Figure 3 illustrates a 2-member hub problem. Additional details may be found in Padula et al. (1997). The utility of the hub structure as a test case stems from its ability to be enlarged by adding as many members as desired without increasing the dimensionality of the load-deflection equations. These remain 3 by 3 equations for a 2-dimensional hub structure regardless of the number of members. While analytically simple, the hub structure design space is complex because the stress, displacement and buckling constraints are rich in nonlinearities and couplings among the design variables. Our goal is to extend this problem so that, in addition to choosing the dimensions of the beams so as to minimize the volume of the hub frame, we would also select (among a finite list) the beam type.

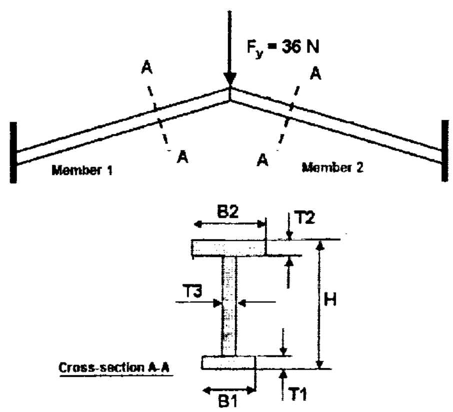

Figure 3. 2-member hub description

As a first step in this direction we consider 3 beam cross-section types—I, circular, and triangular. In all cases the overall length of the beam is held constant amongst the cross-section types. The I-beam has 6 design variables; the circular beam has 2 design variables; and the triangular beam has 3 design variables. We consider the 2-beam hub design problem, yielding 9 possible beam design combinations (e.g. (I,I), (I,circular), (I, triangular) etc.). To determine the quality of the designs found by BCB we examine each of the 9 possible beam design combinations and solve the resulting continuous hub problem with the continuous version of BCB. We repeat this 50 times and the best solutions found for each design are given in table 5. In the design column we let $1 = I$, $2 = circular$, and $3 = triangular$.

Design	rank	Min Vol	Mean Vol	Std. Dev.
1,1	4	570.60	573.27	1.98
1,2	2	569.15	572.53	4.62
1,3	8	596.60	599.02	2.32
2,1	3	570.31	574.12	3.27
2,2	1	568.91	571.05	2.55
2,3	6	593.30	596.69	1.67
3,1	7	594.16	596.69	2.12
3,2	5	591.88	592.73	0.66
3,3	9	617.13	618.44	1.33

Table 5. 2-beam hub design enumeration: 50 reps of continuous BCB

We see that most designs yield similar minimum volumes, but there is a range of about 50 units from the best to the worst designs. We take these minimum values in Table 5 to be our "known" optimal solutions. Next, we apply mixed BCB to the 3 possible shape, 2-beam hub design problem. The choice of beams is no longer fixed. Table 6 records the frequency with which each design was chosen, along with the minimum volume identified for each chosen design. The table is arranged with the best known designs at the top and the worst at the bottom. Ideally, the frequency of design (1, 3) would be nearly 100%.

#	Freq %	Known Min Vol	Min Vol Found	Mean Vol Found	S.D.
2,2	54	568.34	568.80	576.98	6.84
1,2	2	569.15	592.15	592.03	NA
2,1	20	570.31	570.32	582.10	9.18
1,1	0	570.60			
3,2	4	591.88	595.07	603.09	8.02
2,3	18	593.30	592.81	602.00	9.99
3,1	0	594.16			
1,3	0	596.60			
3,3	2	617.13	623.57	623.57	NA

Table 6. 2-beam hub design optimization: 50 reps of mixed BCB

We can make several observations from the data. The best design, (2,2), was identified over half the time. The second best design was only selected 2% of the time. Several poorer designs were identified more frequently than some better designs. The designs that include circular beams tend to be more frequent, even though they are not the best. This last observation suggests that there is some bias towards choosing circular beams. One explanation is that due to its simpler shape (2 design variables) it's continuous variables are optimized more quickly than the complex I-beam shape (6 design variables). That is, early in the search, circular beams appear more attractive than I-beam or triangular beams, because they reach their optimal shape more quickly. To test this hypothesis we construct an example in which each of the beam shapes have the same number of design variables.

We consider three possible beam types: triangle (shape 3), rectangle (shape 4), and a U shape (shape 5). Each of these shapes have 4 design variables. As before we consider the 2-beam hub structure, which gives 9 possible designs. To determine how good each design is, we enumerate all possible designs, fixing the choice of shapes in each case. Based on 50 replications in table 7, we accept the following minimum volumes as the best "known" solutions. The range from best to worst is approximately 50 units.

Next we apply mixed BCB (level-1 approach) to the 2-beam 3-shape problem, allowing BCB to pick the shapes as well as optimize the design variables for each shape. As before we applied the algorithm 50 times. Table 8 records the frequency with which each design appears out of the 50 replications, along with the minimum volume, mean volume, and standard deviation of the volumes for each chosen design. The table is ordered so that the best "known" designs are at the top.

In each design above, the minimum volume was relatively close to the known best volume. In fact, when we consider the interval [(mean volume - S.D. volume),(mean volume + S.D. volume)] for both the known and the experimental results, we see that these intervals overlap for each of the 9 possible designs.

As indicated in table 8, the known best design was identified most often and the top three designs were identified 60% of the time. Yet, some poorer designs were still identified more frequently than we would have liked. For example design (4, 5) was identified as the best design in 10 percent of the replications. Consequently, our hypothesis that inequality in the number of design variables per shape biased the results in table 6 is only partially supported by the results in table 8. It remains to explore further the interplay between the discrete and continuous BCB algorithms for mixed design optimization problems.

6. Conclusions.

We began by developing a discrete version of BCB and testing its performance against a tabu search code on a previously studied problem. Next, we proposed two ways to link the discrete and continuous version of BCB to solve mixed integer nonlinear programming problems. Preliminary experiments indicated that the 1-level version was superior to the 2-level approach for a modest number of discrete variables. Further testing was completed for the 1-level version for 2-dimensional shape optimization problems. Finally, the 1-level version of BCB was applied to a standard hub design problem in which the beam type is to be selected in addition to determining the continuous dimensional aspects of the beam.

A 2-beam hub design problem in which 3 beam types are available was examined first. Here it was found that the 1-level approach was biased towards the beam type with fewest dimensions. It was conjectured that if all the beams had the same number of dimensions then the biasing would disappear. Towards this end a second 2-beam hub design problem was tested in which 3 beam types, each with 4 continuous design variables, were available for selection. The results of this latter test were encouraging but still did not fully explain why some poorer designs were chosen with relatively high frequency.

In addition to gaining a better understanding of the interactions between the discrete and continuous version of BCB for mixed integer nonlinear problems our future research includes further tests for the discrete BCB algorithm by itself. In particular, we will develop and test versions of discrete BCB for problems in which the order of the attributes matter and when parent solutions are allowed to have an unequal number of attribute positions.

Design	rank	Min Vol	Mean Vol	S.D. Vol
3,3	4	592.527	606.995	9.132
3,4	2	570.411	579.438	18.332
3,5	9	612.744	630.374	14.484
4,3	7	610.161	621.382	7.494
4,4	5	601.918	607.150	4.079
4,5	8	612.488	624.700	7.642
5,3	3	580.845	596.797	17.342
5,4	1	560.001	567.763	8.399
5,5	6	606.040	630.778	26.936

Table 7. 2-beam hub design enumeration: 50 reps of continuous BCB

#	Freq %	Known Min Vol	Min Vol Found	Mean Vol Found	S.D.
5,4	26	560.001	561.124	585.633	66.195
3,4	16	570.411	571.198	577.874	5.884
5,3	18	580.845	596.897	620.329	30.450
3,3	8	592.527	577.540	589.838	10.278
4,4	4	601.918	601.950	604.141	2.211
5,5	8	606.040	610.614	624.926	18.084
4,3	4	610.161	618.275	623.161	4.886
4,5	10	612.488	612.781	627.202	12.751
3,5	6	612.744	600.438	670.673	98.640

Table 8. 2-beam hub design optimization: 50 reps of mixed BCB

7. References.

1. Back, T. (1996) Evolutionary Algorithms in Theory and Practice, Oxford University Press.

2. Balling, R.J. and J. Sobieszczanski-Sobieski, "An Algorithm for Solving the System-Level Problem in Multilevel Optimization," ICASE Report No. 94-96 and NASA Contractor Report 195015, December 1994.

3. Dasgupta, D. and Z. Michalewicz (1997) "Evolutionary Algorithms—An Overview," in Evolutionary Algorithms in Engineering Applications, Dasgupta and Michalewicz (eds.) Springer, 3-28.

4. Glover, F. (1998) "Genetic Algorithms, Evolutionary Algorithms and Scatter Search: Changing Tides and Untapped Potentials," Informs Computer Science Technical Section Newsletter, Vol. 19, No. 1.

5. Kelly, J.P., M. Laguna, and F. Glover (1994) "A Study on Diversification Strategies for the Quadratic Assignment Problem," Computers and Operations Research, Vol. 22, No.8, pp. 885-893.

6. Kincaid, R.K., K. Laba, and S. Padula, "Quelling Cabin Noise in Turboprop Aircraft via Active Control," *J. of Combinatorial Optimization* 1, Issue 3 (1997) pp. 1-22.

7. Kincaid, R.K., M. Weber and J. Sobieszczanski-Sobieski, "Performance of a Bell-Curve Based Evolu-

tionary Optimization Algorithm," to appear in *Proceedings of the AIAA SDM conference*, April 2000.

8. Kincaid, R.K., M. Weber and J. Sobieszczanski-Sobieski, "An Atypical ($\mu + \mu$) Evolutionary Algorithm," submitted for publication 2000.

9. Padula, S.L., N. Alexandrov, and L.L. Green, "MDO Test Suite at NASA Langley Research Center," in *Proceedings of the 6th AIAA/NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellvue, WA, September 1996, AIAA, 410-420. (Accessible as a website http://fmad-www.larc.nasa.gov/mdob/)

10. Reeves, C.R. (1997). "Genetic Algorithms for the Operations Researcher," Journal on Computing, Vol. 9, No. 3, 231-250 (with commentaries and rejoinder).

11. Sobieszczanski-Sobieski, J., K. Laba, and R. K. Kincaid, "Bell-Curve Based Evolutionary Optimization Algorithm," in *Proceedings of the 7th AIAA/USAF /NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO., September 2-4, 1998. AIAA Paper 98-4971, pp. 2083-2096.

12. Vanderplaats, G.N. "CONMIN-A Fortran Program for Constrained Function Minimization," NASA TM X-62282, August, 1973.

13. Vanderplaats Research & Development, DOT Users Manual, 1767 S. 8th Street Suite 210, Colorado Springs, CO 80906, 1995.