Diversity preservation in multimodal functions using explicit diversity control and adaptive distance based algorithm

Shantanu Chandra, Phillip Lippe, Mahsa Mojtahedi, Rick Halm

***Abstract*— We investigate the performance of two diversity preserving evolutionary algorithms that balance exploration and exploitation over time. The first approach adapts the weights of multiple objectives according to a predefined path while the second algorithm adapts the neighborhood radius of the traditional fitness sharing algorithm. We analyze the effects of both methods on the performance, population distribution and maintenance of multiple optima. Evaluated on the multimodal function Katsuura, both algorithms perform significantly better than standard EA and traditional fitness sharing.**

I. INTRODUCTION

One of the greatest challenges in optimizing multimodal functions is overcoming premature convergence and loss of diversity. An established way to prevent this is by using niching. To this end, techniques such as crowding and fitness sharing are used that enable the user to find multiple optima in the search space. In order to find the global optimum, however, the algorithm needs to start exploiting the optima it has found at some point. Forms of adaptive fitness sharing [1] have been proposed that achieve this by reducing the effects of fitness sharing over time. We investigate the performance of two difference methods of adaptive niching on Katsuura. The complete code used for this paper is available online[1].

The first method, called *Explicit Diversity Control* (EDC), has the advantage of adding a weighted measure of diversity directly to the fitness value of each individual. Consequently, the diversity of the population can be controlled more directly, and is coaxed towards following a predefined, gradually decreasing line. The variables used in defining this line are tuned by another EA, so this method adds no supplementary variables to the problem.

The second method is a form of adaptive fitness sharing, where the radius of neighbourhoods is reduced over time. This method gives the user more control over the point at which the algorithm will focus on exploitation.

In the next section, existing diversity preservation methods are discussed. Section 3 describes our method in detail. In section 4, the results of testing our method and adaptive fitness sharing on Katsura are presented. These results are compared in section 5. Section 6 provides the conclusions.

II. RELATED WORK

A. Fitness sharing

Multiple niching techniques have been introduced to counter premature convergence to local optima and therefore explore the solution space more thoroughly. One of the most well-known techniques is fitness sharing. In principle, fitness sharing divides the fitness over all individuals in a certain neighbourhood. In this way, densely populated regions receive reduced fitness and this encourages search in unexplored areas [2]. The fitness function is defined as follows:

$$f_i^{'} = \frac{f_i^{\beta}}{m_i} \text{ with } m_i = \sum_{j=1}^{N} sh(d_{ij}) \tag{1}$$

$$sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_s}\right)^{\alpha} & \text{if } d_{ij} < \sigma_s \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where f_i is the true fitness and m_i is the number of individuals within its radius σ_s. Also, d_{ij} is the distance measurement between individuals and α determines the shape of the sharing function and is commonly set to one [3].

The parameter β tackles the issue where an individual at the optimum shares its fitness with more individuals compared to an individual at the edge of the hill and, hence, has a lower shared fitness. Therefore, by powering the value of the fitness f_i with β, the shared fitness $f_i^{'}$ makes the optimum more attractive than the surrounding region [4].

B. Issues relating fitness sharing

There are two issues identified with fitness sharing. First of all, finding the optimal parameter value for σ_s requires *a priori* knowledge of the fitness landscape [3]. Also, σ_s is the same for all individuals, hence only works properly if all optima are in equal distance from each other over the whole domain. If this is not the case, it is possible that some individuals share the fitness with individuals on other optima which is undesirable. Another issue relating σ_s is that it is fixed for all cycles. Hence, the algorithm remains exploring while there might be a desire to exploit some peaks in later cycles of the algorithm [2], [5], [6].

Secondly, fitness sharing and other niching techniques remain to have difficulties with maintaining multiple local optima. When the expected number of individual around a peak drops close to one, the peak loses all representation in the population [7]. Therefore, fitness sharing is hardly scalable to high dimensional multimodal functions [2].

C. Multiobjective algorithm

Though niching is aimed at preserving diversity, it is not always successful. This is probably since it optimizes just one objective at the cost of other. Therefore, multiobjective algorithms have been developed in order to balance multiple

[1]Code: https://github.com/phlippe/EC_Assignment

objectives. The optimal solution is most often defined as Pareto optimal when "there is no feasible solution X that will improve some objective values without degrading performance in at least one other objective" [8]. The following section discusses how fitness and diversity can be balanced and changed over time.

III. METHODOLOGY

Keeping in mind the characteristic of a multimodal function, it is evident that a pure distance or pure fitness based optimization function is not enough to search the genotype space for the global optima and prevent the EA from converging to local optima in a subspace. We therefore compare different diversity preserving algorithms in the following section.

A. Explicit Diversity Control (EDC)

The *Explicit Diversity Control* algorithm optimizes a weighted sum of the *fitness* and *diversity* metrics of an individual to ensure that the evolution favors exploration and exploitation in the desired stages. The objective function of this algorithm is defined as follows:

$$\delta_i = \frac{\sum_{j=1}^{N} d_{ij}}{N} \tag{3}$$

$$CF_i = \alpha \cdot f_i + (1 - \alpha) \cdot \delta_i \tag{4}$$

where f_i is the fitness evaluation score of the individual and δ_i is the mean distance to all other individuals. This objective function assigns a comprehensive score to an individual and prevents premature convergence to local optima. It does so by reducing the objective of the individuals who tend to optimize either fitness or diversity at the cost of other.

The transitions between *exploration* and *exploitation* is controlled by α, defined as:

$$\alpha = \frac{\mu_{currentDist}}{\mu_{idealDist}}$$

where, $\mu_{currentDist}$ of the population is the sum of the distances between individuals averaged over the population size, and the $\mu_{idealDist}$ is the ideal mean distance that an algorithm with optimal balance between exploration and exploitation might have.

As we expect the population to have a high diversity during exploration and a lower mean distance during exploitation, the function of $\mu_{idealDist}$ should decrease over time and slowly approaches 0. One function that satisfies these properties is:

$$\mu_{idealDist}(t) = \left(\frac{1}{a \cdot t + b} - 1\right) \cdot c$$

where the parameters *a* and *b* are used to control the t- and the *y-intercept* of the curve respectively, and *c* is responsible for the slope. The time is measured by the number of generations, and negative values are mapped to 0. Figure 1 visualizes the function for different values of *c*.

The function is adaptive as it gradually transitions from *exploration* to *exploitation* over generations. The $\mu_{idealDist}$ is a known constant value for each generation. As a result, when the $\mu_{currentDist}$ begins to decrease too rapidly over generations, the weight of the f_i contribution (α) goes down in the objective function of the individual and that of its diversity metric $(1 - \alpha)$ goes up. Hence, for the next generations, the individuals with higher diversity dominate the ones with higher fitness and are more successful in surviving as well as creating offspring. This restores the diversity of the EA population by encouraging it to keep exploring and preventing it from being dominated by fitter individuals too early in the process. Consecutively, if the *current mean distance* drops at a slower rate than expected, then the weight of the fitness contribution(α) increases for the consecutive generations compared to previous ones. As a result, highly fit individuals get a higher objective score and dominate the fitness space for a series of runs until an acceptable overall fitness of the population is restored without compromising significantly on the diversity.

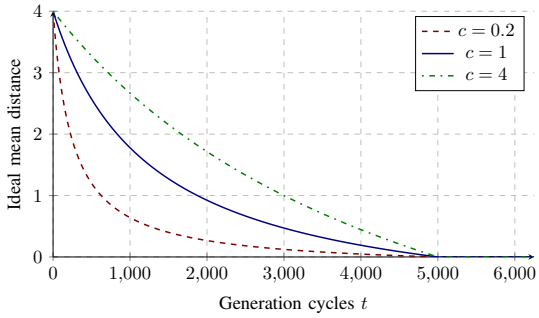

Fig. 1: Ideal mean distance of individuals over generations for $c = 0.2, 1$ and 4. For this example, the initial value is set to 4, and the last iteration to 5,000 compared to 6,250 overall generations.

B. Fitness sharing with adaptive radius

As discussed in Section 2.B, one of the most popular diversity preserving algorithm is the *Fitness Sharing*. In the classical setup, the σ_s is a user-defined value that stays constant for the entire course of the run.

However, as we have observed with the mutation step sizes, we require different values of the step size σ_s at different stages of the evolution. We can adopt a similar adaptive strategy for the σ_s in *Fitness Sharing* as well. The advantage of doing that is two folds. Firstly, we adapt the size of the neighbourhood in line with the stage of evolution (exploring in the beginning and exploiting later). Secondly, we reduce the dependency of the performance of the *Fitness sharing* on the initial initialization of the σ_s.

We do so by varying the neighbourhood size of an individual (controlled by σ_s) over the course of the run. We keep the neighbourhood size constant for the initial X generations and then linearly decrease it for the later stages. The ideal break-up point X depends on the problem at hand and its effects are discussed in the section *Analysis and Discussion*. This ensures that initially, the distant points contribute to the *shared fitness* of the individual more which

helps in preserving diversity. Later, as the σ_s decreases, only the nearby points influence the fitness of the individual and hence this can be seen as a localized exploitation stage.

C. Experimental Setup

The specifications of the EA optimizing the multimodal function *Katsuura* are given below:

Representation	Real-valued
Parent Selection	Tournament
Survivor Selection	Round-Robin Tournament
Mutation	Self-adaptive Multi-Sigma
Recombination	Uniform crossover
Population size	400
Generation gap	0.4
Tournament size	2
Mutation step size init	0.01
Crossover rate	1.0

TABLE 1. EA specification

We have chosen the genotype space to be a one-to-one mapping of the phenotype phase and hence each individual is a 10 dimensional real valued vector with permissible values in the range $[-5, 5]$. The *parent selection* method is *tournament selection* to easily regulate the selection pressure (lower is preferable in this experiment). The survivors for the next generation were (in a $\mu + \lambda$ steady state setup) determined by round-robin tournament. The self-adaptive sigma ensures an adaptive mutation over time and aids in initial exploration and later exploitation. The recombination is done by *uniform crossover* as the genes are not related to each other and their relative position with respect to each other holds no value. The population size is 400. A population of 100 would not have provided enough 'diversity' in the initial population and they would not have been spread over the landscape (essential for a function like *Katsuura*). A population of 1000 would have meant exponentially more evaluations in each run and we would have exhausted our budget of 1 million evaluations without fully exploiting the peak. Keeping the evaluation budget in mind, the number of offsprings are 160 and thus using a $(\mu + \lambda)$ steady state survivor selection was adopted to maintain the population size. The initial mutation step size was set at 0.01.

The a, b and c parameters for the $\mu_{idealDist}$ function are tuned using another EA running on top of the principal EA. Since the parameters are completely tuned by an EA and involve no manual intervention, the proposed model can generalize well over other multimodal functions as well as none of the elements of the algorithm are specially crafted for the problem function of this experiment i.e., *Katsuura*. The representation of the parameters in the genotype space is using a real-valued vector of the y-intercept, initial value and slope of the curve. The permissible range of the y-intercept is 0 to 6250 (maximum number of generations with a population size of 400, 160 offsprings and maximum of 1 million evaluations), 0 to 30 for the initial value (as the max distance between two individuals is $10 \cdot \sqrt{10}$), and a log scale of 0.01 to 10 for the slope. Based on these parameters, a, b and c are determined and the fitness of an individual is the

Representation	Real-valued
Parent Selection	Tournament
Survivor Selection	Round-Robin Tournament
Mutation	Gaussian mutation
Recombination	Uniform crossover
Population size	24
Generation gap	0.5
Tournament size	4
Mutation step size	0.05 of parameter range
Crossover rate	1.0

TABLE 2. EA on EA specification

average score of the EDC algorithm with these parameters over 10 different seeds. As this process is computationally expensive, we limit the population size to 24 with a higher selection pressure.

For the adaptive *fitness sharing* algorithm, σ_s was initialized with 0.03. This is because our expected movement by mutation is 0.01 times $\sqrt{10}$ to start with. After some iterations, σ_s decreases linearly to 0 and stays constant for the rest of the run. Different adaptive functions of σ_s were tested whereas balancing all three stages equally achieved the best results. Additionally, we compare this to adapt β from 1 to 10 in a similar manner.

IV. EXPERIMENTAL RESULTS

The performance scores of the different algorithms in Figure 2 (averaged over 1000 random seeds) show that the fitness sharing with adaptive σ_s performs better than the adaptive β variant and the Explicit Distance Control algorithm, with all scoring on average above 9.3. The standard EA scored 8.85, but the traditional fitness sharing without adapting any parameter could just manage a score of 5.71. The mean scores and standard deviations are shown in Table 3.

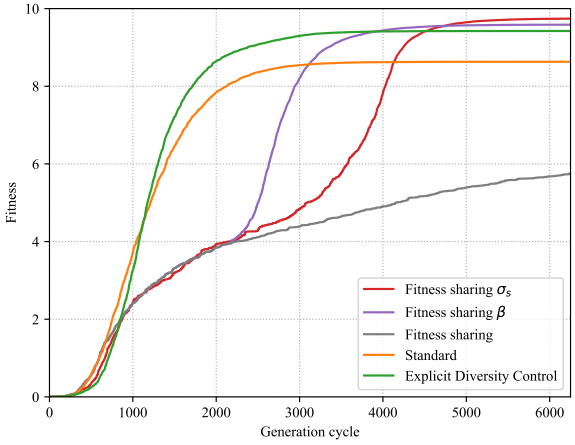

Fig. 2: Visualizing the max fitness over generations of different methods, averaged over 1000 experiments

Also, the statistical t-test which tests for significant differences between the two proposed algorithms and the others are visualized in the table. The results show that there is a significant difference between the algorithms with p-values lower than 0.001. Therefore, fitness sharing with adaptive σ_s

performs significantly better than all other algorithms and EDC performs significantly better than the standard EA and classical fitness sharing.

Method	Mean	SD	t_{σ_s}	t_{EDC}
Standard	8.85	1.24	20.99*	10.99*
Fitness sharing	5.71	1.06	108.28*	92.54*
Fitness sharing σ_s	**9.74**	**0.51**	-	-15.02*
Fitness sharing β	9.63	0.53	4.73*	-10.73*
Explicit Distance Control	9.34	0.67	15.01*	-

*$p < 0.001$

TABLE 3. Statistics of maximum fitness of different methods in 1000 runs

The mean distance graph in Figure 3 shows the distance between individuals over generations. The traditional fitness sharing model converges prematurely in the very first 750 iterations of the run. The mean distance steeply drops to 0.04 and remains the same for the remaining course of the experiment. On the other hand, the fitness sharing with adapting radius σ_s and adaptive β follow the same trajectory as the traditional sharing but the mean distance decreases when the parameter start to change. This suggests that both algorithms successfully shift from exploring to exploiting after 2100 cycles.

The ideal line is found with the EA on EA as specified in Table 2 and found a and b such that the y-intercept is 4 and x-intercept equals 3000 also optimal c equals 1. We can see that the EDC algorithm is clearly pushed towards this ideal line and hence results in a higher mean distance until the 2100$^{\text{th}}$ iteration. At that point, the objective function places more weight on fitness such that diversity is ignored and begins exploitation. Concluding, both proposed algorithms show different ways of shifting from exploration to exploitation.

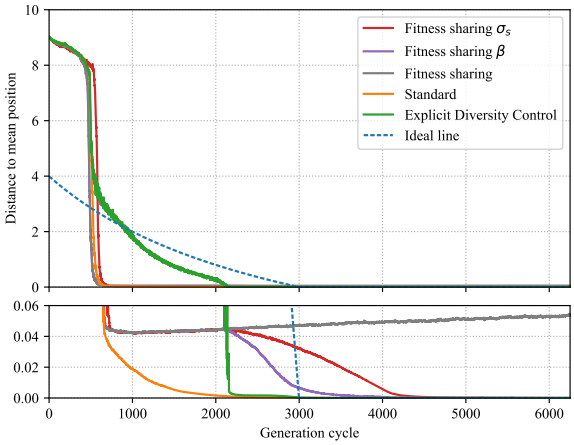

Fig. 3: Comparing the mean distance between individuals over generations for different methods. The bottom plot focuses on small distances lower than 0.06.

V. ANALYSIS AND DISCUSSION

As shown by the experimental results, preserving diversity is crucial for finding a high fitness value on a multimodal function. However, exploiting is similarly important as otherwise the algorithm would never reach the full potential of an optima. This is why making fitness sharing adaptive improves its performance remarkably, as is seen in Figure 2. During the initial stages, all three fitness sharing algorithms are identical in their procedure and therefore behave similarly. However, after 2100 generations, the algorithms with adaptive β and σ_s start to reduce fitness sharing, and this gives the individuals a chance to start exploiting. Since both adaptive algorithms have already explored the search space very well, exploiting reaches a high fitness value in a short period.

It is worth mentioning that although adaptive fitness sharing methods explore longer than the standard EA, they are doing this in a limited space. A closer look at Figure 3 shows that for all fitness sharing methods, population diversity drops significantly early into the run. This indicates that the individuals are close to each other in the genotype space and the algorithm only searches a local subspace. The reason for that is that the maximum amount a fitness score can be reduced by is the population size. Therefore, if the algorithm finds a position that is 400 times better than previous ones (which easily happens in early stages of the algorithm), the population will focus on that position. Hence, fitness sharing is not able to maintain multiple far-away optima. Nevertheless, as the testing function *Katsuura* has many optima close together, exploring in a local subspace still improves the overall performance.

Note that controlling the rate of exploration is important. If fitness sharing is not reduced on time, the discovered optimum will not be fully exploited before termination, resulting in poor performance results. Generally, σ_s gives the user considerably more control over the shifting point between exploration and exploitation in contrast to β. There are several reasons for this.

In order to stop fitness sharing completely, β has to be infinite. Practically this is not possible as β is limited by a finite value β_{max}. Thus, this maximum value is an approximation, and the slope with which the value of β changes from 1 to β_{max} is difficult to control, as β is the power of fitness, and hence increasing it will exponentially increase the importance of fitness. This causes the algorithm to switch to exploiting too rapidly. Additionally, the algorithm will always apply a small amount of fitness sharing till the end, which can slightly affect its chances to reach the hill's peak.

This is not the case for σ_s method, since the algorithm can completely cease fitness sharing when σ_s is turned to zero. Moreover, by directly influencing the neighborhood radius, σ_s, the user exercises a greater control over the length of the exploration/exploitation phase, as shown in Figure 4. If σ_s is decreased early, the algorithm starts exploiting and reaches higher fitness values in the first few generations. However, as it has explored much less, the algorithm might have chosen a lower optimum. In contrast, if σ_s is decreased too late, the algorithm explores too much and spends almost no time on exploiting. As a consequence, the algorithm runs out of evaluations while still significantly improving its score. To find the optimal adaptation curve of σ_s, one might be

able to apply a similar EA on EA approach as presented in Section III-C for EDC.

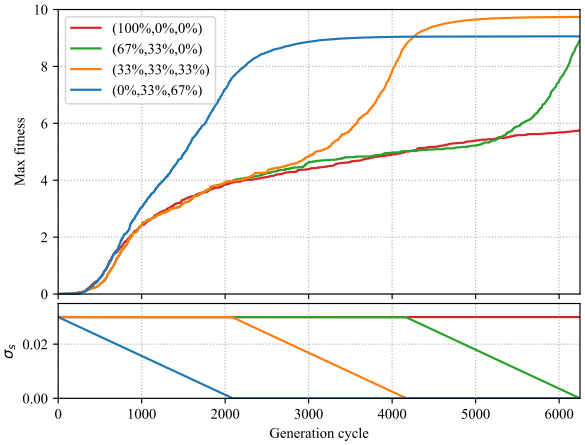

Fig. 4: Comparing the maximum fitness over generations (top) to different adaptation curves of σ_s (bottom).

Compared to the other methods, explicit distance control is significantly more successful in preserving a higher mean distance between individuals. Even fitness sharing experiences an abrupt depletion in diversity at the initial stages due to the small σ_s. Nevertheless, taking a look at Figure 2 shows that EDC has a much higher fitness at early stages of the run than with fitness sharing, although it also has a higher diversity in its population. This indicates that EDC allows the algorithm to explore **and** exploit at the same time. While some individuals are looking for a different hill (having high mean distance δ_i), a subpopulation already starts to exploit the best optimum so far (having high fitness f_i). Figure 5 visualizes this population behavior and shows that compared to the standard EA, EDC is able to maintain multiple optima (light blue points) and even finds a different optimum than the standard EA (in this experiment, EDC achieves a score of 9.9, while the standard EA reaches 5.5).

However, in EDC, the distance of an individual is proportional to its objective score (depending on $1 - \alpha$). Consequently, an optimum with the same fitness and a higher distance would be more desirable, although technically, they should be equal. Furthermore, directly summing up distance and fitness requires them to be on a similar scale. This effects how strongly the mean distance of the population differs from the ideal line (see Figure 3).

Overall, both fitness sharing and EDC improve the diversity of the population in a different manner. For the specific case of the evaluation function *Katsuura*, fitness sharing with adaptive σ_s outperforms EDC slightly.

VI. CONCLUSIONS

This paper investigates two algorithms that aim to overcome the challenge of premature convergence in the optimization of multimodal functions. One of the proposed methods adapts the radius of the fitness sharing algorithm over time. In this way, the EA explores a local space more carefully

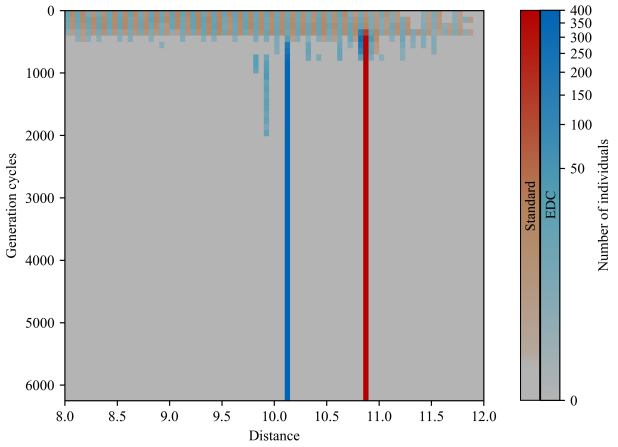

Fig. 5: Population distribution of a single run over generations (y-axis). The x-axis show the distance to the center of the search space to reduce dimension size (cropped to 8.0 to 12.0 for this certain experiment). The color indicates the number of individuals at a certain distance (blue: EDC, red: standard EA). Both algorithms used the same seed. Note that the distance scale on the x-axis does not reflect the distance between the individuals itself.

than traditional fitness sharing. The results suggest that by adapting σ_s it is possible to directly affect the shift from exploration to exploitation.

The second algorithm optimizes a bi-objective function that directly balances fitness and diversity. Also, the results suggest that it is possible to make an EA explicitly maintain a desired distance between its individuals. Both proposed algorithms perform significantly better than standard and traditional fitness sharing and show promise for future research.

VII. BIBLIOGRAPHY

[1] O. M. Shir and T. Bäck, "Niche radius adaptation in the cma-es niching algorithm," in *Parallel Problem Solving from Nature - PPSN IX* (T. P. Runarsson, H.-G. Beyer, E. Burke, J. J. Merelo-Guervós, L. D. Whitley, and X. Yao, eds.), (Berlin, Heidelberg), pp. 142–151, Springer Berlin Heidelberg, 2006.

[2] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 666–685, 2013.

[3] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE transactions on Evolutionary Computation*, vol. 2, no. 3, pp. 97–106, 1998.

[4] P. Darwen and X. Yao, "A dilemma for fitness sharing with a scaling function," in *Evolutionary Computation, 1995., IEEE International Conference on*, vol. 1, p. 166, IEEE, 1995.

[5] J. rey Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Proceedings of the first IEEE conference on evolutionary computation, IEEE world congress on computational intelligence*, vol. 1, pp. 82–87, Citeseer, 1994.

[6] R. E. Smith, S. Forrest, and A. S. Perelson, "Searching for diverse, cooperative populations with genetic algorithms," *Evolutionary computation*, vol. 1, no. 2, pp. 127–149, 1993.

[7] S. W. Mahfoud, "Niching methods for genetic algorithms," *Urbana*, vol. 51, no. 95001, pp. 62–94, 1995.

[8] Y. Tang, P. Reed, and T. Wagener, "How effective and efficient are multiobjective evolutionary algorithms at hydrologic model calibration?," *Hydrology and Earth System Sciences Discussions*, vol. 2, no. 6, pp. 2465–2520, 2005.