Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

OPTIMAL COMPUTING BUDGET ALLOCATION FOR SMALL COMPUTING BUDGETS

G. Jake LaPorte

George Mason University
4400 University Drive
Fairfax, VA 22030

Juergen Branke

Warwick Business School
The University of Warwick
Coventry, CV4 7AL, UK

Chun-Hung Chen

National Taiwan University
No.1, Sec. 4, Roosevelt Road
Taipei, Taiwan 106

ABSTRACT

In this paper, we develop an optimal computing budget allocation (OCBA) algorithm for selecting a subset of designs under the restriction of an extremely small computing budget. Such an algorithm is useful in population based Evolutionary Algorithms (EA) and other applications that seek an elite subset of designs.

1 INTRODUCTION

This paper addresses the problem of selecting the best *m* out of a finite number of *k* potential solutions, referred to in this paper as designs. Additionally, the performance measure for each design is stochastic and the simulation budget is extremely small. In the context of a simulation, the objective of the procedure that follows is to determine the best allocation of an extremely small number of simulation replications in order to identify as many of the set of elite designs as possible. By allocating the simulation budget, we resample these designs using multiple evaluations to gain more confidence in the estimated performance measures. This type of ranking and selection (R&S) problem has been considered previously in (Chen, He, and Fu 2008) and (Koenig and Law 1985) except these papers were concerned with selecting all of the top *m* solutions, and assumed a larger number of simulation replications available.

Our primary motivation for this procedure is to use it in conjunction with population based evolution algorithms (EA). An evolution strategy (ES) is a population based direct search algorithm which produces λ offspring (designs) based on mutations of some start point for each generation, evaluates these offspring according to some fitness function (performance measure), selects the μ best of these offspring based on the fitness values to be the parental population of the next generation, and uses the centroid of the parental population to be the starting point of the next generation. The distance travelled toward the optimum in solution space by the population's centroid from one generation to the next is known as the progress rate. For more details, see Beyer and Schwefel (2002). Another potential use of this procedure is screening a set of designs for an elite subset when the number of simulation replications is limited due to time or budget constraints. There are numerous applications of this situation as many simulations have a very high computational effort and decision makers often want more than one design to consider when the number of inputs into the simulation model are high.

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

For an ES, efficiency is defined as the progress rate per function evaluation. Arnold and Beyer (2000) analytically show that an ES with an offspring population size of λ with κ function evaluations (simulation replications) each cannot outperform the efficiency of an evolution strategy with population size $\lambda\kappa$ on simple noisy fitness functions. This means that a strategy which resamples each offspring κ >1 times is inefficient compared to a traditional ES. For this reason, the procedure in this paper is built in order to develop a methodology which is able to resample in an ES environment where the efficiency concerns limit the number of simulation replications available. Therefore, we consider each generation of an ES as a ranking and selection problem with an extremely small computing budget which, depending on the amount of noise in the problem, may not detect all of the members of the elite population. An assumption made now and tested in the experimental section of this paper is the correlation between increasing the number of correctly identified members of an ES's elite population and an increase in its progress toward the optimum. Therefore, the ES problem on a generational basis becomes identical to the ranking and selection problem of identifying as many of the elite population as possible with a small computing budget.

Optimal Computing Budget Allocation (OCBA) is a R&S procedure used to distribute a simulation computing budget to resample a fixed number of alternatives in order to find the best design. OCBA has been shown to be very efficient, controllable and robust when compared to other ranking and selection algorithms (Branke, Chick, and Schmidt 2007). OCBA-m is an extension of OCBA which seeks to select the best *m* out of *k* designs. It has been shown by Chen, He, and Fu (2008) that OCBA-m significantly enhances the computational efficiency of a population based incremental learning (PBIL) algorithm and a neighborhood random search (NRS) algorithm both of which require the selection of an elite subset of the original population for each iteration. OCBA-m+ is an enhancement of OCBA-m which removes the need to choose a constant to separate the elite subset from the non-elite subset and has been shown in (Zhang, Lee, and Chew 2012) to perform better than OCBA-m in a large computing budget environment. Our paper is different than these approaches since we are operating in a small computing budget environment and are interested in finding as many of the elite population as possible.

OCBA-m and OCBA-m+ are asymptotic solutions for maximizing the approximate probability of correct selection for *m* best (APCS*m*) by using a Bayesian framework relying on the posterior probability distributions of each design. The APCS*m* objective is an approximation of the probability of correct selection (PCS) which is the probability that all *m* top designs are found. In a small computing budget environment where the estimates for the mean and variance are not very reliable, many designs are likely to be misclassified and the PCS measure is likely to be small. Instead of maximizing the APCS*m*, the procedure in this paper distributes the computing budget sequentially in order to gain information about the design which has the greatest probability of changing its classification (PCC). We use the OCBA-m and OCBA-m+ procedures as the current state of the art comparing both the PCS and number of elite designs identified.

Since the asymptotic solution is not of use under these conditions, we take a Knowledge Gradient (KG) approach of "choosing the measurement that would be optimal if it were our last chance to learn." The KG approach to choosing the best member of a population of alternatives was introduced by Gupta, S.S. and Miescke, K.J. (1996), and then refined in Frazier, Powell, and Dayanik (2008) and Chick, Branke, and Schmidt (2010). The dominant theme in this process is to maximize the single-period expected increase (in a maximization problem) of the performance measure. Ryzhov and Powell (2009) formulated a decision rule for subset selection based on KG where the objective was to maximize the total reward of the subset, measurements are taken for a subset (rather than individual designs as in our case), and designs in a subset can interact in unknown ways.

The paper is organized as follows. In section 2, we formulate the problem, state the assumptions and determine the budget allocation for the next simulation replication. Implementation issues of OCBAsb are discussed in section 3. Section 4 provides experimental evidence that the procedure described here outperforms the current state of the art in the environment described. Section 5 concludes with a summary and considerations for future work.

2 FORMULATION

The formulation presented here is based on finding a subset of designs which have the minimum expected simulated performance measure.

2.1 Notation

S_m	: the current perceived set of elite designs which are classified as elite
S_t	: the true set of elite designs
N_i	: number of simulation replications allocated to design *i*
T	: the total number of simulation replications to be performed
n_0	: initial number of simulation replications per design
n	: the iteration of the procedure, there are a total of $n = T - kn_0$ iterations
T^n	: the total number of simulation replications performed after iteration *n*, $= \sum_{i=1}^{k} N_i$
$J_{i,j}$	: the *j*th simulated performance measure from design *i*
$\bar{J}_{..}$	: average performance measure for all samples of all designs, $= \frac{\sum_{i=1}^{k}\sum_{j=1}^{N_i} J_{i,j}}{T^n}$
$\bar{J}_i$	: average simulated performance measure for design i, $= \frac{\sum_{j=1}^{N_i} J_{i,j}}{N_i}$
i_r	: the index for the *r*th ordered performance measure sample mean
(i_r)	: the index for the *r*th best design

2.2 Problem Statement

A design is a p dimensional vector which contains a set of decision variables used as input into a simulation. We consider a finite set of designs $i = 1,2,\dots,k$ which have stochastic performance measures. The objective of the following procedure is to determine an allocation method which finds as many of the top *m* of *k* designs with the lowest expected performance measure as possible with the following conditions

- there is an extremely small computing budget (*T*),
- the simulated performance of each design *i* is stochastic and has a $N(\mu_i, \sigma)$ distribution,
- the rank order within each subset is not important.

If we let $J_{i,j}$ be the *j*th simulated performance and N_i be the number of simulation replications for the *i*th design, then the sample mean of the performance measure,

$$\bar{J}_i = \frac{\sum_{j=1}^{N_i} J_{i,j}}{N_i},$$

is used to estimate the true expected value. By ordering these average performance measures such that the *r*th best sample mean is $\bar{J}_{i_r}$ and $\bar{J}_{i_1} \le \bar{J}_{i_2} \le \cdots \le \bar{J}_{i_k}$, then the set of *m* designs which have the smallest perceived expected value is the set

$$S_m \equiv \{i_1, i_2, \dots i_m\}.$$

These designs have the top *m* performance sample means after $T = \sum_{i=1}^{k} N_i$ simulations have been performed and are classified as elite. The complement of S_m is the set which contains the designs which are classified as non-elite. These are the designs whose sample means are not in the top *m* and are noted as

$$S'_m = \{i_{m+1}, i_{m+2}, \dots i_k\}.$$

If we let $\mu_{(i_r)}$ be the mean of r^{th} best design such that $(i_1) \leq (i_2) \leq \cdots \leq (i_k)$, then the true set of elite designs, referred to as the elite population, is $S_t \equiv \{(i_1), (i_2), \ldots (i_m)\}$
The OCBAsb problem is to classify as many of the elite population as possible as elite using a limited simulation budget. It is shown as

$$\max_{N_1,N_2,\ldots,N_k} |S_m \cap S_t| \qquad (1)$$
$$\text{s.t.} \sum_{i=1}^{k} N_i = T.$$

Note that a system which has a cardinality of *m* in (1) has classified all of its members correctly.

In this small computing budget problem, we take an approach where the allocation procedure is initialized with as few replications as possible ($n_0 = 1$) and the computing budget is distributed one simulation at a time. With few simulation replications per design, it is likely that many designs are classified incorrectly. The budget allocation procedure which makes the most intuitive sense with these conditions is to allocate the next replication to the design which has the highest probability of changing its classification after the next simulation. If the next replication has little chance of changing the classification of a design, then there is little reason to simulate that design. To put it another way, the design which is chosen to be simulated each step should be the one which has the greatest probability of moving from one group to the other. A design changes its classification if it moves from S_m to $S_m{}'$ or $S_m{}'$ to S_m. The procedure developed in the next section is designed to decrease the probability that a design will change from one group to another by systematically resampling the designs that have the greatest probability of changing classifications each step.

2.3 Single Step Budget Allocation

Let's first examine the designs in S_m. Since the design sample means are used to classify the design, then a current member of the elite population must change its mean such that it is larger than the mean of the "best" member of the non-elite population. Let J_{i,N_i+1} represent the next sample taken of a design currently in S_m and define the mean for the current iteration, n, and the next iteration of the algorithm, $n+1$, to be

$$\bar{J_i}^n = \frac{\sum_{j=1}^{N_i} J_{i,j}}{N_i}, \qquad \bar{J_i}^{n+1} = \frac{\sum_{j=1}^{N_i} J_{i,j} + J_{i,N_i+1}}{N_i+1} = \frac{\sum_{j=1}^{N_i+1} J_{i,j}}{N_i+1}$$

If a member of S_m changes its classification after the next sample, this means that

$$\bar{J_i}^{n+1} > \bar{J}_{i_{m+1}}^{n}, \text{ and } \sum_{j=1}^{N_i+1} J_{i,j} > (N_i+1)\,\bar{J}_{i_{m+1}}^{n}.$$

By definition, $\sum_{j=1}^{N_i} J_{i,j} + J_{i,N_i+1} = \sum_{j=1}^{N_i+1} J_{i,j}$ so

$$\sum_{j=1}^{N_i} J_{i,j} + J_{i,N_i+1} > (N_i+1)\,\bar{J}_{i_{m+1}}^{n}, \text{ and}$$

$$J_{i,N_i+1} > (N_i+1)\,\bar{J}_{i_{m+1}}^{n} - \sum_{j=1}^{N_i} J_{i,j}.$$

Also by definition, $\sum_{j=1}^{N_i} J_{i,j} = N_i \bar{J_i}^n$ so

$$J_{i,N_i+1} > (N_i+1)\,\bar{J}_{i_{m+1}}^{n} - N_i \bar{J_i}^n.$$

Since both means refer to the same iteration, we remove the superscript *n* to get

$$J_{i,N_i+1} > (N_i + 1)\,\bar{J}_{i_{m+1}} - N_i\bar{J}_i.$$

For designs in $S_m{}'$, the same argument can be made to show a change in classification occurs if

$$J_{i,N_i+1} \leq (N_i + 1)\bar{J}_{i_m} - N_i\bar{J}_i.$$

Next, we assess the probability that this occurs given the results of the simulation replications which have occurred up to iteration *n* and the assumption that the performance measure of each design is sampled from a $N(\mu_i, \sigma)$ distribution

$$P\big(J_{i,N_i+1} \geq (N_i + 1)\bar{J}_{i_{m+1}} - N_i\bar{J}_i\big)$$

$$= P\left(\frac{\big(J_{i,N_i+1} - \mu_i\big)}{\sigma} \geq \frac{\big((N_i+1)\bar{J}_{i_{m+1}} - N_i\bar{J}_i\big) - \mu_i}{\sigma}\right)$$

$$= P\left(Z \geq \frac{(N_i+1)\bar{J}_{i_{m+1}} - N_i\bar{J}_i - \mu_i}{\sigma}\right) = P\left(Z \geq \frac{(N_i+1)\bar{J}_{i_{m+1}} - (N_i\bar{J}_i + \mu_i)}{\sigma}\right).$$

using $\bar{J}_i$ to estimate μ_i and s to estimate σ, we get the approximate probability that a design will change its classification from elite to non-elite after the next simulation

$$P\left(Z \geq \frac{(N_i+1)\bar{J}_{i_{m+1}} - (N_i+1)(\bar{J}_i)}{s}\right) = P\left(Z \geq \frac{(N_i+1)\big(\bar{J}_{i_{m+1}} - \bar{J}_i\big)}{s}\right) = P\left(Z \leq \frac{(N_i+1)\big(\bar{J}_i - \bar{J}_{i_{m+1}}\big)}{s}\right).$$

Similarly, designs from $S_m{}'$ have an approximate probability of changing classifications of

$$P\left(Z \leq \frac{(N_i+1)\big(\bar{J}_{i_m} - \bar{J}_i\big)}{s}\right).$$

Depending on the current classification of each design, we calculate the approximate probability that a design will change its classification (APCC) and assign the next simulation replication to the design with the largest of these probabilities according to

$$i^* = \arg\max_i \; p_i = \begin{cases} \Phi\left(\frac{(N_i+1)\big(\bar{J}_i - \bar{J}_{i_{m+1}}\big)}{s}\right) & i \in S_m \\ \Phi\left(\frac{(N_i+1)\big(\bar{J}_{i_m} - \bar{J}_i\big)}{s}\right) & i \notin S_m \end{cases}, \quad N_{i^*} = N_{i^*} + 1 \qquad (2)$$
$$\text{s.t.} \sum_{i=1}^{k} N_i \leq T$$

where estimating the standard deviation, s, is discussed in the next section.

This policy makes intuitive sense as the purpose of the budget allocation in this environment should be to identify the designs which have the least confidence of being classified correctly and verifying their membership or discovering that they were indeed misclassified. If the confidence level of all designs is high, then there is no need to continue the replication of any design, thus signaling a stopping point. In this paper, we do not experiment with stopping criteria and instead use the entire computing budget.

3 IMPLEMENTATION

3.1 Estimating the Mean and Variance

There are two ways that we calculate the variance in this algorithm. The first involves finding a common variance for a set of designs when there is only one replication available per design. The second method involves using as much of the available information as possible to estimate a common variance for a set of designs. When $N_i = 1$ for all designs, the variance is calculated using all of the simulations

$$s^2 = \frac{\sum_{i=1}^{k} \left(J_{i,1} - \bar{J}_{\cdot\cdot}\right)^2}{k-1},$$

where $\bar{J}_{\cdot\cdot}$ is the overall mean of the initial set of simulated values. After this initial case when at least one design has more than 1 simulation, the variance is calculated by pooling the variances of the designs that have more than one replication

$$s^2 = \frac{\sum_{i=1}^{k} {s_i}^2 (N_i - 1)}{\sum_{i=1}^{k} (N_i - 1)},$$

where ${s_i}^2$ is the estimated variance of a design that has more than one simulation replication. Here, the decision is made to use the overall estimate of the variance for each design in the allocation procedure rather than using the individual variance estimates. This decision is based on the fact that the pooled variance estimate is more accurate than the individual estimates if the assumption of equal variances is true.

3.2 OCBAsb Procedure

This new procedure called OCBAsb (OCBA for small budgets) is implemented using a cycle of simulating the appropriate design(s), estimating the mean and variance associated with the designs, and allocating the next simulation replication according to (2) until the number of simulations performed is equal to T. We measure the performance of each procedure by calculating the number of correctly identified elite designs, and by calculating the probability of correct selection (PCS). The OCBAsb procedure is presented in the following table.

```
Algorithm  OCBAsb Procedure
INTIALIZE    Let $n$ equal to 0; Set $N_1{}^n = N_2{}^n = \cdots = N_k{}^n = 1$ and
             $T^n = \sum_{i=1}^{k} N_i{}^n$; Perform $n_0 = 1$ replications for all designs.
WHILE $T^n < T$ DO
    UPDATE      Calculate sample means and sample variance of each design.
    ALLOCATE    Set $N_i{}^{n+1} = N_i{}^n$ for all $i$=1…k.
                Find $i^*$ according to (2). Set $N_{i^*}{}^{n+1} = N_{i^*}{}^{n+1} + 1$
    SIMULATE    Perform replication of design $i^*$. Set $T^n = \sum_{i=1}^{k} N_i{}^n$ and $n = n + 1$
END (WHILE)

OUTPUT       Select the designs after $T$ simulations in the set, $S_m$
```

OCBAsb is tested in three environments. The first is a ranking and selection environment where we are tasked to find a subset of the best designs given a pre-determined number of candidate solutions. The second environment is that of an evolution strategy (ES) where it is well known that equal resampling on a noisy sphere objective function is inefficient. In this well controlled environment, we find the most efficient parameter settings on the noisy sphere using equations from (Arnold and Beyer 2002). We compare an OCBAsb and OCBA-m allocation scheme to determine which method is able to increase the progress rate and discover that it is possible to gain a slight increase in the efficiency over a standard ES. The last environment is that of a simulation optimization where the noise is not fitness proportionate.

4 RESULTS

4.1 Ranking and Selection

We compare OCBAsb in a ranking and selection problem against OCBA-m, OCBA-m+, and equal allocations to see how it performs. Remember that this allocation method is strong when there is less information known about the individual designs (n_0 is small) and when the simulation budget (*T*) is small. The first test is taken out of Chen and Lee (2011) and is a simple generic example. In this experiment, we assume that there are ten design alternatives with normal $N(i, 6^2), i = 1, \ldots, 10$ distributions. We use OCBA-m+, OCBA-m, OCBAsb, and equal allocations to find the top *m*=3 designs. For a total computing budget of T = 70 and T = 80, we search for the method which gives the largest probability of correct selection understanding that there are different options for our initial number of computations. If too many computations are used during the initialization of the algorithm (n_0 is big), then there are fewer simulation replications to allocate in the second stage. If too few computations are used during the initialization (n_0 is small), then it is more likely that many designs are misclassified which also leads to error. Since OCBA-m and OCBA-m+ rely on a good initial approximation for the parameters in the problem, it is only fair to look for possible strategies that favor these algorithms.

Table 1: Probability of correct selection and standard error for 5000 replications for finding the top *m*=3 out of *k*=10 designs. Five different initializations for a computing budget of 70 are compared. The highlighted cells indicate a statistically significant result at a significance level of better than 0.05 compared to the second highest result.

T=70, P{CS}					T=70, Std Error				
n0	OCBAm+	OCBAm	OCBAsb	Equal	n0	OCBAm+	OCBAm	OCBAsb	Equal
1	0.3094	0.3024	**0.4102**	0.2956	1	0.0065	0.0065	0.0070	0.0065
2	0.3652	0.3528	**0.4040**	0.3002	2	0.0068	0.0068	0.0069	0.0065
3	0.3902	0.3686	**0.3982**	0.2948	3	0.0069	0.0068	0.0069	0.0064
4	0.3812	0.3862	**0.4090**	0.2994	4	0.0069	0.0069	0.0070	0.0065
5	0.3782	0.3802	**0.3918**	0.3004	5	0.0069	0.0069	0.0069	0.0065

Table 2: Continuation of Table 1 for a computing budget of 80.

T=80, P{CS}					T=80, Std Error				
n0	OCBAm+	OCBAm	OCBAsb	Equal	n0	OCBAm+	OCBAm	OCBAsb	Equal
1	0.3308	0.3180	**0.4526**	0.3246	1	0.0067	0.0066	0.0070	0.0066
2	0.3894	0.3858	**0.4548**	0.3334	2	0.0069	0.0069	0.0070	0.0067
3	0.4192	0.4022	**0.4458**	0.3210	3	0.0070	0.0069	0.0070	0.0066
4	0.4140	0.4200	**0.4588**	0.3234	4	0.0070	0.0070	0.0070	0.0066
5	0.4180	0.4242	**0.4416**	0.3288	5	0.0070	0.0070	0.0070	0.0066

Table 1 and Table 2 show that for a small computing budget, OCBAsb is a good choice for finding the top-*m* designs out of *k* alternatives. The table also brings up the possibility that OCBAsb may be a good choice to initialize OCBA-m or OCBA-m+ instead of using the current procedure of setting n_0 equal to some arbitrary number prior to gaining any knowledge on the system of designs. Obviously, we need more research to confirm this idea.

Experiment 2 is a continuation of experiment 1 with *m*=3, *k*=10, and the distribution of each design $N(i, 10^2), i = 1, \ldots, 10$. Here, we compare the PCS and the number of correctly identified elite designs in an increased variance setting. Figure 2 summarizes the results after 10,000 iterations. Notice that after ~40 budget allocations, OCBAsb separates from the other procedures and performs better in both measures for the rest of the experiment's computing budget.

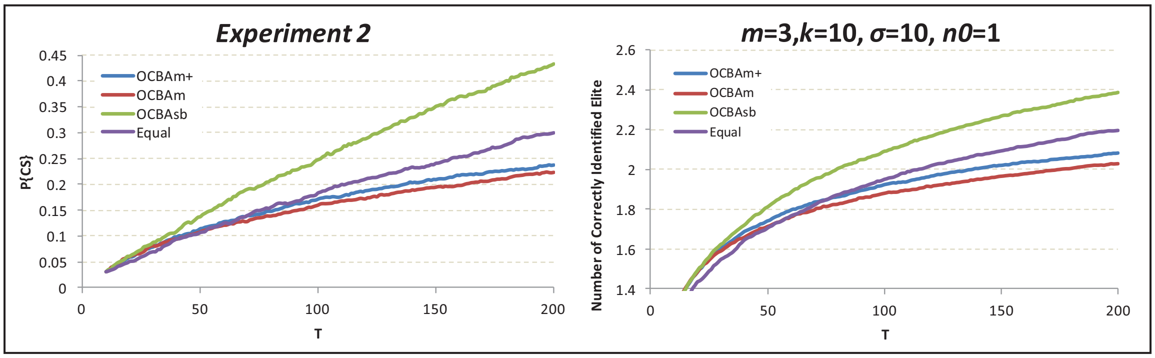

Figure 1: For *m*=3 and *k*=10, P{CS} and the number correctly identified elite are shown for all four techniques.

Experiment 3 is a larger scale version of experiment 2. We increase the number of alternatives to 50 and use the larger variance in order to find the best *m*=10 out of *k*=50 where the designs have normal $N(i, 10^2), i = 1, \ldots, 10$ distributions. Due to the size of this problem, the initial probability of correct selection is small. The propensity of each procedure to grow quickly after only a few replications leads to near linear growth for OCBA-m, OCBA-m+, and equal allocations. The growth potential of the OCBAsb procedure is even greater and is seen as convex in the early budget allocation stage. As noted in experiment 1, we expect OCBAsb to gain an advantage when $n_0 = 1$ so it is not surprising that it outperforms the other procedures in this setting.

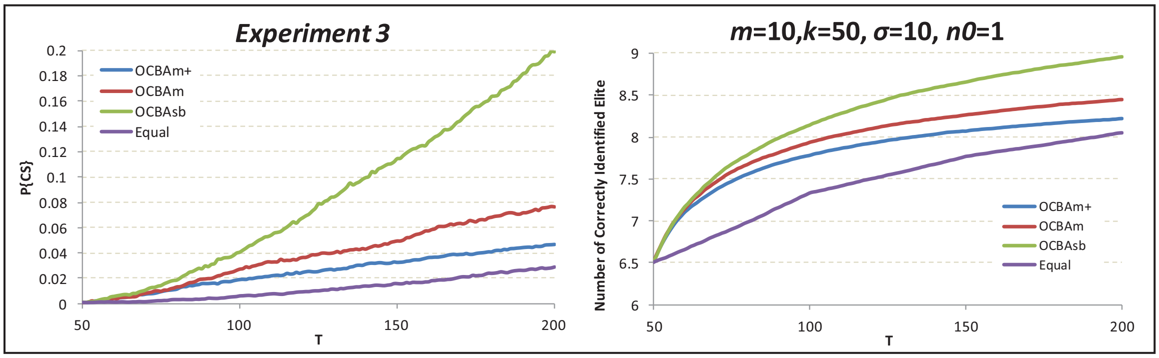

Figure 2: OCBAsb comparison in a larger scaled experiment

4.2 Traditional Evolution Strategy Environment

The purpose of the experiment in this section is to verify the assumption that identifying more of the elite population actually translates to moving closer to the optimum and to determine if it is possible to empirically observe an increase in efficiency of a resampled ES over a traditional ES. Due to the results of Arnold and Beyer (2002) previously mentioned in the introduction, the number of function evaluations, or computing budget, is extremely limited for each generation. Therefore, we compare the budget allocation procedures OCBA-m and OCBAsb in the framework of an ES where the noise in the system is fitness proportionate and equal, the number of designs each generation is λ offspring, the objective each generation is to chose the best μ of λ offspring which have the "best" fitness values, and the comparison measure is the progress rate of the ES.

Parameters which effect the progress rate of an ES include its mutation strength, noise strength, offspring population size and parental population size. In order to select a good evolution strategy, we use the progress rate equations on a noisy sphere from Arnold and Beyer (2002) which rely on normalization of the progress rate, noise strength, and mutation strength. The normalizations are used to make the values independent of the optimization's location and solution space dimensionality. Due to the large number of parameter combinations for evolution strategies, these equations are used to find an ES with which to compare our resample strategy.

We choose ${\sigma_\varepsilon}^* = 16$ to be the normalized noise in the system and the search space dimensionality to be $N = 40$. Using these values, we subsequently choose the following parameters based off (Arnold and Beyer 2002): $\lambda = 76$ from figure 6 and $\mu = 20$ based on the optimal truncation ratio of 0.27. Using the progress rate equation and Mathematica, we find that the optimal mutation strength with the parameters described above to be $\sigma^* = 12.2419$. The ES strategy is a (20/20,76) – ES on the noisy sphere fitness function with normalized noise level of 16 and a normalized mutation strength of 12.2419.

Figure 3 compares the state of the art OCBA-m procedure with the proposed method on a (20/20,76) – ES for 5000 iterations. For this small computing budget and low initial confidence situation, the OCBAsb method does a better job of correctly identifying the top offspring which in turn leads to higher progress rates.

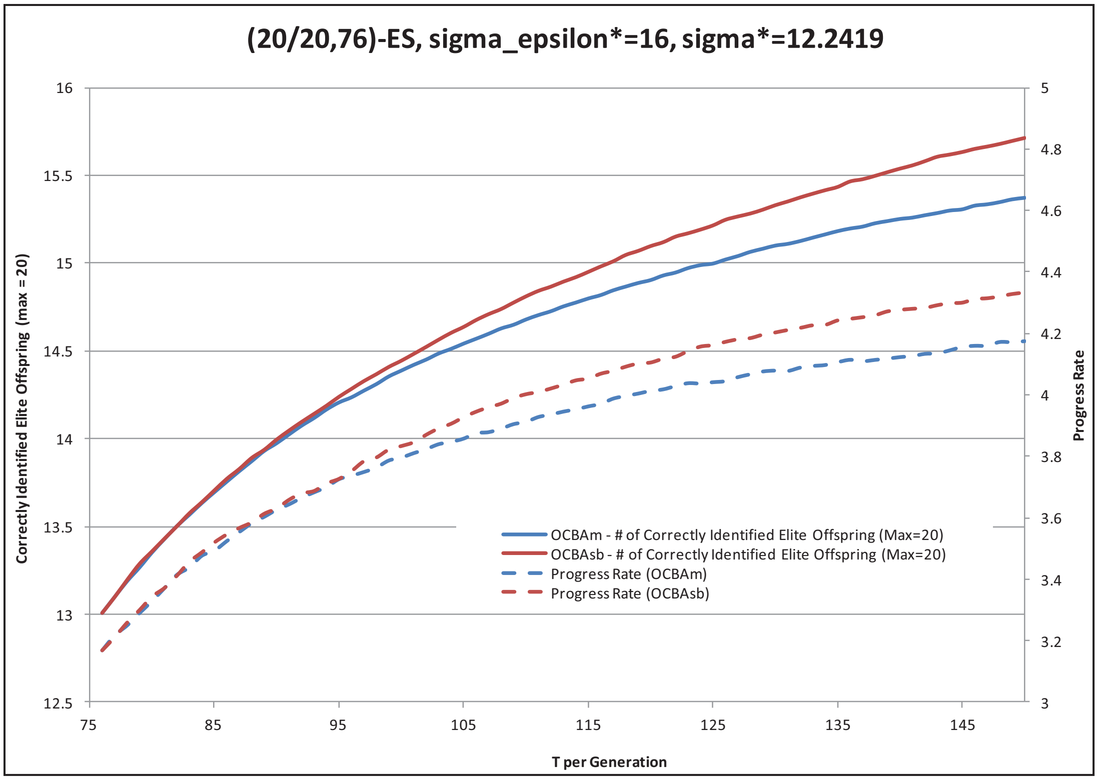

Figure 3: (20/20,76)-ES progress rate and the number of correctly identified offspring for each procedure

Efficiency is measured as the progress rate per function evaluation. Figure 4 shows the difficulty of achieving an efficient resampled evolution strategy compared with the traditional non-resampled approach in this environment. This shows that an increase in progress rate alone is not enough to warrant additional replications of the offspring and that OCBAsb has a slight increase in efficiency for a higher computing budget than OCBA-m.

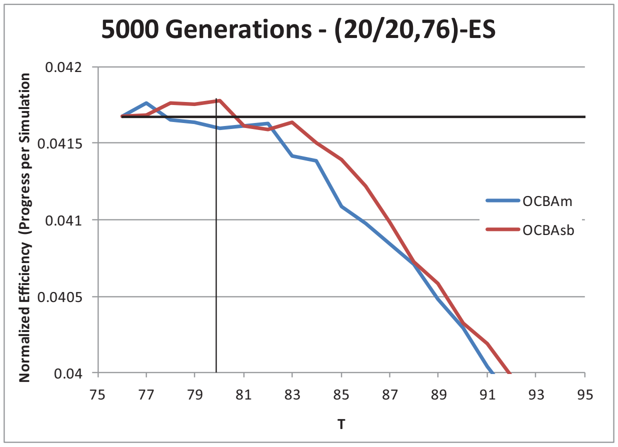

Figure 4: Efficiency comparison after 5000 iterations of resampled ES vs. traditional ES

4.3 Simulation Optimization Evolution Strategy

Noting the difficulty of improving the efficiency of an evolution strategy in an ES environment, we use an ES to direct a simulation optimization in order to efficiently find the optimum for a simulated system. The residual location error is defined in Arnold and Beyer (2002) as the steady state distance from the optimum of an evolution strategy in a problem with constant noise. In the simulation environment which follows, we expect an ES to converge to some steady state location for the given noise surface. The idea is to utilize the power of the ES to quickly get close to this steady state location, and then use resampling and the simulation budget to get as close as possible to the optimal solution given a limited number of model evaluations (simulations). For this test, we use the experimental model from Buchholz and Thummler (2005) where they define the fitness function as a maximization in two dimensional search space of

$$f(\boldsymbol{x}) = 1 - \tfrac{1}{8}\,\boldsymbol{x}^T\boldsymbol{x}.$$

A simulation is defined as a combination of this fitness function and a noise surface to be

$$f_{\text{sim}}(\boldsymbol{x}) = \ f(\boldsymbol{x}) + \ g(\boldsymbol{x}) \cdot \mathrm{N}(0,1),$$

where N(0,1) stands for a standard normal random variate and the noise associated with each simulation is the function $g(\boldsymbol{x})$ which varies depending on the offspring's location as

$$g(\boldsymbol{x}) = \sigma_\varepsilon \left(1 + \tfrac{1}{4}\textstyle\sum_{i=1}^{N} \sin(\pi x_i)\right).$$

where $\sigma_\varepsilon = 0.2$. We use their experimental model, but choose a (13/13,50) – ES with 1 initial replication instead of a (5+5)-ES with 10 initial replications. We adapt the ES mutation strength using a cumulative mutation strength adaption (CMSA) with an initial mutation strength of 1 – see (Arnold 2002) for more details on CMSA. Table 3 shows the distance to the optimal response after 10,000 simulation replications. Since the number of simulations is fixed, the number of ES generations depends on the generational computing budget and is equal to 10,000/*T*. Each generation distributes *T* model evaluations according to its strategy. In the table, equal allocations uses a (13/13,50) - ES and evaluates each offspring T/50 times equally. OCBAsb uses a (13/13,50)-ES, initially evaluating its offspring once and utilizing the algorithm from section 3 to allocate its remaining computing budget (*T*-50). The (13/13,*T*)-ES uses *T* offspring and normal ES procedures to move from generation to generation which traditionally does not include any resampling of its offspring.

Table 3: Distance to the optimal response and standard error for three allocation procedures after 10,000 model evaluations of an ES replicated 100 times. *T* represents the number of simulation replications used each ES generation.

	Optimal Response Difference			**Standard Error**		
T	**Equal Alloc**	**OCBAsb**	**(13/13,*T*)-ES**	**Equal Alloc**	**OCBAsb**	**(13/13,*T*)-ES**
50	0.027173	0.027173	0.027173	0.000211	0.000211	0.000211
100	0.011932	0.011132	0.025980	0.000095	0.000076	0.000118
200	0.007647	0.006719	0.029985	0.000677	0.000534	0.000973
300	0.005821	0.005398	0.033563	0.000427	0.000414	0.001019
500	0.004082	0.003173	0.020733	0.000360	0.000358	0.001186

Table 4: Statistical significance calculation for a computing budget of *T*=500 from Table 3.

2 Sample t-Test for *T*=500				
Procedure	**Avg Diff**	**Var**	**t - Statistic**	**p-value**
OCBAsb	0.003173	6.520E-06	1.78920416	0.03780667
Equal	0.004082	1.299E-05		

From the Table 3, we see that OCBAsb gets consistently (although only slightly) closer to the optimal than equal allocations. The detrimental effects of increasing the number of offspring in the ES is probably due to a sub-optimally tuned initial mutation strength or parental population size, but that is not the focus of this paper. We simply show that combining ES with OCBAsb can be much better than increasing the size of the offspring population if the optimal mutation strength or population size are not known. In Table 4, we see the results of a two sample t-test comparing the equal allocation and OCBAsb optimal response difference for the generational computing budget *T*=500.

5 CONCLUSION

Ideally, we would like to use the computing budget to maximize the approximate probability of correct selection. When the computing budget is small, we are not able to accurately estimate the mean or standard deviation of the designs leading to poor estimates of the APCS. In this environment, it makes sense to allocate the computing budget in a single step look ahead policy similar to KG. By using the limited computing budget to gain confidence on the designs which have the most uncertainty, we are able to identify as many members of the true elite class as possible subject to the constraints of the computing budget. In this paper, we have shown the ability of OCBAsb to use the computing budget to increase the number of correctly identified members of the elite population as well as increasing the PCS in the process for situations where there is a small computing budget. We have furthermore demonstrated that OCBAsb can be (slightly) beneficial in combination with an optimally tuned Evolution Strategy on a noisy sphere, a scenario for which it had been theoretically shown that resampling with equal allocation cannot be beneficial. Finally, we have tested OCBAsb in combination with an Evolution Strategy on a more realistic setting where optimal ES parameters are not known. In this setting, ES with OCBAsb came by an order of magnitude closer to the optimal response than ES alone.

Future work includes changing the assumptions of equal variances for all designs, investigating the possibility of using this process for initializing other OCBA procedures instead of equal allocations, and utilizing a linear loss function which puts more weight on the designs further from the classification boundary. We also intend to investigate the robustness of the OCBAsb with respect to its assumptions.

REFERENCES

Arnold, D. 2002. *Noisy Optimization with Evolution Strategies*. Genetic Algorithms and Evolutionary Computation. Boston: Kluwer Academic Publishers.

Arnold, D., and H. Beyer. 2000. "Efficiency and Mutation Strength Adaptation of the (μ/μI, λ)-ES in a Noisy Environment." In *Parallel Problem Solving from Nature PPSN VI*, 39–48. 2002. "Performance Analysis of Evolution Strategies with Multi-recombination in High-dimensional RN-search Spaces Disturbed by Noise." *Theoretical Computer Science* 289: 629–647.

Beyer, H., and H. Schwefel. 2002. "Evolution strategies–A Comprehensive Introduction." *Natural Computing* 1 (1): 3–52.

Branke, J., S. Chick, and C. Schmidt. 2007. "Selecting a Selection Procedure." *Management Science* 53 (12): 1916–1932.

Buchholz, P., and A. Thummler. 2005. "Enhancing Evolutionary Algorithms with Statistical Selection Procedures for Simulation Optimization." In *Winter Simulation Conference*, 842–852.

Chen, C. H., D. He, and M. Fu. 2008. "Efficient Simulation Budget Allocation for Selecting an Optimal Subset." *INFORMS Journal on Computing* 20 (4): 579–595.

Chen, C.H., and H. L. Lee. 2011. *Stochastic Simulation Optimization*. Singapore: World Scientific.

Chick, S., J. Branke, and C. Schmidt. 2010. "Sequential Sampling to Myopically Maximize the Expected Value of Information." *INFORMS J. on Computing* 22 (1) (January): 71–80.

Frazier, P., W. Powell, and S. Dayanik. 2008. "A Knowledge-Gradient Policy for Sequential Information Collection." *SIAM Journal on Control and Optimization* 47 (5): 2410.

Gupta, S., and Miescke, K. 1996. “Bayesian Look Ahead One-stage Sampling Allocations for Selection of the Best Population.” *Journal of Statistical Planning and Inference* 54 (2): 229–244.

Koenig, Lloyd W., and Averill M. Law. 1985. “A Procedure for Selecting a Subset of Size m Containing the l Best of k Independent Normal Populations, with Applications to Simulation.” *Communications in Statistics - Simulation and Computation* 14 (3): 719–734. doi:10.1080/03610918508812467.

Ryzhov, I., and W. Powell. 2009. “The Knowledge Gradient Algorithm for Online Subset Selection.” In *IEEE Symposium On Adaptive Dynamic Programming and Reinforcement Learning,* , 137–144.

Zhang, S., L.H. Lee, and E.P. Chew. 2012. “A Simulation Budget Allocation Procedure for Enhancing the Efficiency of Optimal Subset Selection.” *Working Paper*.

AUTHOR BIOGRAPHIES

G. JAKE LaPORTE is a graduate student and PhD candidate at George Mason University. His email address is glaporte@gmu.edu.

JUERGEN BRANKE is a Professor of Operational Research and Systems at Warwick Business School. He has published on various topics including handling of uncertainty in optimization, dynamically changing optimization problems, multiobjective optimization, and the design of complex systems. Professor Branke is Associate Editor for the Evolutionary Computation Journal and Area Editor for the Journal of Heuristics. His email address is juergen.branke@wbs.ac.uk.

CHUN-HUNG CHEN is a Professor at National Taiwan University. Dr. Chen served as Co-Editor of the Proceedings of the 2002 Winter Simulation Conference and Program Co-Chair for 2007 INFORMS Simulation Society Workshop. His email address is cchen9@cc.ee.ntu.edu.tw.