Efficient Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters

Volker Nannen*
Institute for Scientific Interchange, Turin &
Vrije Universiteit Amsterdam
volker@cs.vu.nl

A. E. Eiben
Vrije Universiteit Amsterdam
gusz@cs.vu.nl

Abstract— **Calibrating the parameters of an evolutionary algorithm (EA) is a laborious task. The highly stochastic nature of an EA typically leads to a high variance of the measurements. The standard statistical method to reduce variance is measurement replication, i.e., averaging over several test runs with identical parameter settings. The computational cost of measurement replication scales with the variance and is often too high to allow for results of statistical significance. In this paper we study an alternative: the REVAC method for Relevance Estimation and Value Calibration, and we investigate how different levels of measurement replication influence the cost and quality of its calibration results. Two sets of experiments are reported: calibrating a genetic algorithm on standard benchmark problems, and calibrating a complex simulation in evolutionary agent-based economics. We find that measurement replication is not essential to REVAC, which emerges as a strong and efficient alternative to existing statistical methods.**

I. Introduction

One of the big challenges in evolutionary computing is the design and control of evolutionary algorithm (EA) parameters [1]. Part of the problem lies in the fact that most EAs are non-deterministic and highly path-dependent, which makes it difficult to obtain a reliable measure of EA performance on a given problem. On the other hand, performance can be very sensitive to parameters that control for example mutation, and such parameter need a careful and reliable calibration if the EA is to perform well.

The standard statistical method to reduce variance and improve measurement reliability is measurement replication. With measurement replication, a set of parameter values is chosen and the performance (also called response) of the EA with these values is measured several times on the same problem to get a good estimate of the expected response. A classical example of this approach is Analysis of Variance (ANOVA), which provides a clear set of rules how to optimally combine a number of carefully chosen parameter values, how to calculate the number of replications needed to decide whether one combination of values has a significantly better response than another, and how to infer parameter interaction. An exhaustive overview of how to apply ANOVA to EA calibration is given in [2].

There are a number of disadvantages with this approach, particularly when applied to an EA with several sensitive parameters. First, the choice of parameter values is far from trivial and experiments in this vain often allow for no other conclusion than that a given choice was wrong. Second, the variance of an EA can easily be so high and its distribution so bad-behaved that the number of replications needed to produce significant results is not feasible. Third, there is disagreement in the statistical community on how to treat non-numerical results, for example when an EA does not find an acceptable solution within given computational constraints. Fourth, replications divert computational resources that could otherwise be used to obtain a better cover of the search space. This is a serious drawback, since it is virtually impossible to infer from a small number of measurements in a multi-dimensional search space, reliable as they might be, important measures of robustness like sensitivity to small changes and the range of values for which a certain EA performance can be achieved. This problem has been clearly recognized since [3]. As [2] points out, with ANOVA it is difficult to fit anything more sophisticated than a cubic curve to the response curve of an EA.

[4] proposes to use an Estimation of Distribution Algorithm (EDA) for the parameter control of an evolutionary algorithm: REVAC, which stands for Relevance Estimation and Value Calibration. REVAC was designed to a) calibrate the parameters of an EA in a robust way and b) quantify the minimum amount of information that is needed to calibrate each parameter in a robust way. REVAC, like Meta-GA [5], is an evolutionary method, a Meta-EDA, that dynamically explores the complex response surface of an evolutionary algorithm. Starting from a wide distribution over the possible parameter values, REVAC progressively zooms in on the more promising regions of the parameter space, avoiding ANOVA's problem of choosing the correct parameter values right from the beginning. And unlike ANOVA, REVAC only uses rank based statistics to decide where to zoom in. Instead of measuring EA-performance for the same parameter values again and again to increase the reliability of a few estimates, REVAC uses these measurements to get a better cover of the response curve. The resulting distribution is a robust estimate of which regions of the search space give the highest performance. The Shannon entropy of the resulting marginal distributions over each parameter can be used to estimate the relevance of that parameter in an intuitive way.

In [4] we have tested REVAC on calibration benchmarks and have shown that REVAC can indeed calibrate an EA of highly variable performance, and that it can give good

*) Supported by E.C. NEST Contract n. 018474-2 on "Dynamic Analysis of Physiological Networks" (DAPHNet)

estimates of parameter variance and robustness intervals. In [6] we have compared REVAC to Meta-GA and found hat REVAC performance is roughly comparable to that of Meta-GA, if not better. This paper addresses the open question whether REVAC can do away with the computationally expensive replication of measurements, or, conversely, whether measurement replications improve the speed and quality of the REVAC estimate. This question is particularly pressing since REVAC is intended for calibration problems where established methods like ANOVA are inefficient, and where a maximum of information has to be extracted from every available run of the EA. We formulate two research questions:

1) How does the replication of measurements affect the quality of REVAC estimates?
2) How does the replication of measurements affect the computational efficiency of the REVAC search process?

A detailed description of REVAC can be found in Section II. REVAC performance with different levels of measurement replication is studied in Section III: Section III-A tests REVAC on the same calibration problem as used in [2]. Section III-B reports on extensive testing of REVAC as part of ongoing research in evolutionary agent-based economics. A summary and conclusions can be found in Section IV.

Related work includes [7], which estimates response curves for EAs across multiple test cases to measure generalizability. [8] uses a Gaussian correlation function to dynamically build a polynomial regression model of the response curve. Estimation of Distribution Algorithms, in particular those based on univariate marginal distributions, to which the present type belongs, were pioneered in [9]. The relationship between Shannon entropy and EDAs is discussed extensively in [10]. A different approach of using EDA as a meta method to refine a heuristic can be found in [11].

II. Relevance Estimation and Value Calibration

REVAC uses information theory to measure parameter relevance. Instead of estimating the performance of an EA for specific parameter values or ranges of values, REVAC estimates the expected performance when parameter values are chosen from specific probability density distributions. As these density distributions are meant to approximate the maximum entropy distribution for a given level of performance, their Shannon entropy can be used to measure the amount of information needed to reach this level of performance[1].

We define the differential Shannon entropy of a distribution $\mathcal{D}$ over the continuous interval $[a, b]$ as

$$H(\mathcal{D}_{[a,b]}) = -\int_a^b \mathcal{D}(x) \log_2 \mathcal{D}(x)\, dx,$$

with $H(\mathcal{D}_{[0,1]}) = 0$ for the uniform distribution over $[0, 1]$ and negative for any other distribution over $[0, 1]$. The sharper the peaks, the lower the entropy. When a joint distribution $\mathcal{C}$ can be separated into marginal distributions for each parameter, we can take the absolute value of the entropy of each marginal distribution as an indicator of parameter relevance: how much information is needed to calibrate the particular parameter. In these terms the objectives of REVAC can be formulated as follows:

- the entropy of the joint distribution $\mathcal{C}$ is as high as possible for a given level of performance,
- the expected performance of the EA in question is as high as possible for a given level of Shannon entropy.

A. Algorithm Details

Since REVAC is intended for the continuous domain, the choice of suitable EDAs is limited. The present algorithm is a variation of the Univariate Marginal Distribution Algorithm [9]. For efficiency, only a single parameter vector is replaced every generation, and not the whole population.

Given an EA with k parameters REVAC iteratively refines a joint distribution $\mathcal{C}(\vec{x})$ over possible parameter vectors $\vec{x} = \{x^1, \ldots, x^k\}$. Beginning with a uniform distribution $\mathcal{C}^0$ over the initial parameter space $\mathcal{X}$, REVAC gives a higher and higher probability to those regions of $\mathcal{X}$ where the associated EA performs best, increasing the expected performance of the generated EAs. On the other hand, REVAC continuously *smoothes* the distribution $\mathcal{C}$, to reduce the variance of stochastic measurements and to prevent premature convergence. It is the unique combination of these two operators, selection and smoothing, that make REVAC an estimator of the maximum entropy distribution.

For a good understanding of how an EDA works it is helpful to distinguish two views on a set of parameter vectors as shown in Table I. Taking a *horizontal* view on the table, a row is a parameter vector and we can see the table as m of such vectors $X = \{\vec{x_1}, \ldots, \vec{x_m}\}$. Taking the *vertical* view on the columns, each of the k columns shows m values from the domain of the associated parameter i.

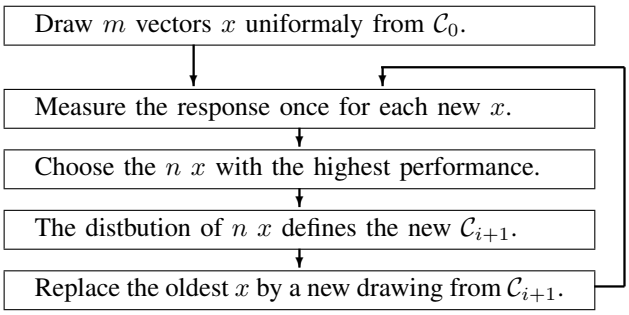
TABLE I
A TABLE X OF m VECTORS OF k PARAMETERS

	$\mathcal{D}(x^1)$	$\cdots$	$\mathcal{D}(x^i)$	$\cdots$	$\mathcal{D}(x^k)$
$\vec{x_1}$	$\{x_1^1$	$\cdots$	x_1^i	$\cdots$	$x_1^k\}$
$\vdots$		$\ddots$			
$\vec{x_j}$	$\{x_j^1$	$\cdots$	x_j^i	$\cdots$	$x_j^k\}$
$\vdots$				$\ddots$	
$\vec{x_m}$	$\{x_m^1$	$\cdots$	x_m^i	$\cdots$	$x_m^k\}$

These m values allow us to define a marginal density function $\mathcal{D}(x^i)$ over the domain of parameter i, scaled to the unit range $[0, 1]$: provided there are no equal values, the m values and the limits can be arranged such that they form $m + 1$ non-overlapping intervals that cover the range $[0, 1]$.

[1]The entropy of a distribution can be understood as the amount of information needed to specify a value drawn from this distribution. The difference between the uniform distribution and a maximum entropy distribution for a given level of performance can be understood as the minimum amount of information needed to achieve that performance.

Fig. 1. Diagram of the update process



We define the density over any such interval $[x_a^i, x_b^i]$ as

$$\mathcal{D}(x^i) = \frac{1}{(m+1)(x_b^i - x_a^i)},$$

satisfying $\int_0^1 \mathcal{D}(x^i) = 1$.

In this way the k columns of Table I define k marginal density functions $\{\mathcal{D}(x^1), \ldots, \mathcal{D}(x^k)\}$ which in turn define a joint density function $\mathcal{C}$. This definition of a density function can be extended to allow intervals to overlap, for example by taking intervals not between first neighbors along the domain of the parameter, but between second or third neighbors. The resulting distribution is a smoothed version of the original one and has a higher Shannon entropy. The more the intervals overlap, the higher the resulting Shannon entropy and Shannon entropy is maximized when all intervals overlap and form a uniform distribution.

As can be seen in Figure 1, REVAC starts from an initial table X_0 that was drawn from the uniform distribution over $\mathcal{X}$. The update process that creates a new table X_{t+1} from a given X_t can be described from both the horizontal and the vertical perspective. Looking at Table I from the *horizontal* perspective we can identify two basic steps:

1) *Evaluating parameter vectors:* Given a parameter vector $\vec{x}$ we can evaluate it: the expected performance of $\vec{x}$ is the performance of the EA executed with these parameter values. The evaluation can be based on one or more replications.
2) *Generating parameter vectors:* Given a set of parameter vectors with known utility we can generate new ones that have higher expected utility.

Step 1 is straightforward, let us only note that we call the performance that an EA achieves on a problem using parameters $\vec{x}$ the *response*. Response r is thus a function $r = f(\vec{x})$; the surface of this function is called a *response surface*. As for step 2, we use a method that is evolutionary itself, (but should not be confused with the EA we are calibrating). We work with a population of m parameter vectors. A new population is created by selecting $n < m$ parent vectors from the current population, recombining and mutating the selected parents to obtain a child vector and replacing one vector of the population.

We use a deterministic choice for parent selection as well as for survivor selection. The best n vectors of the population are selected to become the parents of the new child vector, which always replaces the oldest vector in the population. Only one vector is replaced in every generation. Recombination is performed by a multi-parent crossover operator, uniform scanning, that creates one child from n parents, cf. [12].

The mutation operator—applied to the offspring created by recombination—is rather complicated. It works independently on each parameter i in two steps. First, a mutation interval $[x_a^i, x_b^i]$ is calculated, then a random value is chosen from this interval. To define the mutation interval for mutating a given x_j^i all other values $x_1^i, \ldots, x_n^i$ for this parameter in the selected parents are also taken into account. After sorting them in increasing order, the begin point of the interval can be specified as the h-th lower neighbor of x_j^i, while the end point of the interval is the h-th upper neighbor of x_j^i. The new value is drawn from this interval with a uniform distribution.

From the *vertical* perspective we consider each iteration as constructing k new marginal density functions from the old set $X_t = \{\mathcal{D}_t(x^1), \ldots, \mathcal{D}_t(x^k)\}$. Roughly speaking, new distributions are built on estimates of the response surface that were sampled with previous density functions, each iteration giving a higher probability to regions of the response surface with higher response levels. Each density functions is constructed from n uniform distributions over overlapping intervals. In this context, the rationale behind the complicated mutation operator is that it heavily smoothes the density functions. Like all evolutionary algorithms EDA is susceptible for converging on a local maximum. By consistently smoothing the distribution functions we force it to converge on a maximum that lies on a broad hill, yielding robust solutions with broad confidence intervals. But smoothing does more: it allows REVAC to operate under very noise conditions, it allows it to readjust and relax marginal distributions when parameters are interactive and the response surface has curved ridges, and it maximizes the entropy of the constructed distribution. Smoothing is achieved by taking not the nearest neighbor but the h-th neighbors of x_j^i when defining the mutation interval[2]. Choosing a good value for h is an important aspect when using REVAC. A large h value can slow down convergence to the point of stagnation. A small h value can produce unreliable results. Based on our experience so far, we prefer $h \approx n/10$.

REVAC, like any EDA, is a random process. Not all calibrations achieve results of the same quality. Independently of the fact whether measurement replication is used during the calibration itself, REVAC results can be made more reliable by calibrating an EA more than once, and either choosing the calibration that resulted in a higher performance measure, or averaging over several calibration results. This is indeed a higher level replication of measurement. But unlike measurement replication, which can be too expensive

[2]At the edges of the parameter ranges are no neighbors. We solve this problem by mirroring neighbors and chosen values at the limits, similar to what is done in Fourier transformations.

to extract any useful information, REVAC can always provide a first approximation, which can then be refined if resources permit. Further documentation, a Matlab implementation and graphical demonstrations are available on these web sites:

- http://www.complexity-research.org/revac
- http://www.cs.vu.nl/∼gusz/resources

B. Interpreting the Calibration Results

Because REVAC is implemented as a sequence of distributions with slowly decreasing Shannon entropy we can use the Shannon entropy of these distributions to estimate the minimum amount of information needed to reach a target performance level. We can also measure how this information is distributed over the parameters, resulting in a straightforward measure for parameter relevance. This measure can be used in several ways. First, it can be used to choose between different sets of operators. A set of operators that needs little information to be tuned is more fault tolerant in the implementation, easier to calibrate and robuster against changes to the problem definition. Second, it can be used to identify the critical components of an EA. For this we measure relevance as the absolute difference between the entropy of a distribution and the entropy of the uniform distribution over the same interval, calculating the absolute amount information needed to calibrate the particular parameter. A highly relevant parameter typically has a sharp peak in the distribution, indicating a narrow confidence interval. When an EA needs to be adapted from one problem to another, relevant parameters need the most attention and with this knowledge the practitioner can concentrate on the critical components straight away. Third, it can be used to define confidence intervals for parameter choices. Given a distribution that peaks out in a region of high probability (except for the early stage of the algorithms the marginal distributions have only one peak), we give the 25^{th} and the 75^{th} percentile of the distribution as a confidence interval for the parameter. That is, every value from this range leads to a high expected performance, under the condition that the other parameters are also chosen from their respective confidence interval.

III. Experiments

In order to evaluate the need for measurement replication in REVAC we distinguish 3 layers in the analysis of an EA:

- **Application layer**—the problem(s) to solve.
- **Algorithm layer**—the EA with its parameters operating on objects from the application layer (candidate solutions of the problem to solve).
- **Design layer**—REVAC operating on objects from the algorithm layer (parameters of the EA to calibrate).

In order to study the merits of measurement replication for the design layer we need to define an EA for the algorithm layer and a problem for the application layer. In Section III-A we rely for both purposes on [2], which uses ANOVA to calibrate the mutation and crossover operators in a simple GA of highly variable performance. In Section III-B we report on extensive testing of REVAC as part of ongoing research in evolutionary agent-based economics.

In all experiments reported here REVAC is used with a population of $m = 100$ parameter vectors, from which the best $n = 50$ are selected for being a parent. We smooth by extending the mutation interval over the $h = 5$ upper and lower neighbors. In each experiment REVAC is allowed to evaluate 1,000 parameter vectors. For a test with 3 replications of each parameter vector, this implies 3,000 runs of the EA, evaluating 1,000 different parameter vectors 3 times each.

A. Calibrating a Classic Genetic Algorithm

The EA to calibrate is a generational GA with 22 bits per variable, Gray coding, probabilistic rank-based selection, single point crossover and bit flip mutation. In addition to the two parameters calibrated in [2], mutation $p_m \in [0, 1]$ and crossover $p_c \in [0, 1]$, we also calibrate the population size of $n \in [10, 200]$ chromosomes, a total of 3 parameters. The 4 test functions for the application layer are rather standard: sphere (f_1), saddle (f_2), step (f_3), Schaffer's f_6.

$$f_1(x) = \sum_{i=1}^{3} x_i^2, \quad -5.12 \leq x_i \leq 5.12 \tag{1}$$

$$f_2(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad -2.048 \leq x_i \leq 2.048 \tag{2}$$

$$f_3(x) = \sum_{i=1}^{5} \lfloor x_i \rfloor, \quad -5.12 \leq x_i \leq 5.12 \tag{3}$$

$$f_6(x) = 0.5 + \frac{(\sin\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1 + 0.0001(x_1^2 + x_2^2))^2}, \quad -100 \leq x_i \leq 100 \tag{4}$$

The performance index $\mathcal{F}$ measures the performance of a GA on a specific test function in terms of the computational cost of solving that test function. $\mathcal{F}$ is calculated from the negative (we want to maximize performance) number of generations that are needed to solve the function, times the population size. When a GA needs 100 generations of 100 individuals or 200 generations of 50 individuals, we will say that it has a performance index $\mathcal{F}$ of -10,000. A test function is considered solved as soon as one individual of the population encodes a value that is within certain bounds of the best feasible solution. These bounds are chosen such that a well calibrated algorithm can solve each test function with a performance index $\mathcal{F}$ of between -5,000 and -10,000. If the algorithm doesn't solve the test function with a performance index $\mathcal{F}$ of less than -25,000 (e.g., within 1000 generations if the population size is 25), execution is aborted and a performance index $\mathcal{F}$ of 25,000 is recorded.

We evaluate 5 different levels of replications: 1, 2, 3, 5, and 10 replications. For each level or replication we calibrate each test function 10 times and report the average result. Figure 2 shows the typical time evolution of the median and the 25^{th} and 75^{th} percentile of the distribution of each calibrated parameter. Figure 3 shows the typical time

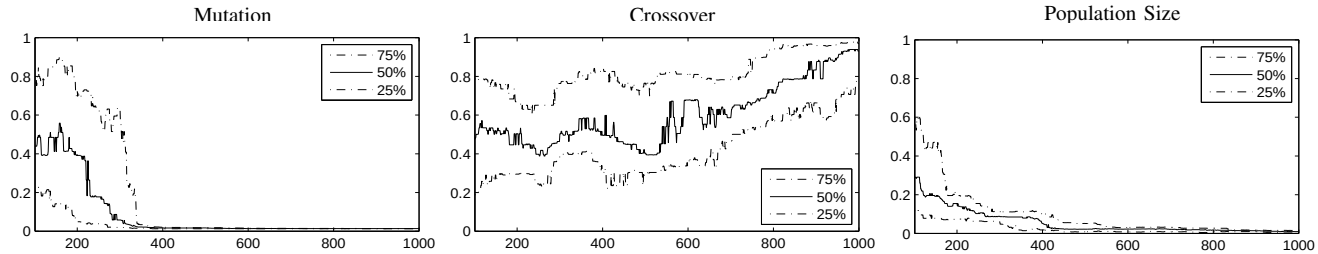


Fig. 2. Typical time evolution of the calibration of the 3 GA parameters. The x-axis shows the progress of the calibration. The y-axis shows the percentiles of the marginal distribution. The central line of each plot shows the evolution of the median of the distribution, the lower and upper lines show the evolution of the 25^{th} and 75^{th} percentile.

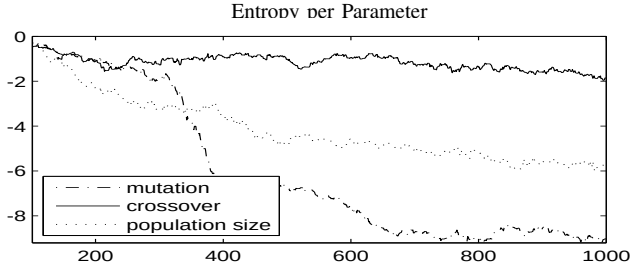


Fig. 3. Typical time evolution of the entropy of the calibrated parameters. The x-axis shows the progress of the calibration. The y-axis shows the entropy for each marginal distribution.

TABLE II

GLOBAL RELEVANCE ESTIMATION OF THE 3 PARAMETERS FOR EACH TEST FUNCTION, BASED ON THE ABSOLUTE ENTROPY MEASURE

	sphere	saddle	step	Schaffer's f_6
mutation	11.1	11.3	10.9	9.6
crossover	1.7	3.5	2.2	0.9
population size	5.9	4.5	6.2	1.0

evolution of the entropy of each marginal distribution during calibration. The test function used in the two figures is the step function and 1 replication is used for REVAC.

Quality of the relevance estimation. To measure the quality of the relevance estimation we need a reliable global estimate with which to compare individual estimations. For this we use the average final estimate of all experiments with all levels of replications, 50 experiments for each test function. This approach is possible because REVAC was designed such that the relevance estimations of different runs of REVAC converge to a unique set of values, the maximum entropy solution.

The quality of the relevance estimation is then measured as the mean error or mean squared distance e from this global estimate. The lower the error e, the better the estimate. The global estimate of parameter relevance, based on the difference in entropy between the uniform distribution and the final calibrated distribution is shown in Table II. We record the following quantities:

- the number of different parameter vectors that REVAC needs to evaluate in order to reach an error $e \leq 0.1$ with regard to the empirical optimum relevance estimation,
- the total number of evaluations that are needed to reach an $e \leq 0.1$ (i.e., the number of parameter vectors times the number of replications),
- the mean squared distance e after evaluating 1,000 vectors, regardless of the total number of evaluations involved, and
- the mean squared distance e after a total number of 1,000 evaluations.

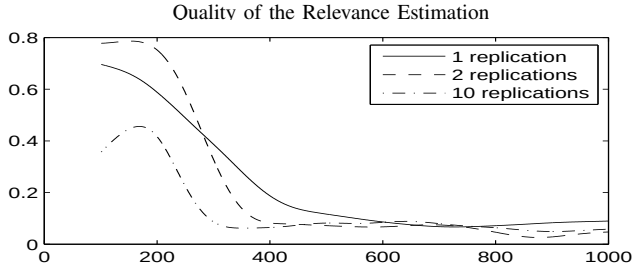


Fig. 4. The y-axis shows the mean squared error with regard to the global relevance estimation, and averaged over all 4 test function. The x-axis shows the process of the calibration, measured in the number of evaluated parameter vectors, not of total evaluations.

TABLE III

QUALITY OF THE RELEVANCE ESTIMATION FOR DIFFERENT LEVELS OF REPLICATION. RESULTS ARE AVERAGED OVER ALL TEST FUNCTION.

number of replications per param. vector	number of parameter vectors at $e \leq 0.1$	number of evaluations at $e \leq 0.1$	error e after 1,000 vectors	error e after 1,000 eval.
1	404	404	0.08	0.09
2	413	826	0.04	0.07
3	741	2,223	0.05	0.23
5	844	4,220	0.04	0.35
10	236	2,360	0.06	0.37

The results in Table III and Figure 4 show that a higher number of replications comes with a heavy computational penalty, but does not improve the quality of the estimation significantly. The final mean squared distance after evaluating 1,000 parameter vectors is higher for the calibration with single replication than for the other cases, but this can simply be due to the fact that with fewer overall evaluations REVAC is still converging on the final value. There is no observable trend in the final results for replication levels of 2–10. We conclude that there is no evidence that replication of measurements leads to a significant improvement of the estimation.

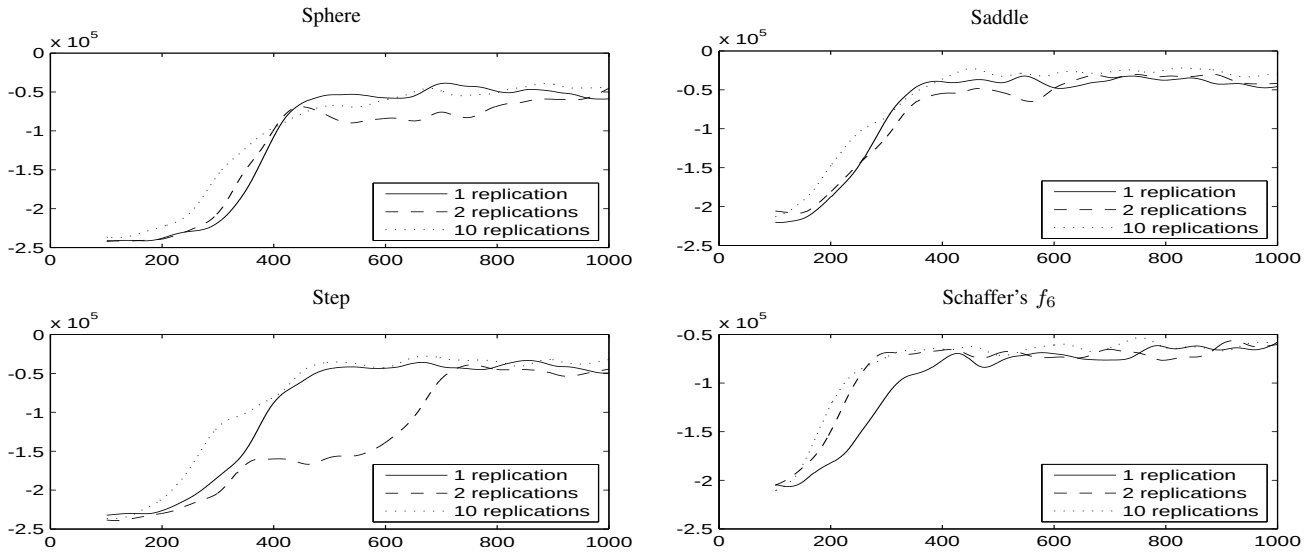


Fig. 5. Time evolution of the average performance of REVAC for three different levels of replications. The x-axis shows the progress of the calibration. The numbers indicate the number of evaluated parameter vectors, not of total evaluations. The y-axis shows the average logarithmic performance of parameter vectors that are chosen according to the current distribution. Graphs are smoothed for readability.

TABLE IV

GLOBAL OPTIMUM PERFORMANCE ESTIMATION FOR EACH TEST FUNCTION.

	Sphere	Saddle	Step	Schaffer's f_6
optimum performance index $\mathcal{F}$	-3,786	-2,770	-2,107	-3,260

Quality of the value calibration. To measure the quality of the value calibration we use the average performance f_t for each test function t of the globally best calibrations. That is, of all calibrations for a test function t with different levels of replication we choose the performance of the best calibration as the global measure. This allows us to evaluate the results obtained with different levels of replication without knowing if a better calibration is possible. The performance of the globally best are shown in Table IV. We again consider 4 cases:

- the number of parameter vectors needed to achieve an average performance index of at least twice the best performance index f_t (i.e., -10,000 if $f_t = -5,000$),
- the number of evaluations needed to achieve the same performance (which is equal to the number of parameter vectors times the number of replications),
- the average performance index reached after evaluating 1,000 parameter vectors, regardless of the number of evaluations involved, and
- the average performance index reached after 1,000 evaluations.

Table V and Figure 5 show the results. Performance levels are almost independent from the level of replication, depending almost entirely on the number of parameter vectors that REVAC has evaluated so far. Note especially how the performance comes close to the maximum around parameter vector 400, independent of the level of replication. Replication does also not improve the absolute capability of REVAC to reach a better calibration. The performance penalty however is huge. The amount of computation needed to reach an arbitrary level of performance increases almost linearly with the level of replication.

TABLE V

QUALITY OF THE CALIBRATION WITH DIFFERENT LEVELS OF REPLICATION. RESULTS WERE FIRST CALCULATED FOR EACH TEST FUNCTION t, USING THE PERFORMANCE f_t OF THE OPTIMUM CALIBRATION ON THAT FUNCTION, AND THEN AVERAGED OVER ALL TEST FUNCTIONS.

number of replications per param. vector	number of parameter vectors at $\mathcal{F} \geq 2f_t$	number of evaluations at $\mathcal{F} \geq 2f_t$	perform. index $\mathcal{F}$ at 1,000 vectors	perform. index $\mathcal{F}$ at 1,000 evalations
1	411	411	-9,954	-9,789
2	397	795	-6,326	-7,250
3	241	722	-4,783	-4,877
5	380	1,901	-10,576	-10,424
10	277	2,772	-9,006	-9,072

B. Calibrating an Evolutionary Agent-Based Simulation

Here we report on extensive testing of REVAC as part of ongoing research in evolutionary agent-based economics [13]. To describe the experimental setup in a nutshell: 200 agents evolve their investment strategies over a period of 500 time intervals. In each interval each agent invests its current income in a number of economic sectors. The agent's income of the next interval is then calculated according to some production function. The production function changes dynamically, so that the same investment strategy will lead to different growth rates at different points in time. Agents adapt their investment strategies through random mutation

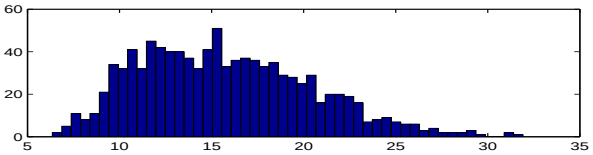


Fig. 6. Histogram of the EA performance, based on 1000 runs.

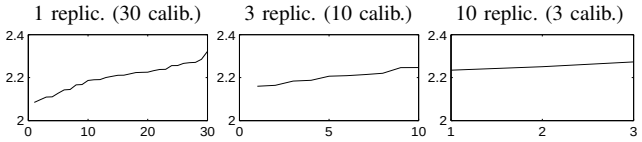


Fig. 7. Distribution of calibration results. The y-axis shows the average performance at the end of the calibration. The x-axes shows the id of the calibration, sorted by performance. Note the similar median.

and selective imitation in a complex social network. Mutation here is a random change to the way the investment is distributed over the economic sectors. For imitation an agent compares its own economic growth rate to that of its peers in the social network. If a peer has a higher growth rate it can copy that peer's strategy, wholly or in part.

The performance measure that REVAC has to maximize is the mean log income of the economic agents at the end of a simulation, corresponding to the preference of an economic agent with constant relative risk aversion. Figure 6 shows a typical histogram of the performance measure, based on 1000 runs with identical calibrated parameter settings. The distribution is skewed and has a flat tail, limiting the value of measurement replication. The distribution is not lognormal, but the estimated mean of the logarithmic performance seems to converge faster than the estimated mean of the performance itself and is a more reliable statistic. For this reason we average over the logarithm of the performance measure when we report the performance reached by different REVAC calibrations, even though the calibration is done in the original domain.

The algorithm layer has 6 parameters that need calibration: *mutation probability*, *mutation variance*, *imitation probability*, *imitation ratio* (how much of the original strategy is perserved), *imitated fraction* (the fraction of well performing peers that are considered for imitation), and the *connectivity* of the social network. For the application layer we consider four different dynamic economic environments: changes occur sudden and with high frequency (*sudden-high*), sudden and with low frequency (*sudden-low*), gradual and with high frequency (*gradual-high*), and gradual and with low frequency (*gradual-low*).

We use REVAC with one, three and ten replications of measurements to calibrate the algorithm layer to each of the four economic environments. All other REVAC parameters are as described before. To improve the reliability of the calibration, we also look into the option of repeating each calibration several times, choosing those calibrations that achieved the highest performance and averaging over their calibration results. Due to limited computational resources we used different numbers of calibration for each replication scheme: 30 for 1 replication, 10 for 3 replications and 3 for 10 replications.

Figure 7 shows the average (log) performance that each calibration achieved during the last 10% of its evaluations. Results are sorted per replication scheme to show how the calibration results vary. With only 3 calibrations for the 10 replication scheme no firm conclusion is possible, but a general trend is visible: the distribution of calibration results is similar for all numbers of replication, with similar mean and variance. The same can be observed for each relevance estimation and each calibrated parameter value: all calibration results follow a similar distribution, regardless of the number of measurement replications.

Since not all calibrations achieve the same performance measure, we decide to take only the better 50% and average over the result. To compare REVAC with 1 measurement replication and with 3 measurement calibrations we start by randomly selecting 10 out of the 30 calibrations with 1 replication. Of these we take the better 5 and compare their average results to those of the better 5 from the implementation with 3 replications. From the implementation with 10 replication we only use the better 2 calibrations. Table VI shows the average relevance estimation (in absolute entropy) for every parameter, and the suggested value for each parameter (the median of the distribution) for one economic environment (sudden-low). Average calibrated values for each parameter are shown in bold, followed by the measured variance. Note how the measured variances for the different measurement replications are all of the same order.

TABLE VI

CALIBRATED PARAMETERS. AVERAGE VALUES IN BOLD, FOLLOWED BY THE MEASURED VARIANCE

relevance estimation (absolute entropy)

	1 replication		3 replications		10 replications	
mutation probability	**0.6**	0.4	**0.5**	0.2	**0.3**	0.2
mutation variance	**0.3**	0.1	**0.2**	0.0	**1.2**	0.3
imitation probability	**0.9**	0.3	**1.2**	0.3	**1.1**	0.4
imitation ratio	**1.2**	0.5	**1.8**	1.0	**1.8**	1.0
imitation fraction	**0.8**	0.4	**0.7**	0.1	**0.9**	0.5
connectivity	**0.1**	0.0	**0.2**	0.1	**0.0**	0.0
all parameters	**3.9**	1.0	**4.6**	1.7	**5.3**	0.3

suggested parameter values

	1 replication		3 replications		10 replications	
mutation probability	**0.20**	0.03	**0.21**	0.02	**0.45**	0.09
mutation variance	**0.28**	0.02	**0.30**	0.03	**0.15**	0.01
imitation probability	**0.83**	0.01	**0.86**	0.00	**0.85**	0.01
imitation ratio	**0.90**	0.00	**0.93**	0.00	**0.93**	0.00
imitation fraction	**0.85**	0.01	**0.77**	0.01	**0.83**	0.01
connectivity	**0.54**	0.04	**0.58**	0.05	**0.51**	0.01

To see if each REVAC implementation correctly differentiates between different problems in the application layer we apply each of the four calibration a thousand time to each economic environment and average over the logarithm of the performance measure. This is done separately for

each replication scheme. Table VII shows the results. Each row stands for one economic environment and has four entries, showing the results when applying its own calibration and the other three calibrations to that environment. The bold values show the highest value for each row. With correct differentiation we expect to see the highest value for each economic environment when parameters from its own calibration are used. As can be seen, this is almost always the case. The variance of the measured means is below 0.001 and therefore insignificant.

In general one can conclude that there is no significant difference in results obtained with 1, 3, or 10 measurement replication, even though in the case of 1 replication the total number of evaluations is significantly smaller. With the exception of one environment, all calibrations differentiate well between different problems in the application layers. One of the design goals of REVAC is to provide robust calibration that work well on similar problems. And indeed, all calibrations achieve good results on all economic environments. To compare, an uncalibrated system has a mean logarithmic performance of between 1.7 and 2, depending on the economic environment.

IV. Summary and Conclusion

Measurement replication is the standard statistical method to reduce variance, but computationally expensive when the variance is high. We studied the benefit of measurement replication when applied to REVAC. In the first set of experiments three parameters of a generational GA of variable performance were calibrated on four different test functions. We studied the quality and the computational cost of the obtained relevance estimation and value calibration as a function of the number of measurement replications.

In the second set of experiments we used REVAC to calibrate a complex evolutionary economic simulation. 6 parameters needed calibration for 4 different environments. The performance measure followed a skewed distribution with a flat tail. We found that calibration results for different levels of measurement replication follow a similar distribution, with a variance of the same order of magnitude. Increasing the number of measurement replications does not reduce the variance of the final results. We also found that average results from more than one calibration are fairly reliable and that they can differentiate well between different problems on the application level.

We conclude that in spite of the heavy computational penalty, more than one replication per measurement does not lead to a significant quality improvement of the relevance estimation and value calibration of REVAC. Using REVAC with only a single replication per parameter vector is sufficient to calibrate an evolutionary algorithm and to estimate the relevance of each parameter. Most of the information that REVAC can give on an EA will be produced by using only a single measurement. If resources permit more evaluations, it is more advisable to run REVAC several times to increase the robustness of the results.

TABLE VII

Performance of Calibrated Simulation. Each row are four calibrations applied to the same dynamic environment.

1 measurement replication, 10 calibrations

	grad.-low	grad.-high	sud.-low	sud.-high
gradual-low	**2.628**	2.619	2.603	2.582
gradual-high	2.269	**2.539**	2.524	2.511
sudden-low	2.686	2.713	2.724	**2.727**
sudden-high	2.089	2.233	2.226	**2.256**

3 measurement replications, 10 calibrations

	grad.-low	grad.-high	sud.-low	sud.-high
gradual-low	**2.610**	2.591	2.597	2.584
gradual-high	2.375	**2.531**	2.520	2.512
sudden-low	2.710	2.716	**2.733**	2.704
sudden-high	2.102	2.247	2.230	**2.258**

10 measurement replications, 3 calibrations

	grad.-low	grad.-high	sud.-low	sud.-high
gradual-low	**2.625**	2.589	2.595	2.577
gradual-high	2.202	**2.540**	2.521	2.502
sudden-low	2.691	2.712	2.710	**2.713**
sudden-high	2.024	2.243	2.202	**2.261**

References

[1] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 124–141, 1999.

[2] A. Czarn, C. MacNish, K. Vijayan, B. Turlach, and R. Gupta, "Statistical Exploratory Analysis of Genetic Algorithms," *IEEE Trans. Evol. Comput.*, vol. 8, no. 4, pp. 405–421, 2004.

[3] G. Taguchi and Y. Wu, *Introdution to Off-Line Quality Control*. Nagoya, Japan: Central Japan Quality Control Association, 1980.

[4] V. Nannen and A. E. Eiben, "Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters," in *Proc. Int. Jt. Conf. Artif. Int., IJCA'07*, 2007, pp. 975–980.

[5] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. Syst. Man and Cybernet*, vol. 16, no. 1, pp. 122–128, 1986.

[6] W. A. de Landgraaf, A. E. Eiben, and V. Nannen, "Parameter calibration using meta-algorithms," in *IEEE Congr. Evol. Comput., CEC'07*, 2007, pp. 71–78.

[7] O. François and C. Lavergne, "Design of evolutionary algorithms-a statistical perspective," *IEEE Trans. Evol. Comput.*, vol. 5, no. 2, pp. 129–148, 2001.

[8] T. Bartz-Beielstein, C. W. G. Lasarczyk, and M. Preuss, "Sequential parameter optimization," in *IEEE Congr. Evol. Comput., CEC'05*, 2005, pp. 773–780.

[9] H. Mühlenbein, "The equation for response to selection and its use for prediction," *Evol. Comput.*, vol. 5, no. 3, pp. 303–346, 1997.

[10] H. Mühlenbein and R. Höns, "The estimation of distributions and the minimum relative entropy principle," *Evol. Comput.*, vol. 13, no. 1, pp. 1–27, 2005.

[11] Q. Zhang, J. Sun, G. Xiao, and E. Tsang, "Evolutionary Algorithms Refining a Heuristic: A Hybrid Method for Shared-Path Protections in WDM Networks Under SRLG Constraints," *IEEE Trans. Syst. Man and Cybernet. B*, vol. 37, no. 1, pp. 51–61, 2007.

[12] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Berlin / Heidelberg: Springer, 2003.

[13] V. Nannen, "Evolutionary Agent-Based Policy Analysis in Dynamic Environments," Ph.D. dissertation, Artificial Intelligence, Vrije Universiteit, Amsterdam, Amsterdam, 2009.