Simulation in Production
and Logistics 2015
Markus Rabe & Uwe Clausen (eds.)
Fraunhofer IRB Verlag, Stuttgart 2015

Reduction of Simulation Effort for Simulation-based Optimisation

Reduzierung des Simulationsaufwandes bei simulationsbasierter Optimierung

Tobias Uhlig, Oliver Rose, Universität der Bundeswehr München, Neubiberg (Germany), tobias.uhlig@unibw.de, oliver.rose@unibw.de

Abstract: Simulation often represents a bottleneck with regard to the applicability of simulation-based optimisation. Generally, simulation-based optimisation requires a sufficient amount of simulation runs to operate reasonably well. Since every simulation run is a valuable resource, it is important to use it efficiently. This is especially true for stochastic simulation, where additional runs are needed to provide statistical significance for the evaluation of candidate solutions. This paper discusses three approaches that aim to reduce the amount of required simulation runs for simulation-based optimisation. The approaches are compared to the classic technique, which relies on a fixed number simulation runs to evaluate a candidate solution. Our experiments show a significant gain in efficiency for the new approaches.

1 Introduction

Simulation-based optimisation (SBO) is an established optimisation approach relying on simulation as a tool to evaluate candidate solutions. In recent years its popularity has increased significantly (Hagendorf 2010). This can at least partially be attributed to the increased availability of computing power. While the optimisation component usually is quite undemanding, the simulation oftentimes requires a lot of computing power. Unfortunately, an effective SBO approach usually needs a large amount of simulation calls to evaluate candidate solutions. This effect is amplified whenever stochastic simulation models are considered, since repeated simulation runs are required for one model to guarantee statistical significance. Generally, the number of simulation runs N reduces the corresponding standard deviation by a factor of square root of N (Law and Kelton 1999). A common practice is to use ten repetitions to evaluate a simulation model. A reduced simulation effort or faster simulation can be attained in many different ways, e.g., the use of more efficient simulation procedures, the use of more abstract simulation models, or the use of distributed parallel simulation. Our goal is to reduce the actual number of simulation runs, without causing a negative impact on the returned quality. In other words, we want to use the available simulation runs more

efficiently. We take general inspiration from the field of metaheuristics. Basically, SBO can be considered a (meta)heuristic that uses simulation to evaluate candidate solutions (see Fig. 1). In fact, simulation can be seen as the basis for a fitness function that asserts the quality of solutions generated by the optimiser. Previous research on metaheuristics has already considered the challenge of noisy fitness functions. On the whole, three classes of approaches were proposed to handle uncertain fitness functions, i.e., explicit averaging, implicit averaging, and modification of selection (Jin and Bratke 2005). With regard to SBO the work of Chen (2003) demonstrated an efficient sampling approach that lead to a significant reduction of simulation runs.

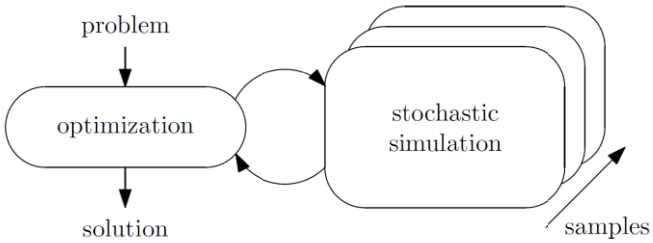

Figure 1: *Basic concept of simulation based optimisation.*

For our paper, we rely on the idea that most selection processes in SBO can be broken down to a comparison of two candidate solutions. With this in mind, we want to compare four approaches to handle uncertainty in SBO.

This paper has the following structure. After this introduction, the next section discusses the optimisation approach used for our experiments. Subsections 2.1 to 2.4 introduce the techniques used to compare potential candidate solutions using simulation. Afterwards, we elaborate our experimental setup (see Section 3) and present the results of our experiments in Section 4. We close with a discussion (see Section 5) and summary (see Section 6) of the content provided by the paper.

2 The Optimisation Approach

This section discusses different optimisation approaches used in conjunction with simulation. In detail, we will focus on the decision processes used in optimisation to identify suitable solution candidates. Generally, we use a classic evolutionary algorithm for optimisation. For selection purposes, we evaluate four approaches to compare two candidate solutions:

- The classic comparison based on averaging the results of a fixed number of simulation runs,
- a phase-based approach that adapts the number of simulation runs used for a comparison as the optimisation progresses,
- a statistical approach, that uses Student's t-test to determine the number of simulation runs used for a comparison on a case by case basis,
- and a novel "Natural" approach.

These approaches rely on two or more simulation runs to assess the relative fitness of two candidates. For each candidate solution we use at least one simulation run to

evaluate it. Additional runs may be used to increase the statistical significance of the evaluation. The results of the evaluation are used by the evolutionary algorithm to select promising candidate solution for further optimisation. Considering, the evolutionary algorithm we recognise three aspects of the optimisation, which require simulation runs:

1. We already mentioned the necessity to use more than one simulation run to evaluate a candidate solution. This explicit *sampling* is required to provide statistical significance. With each additional simulation run we increase the reliability of the observed fitness of a candidate solution. However, eventually adding more simulation runs results in diminishing returns.
2. Generally, evolutionary algorithms operate by accumulating small changes to generate a good result over many *generations*. Accordingly, the number optimisation steps must be large enough to support this process. To this end, we need to allocate a large amount of simulation runs to the evaluation of successive candidate solutions. Once again, we observe diminishing returns when to many simulation runs are used for the generations, since the optimisation process eventually converges. At the point of convergence, the process has reached a local optimum and further optimisation will not result in an improvement.
3. Finally, we can assign simulation runs to consider more than one candidate solution in parallel – a *population* of potential solutions. Basically, a large population increases the robustness of an optimisation process. It can prevent premature convergence to a single local optimum by considering alternative solutions. Additionally, a larger population facilitates implicit sampling and thereby improves the performance of the optimisation in the context of uncertain evaluation functions like stochastic simulation (Arnold and Beyer 2003).

Fundamentally, all three aspects – sample size, generations, and population size – benefit from additional simulation runs. To realise an efficient optimisation it is necessary to balance their demands to match the number of available simulation runs. The following subsections discuss four strategies to distribute the available runs to the three aspects.

2.1 The Classic Approach Using Fixed Sample Sizes

The classic approach for SBO relies on fixed parameter assignments. To this end, the experimenter defines a parameter setup in advance to the optimisation. During the optimisation process these parameters remain unchanged. For our scenario, using SBO with an evolutionary algorithm, the parameters are: number of generations, population size, and the number of simulation runs used to sample one candidate solution. These values are often assigned based on intuition or experimentation. There are no exact guidelines but rather rules of thumb for their setup. Considering, for example the sample size used to provide statistical significance a value of ten is often used. Employing ten simulation runs for the evaluation is regarded as a reasonable compromise between calculation time and accuracy. Subsection 4.1 illustrates the relative influence of the three parameters based on experimental evidence.

2.2 Phase-based Approach

The phase based approach was proposed in Frank et al. (2013). It divides the optimisation into three phases: exploration, exploitation and accurate simulation. Based on the assumption that during the initial exploration phase candidates solutions are more varied it uses only a small amount of simulation runs in the beginning. As the optimisation progresses the algorithm switches to the exploitation phase and eventually to accurate simulation phase and uses more simulation runs. More simulation runs are required later on, since it is more difficult to distinguish candidates the closer we get to the optimum. The algorithm decides when to switch to the next phase based on the observation of the optimisation progress. It monitors the progress of the fitness and calculates the slope of the fitness curve. Whenever the algorithm reaches a predefined slope it switches to the next phase. Equation 1 defines the calculation of slope (m) for each generation (g) using an offset (o). The offset is used to avoid erratic switching caused by short-term stagnation of the optimisation. Using the offset the algorithm only reacts to longer-term changes over multiple generations.

$$m = \frac{fitness_{g-o} - fitness_g}{d} \tag{1}$$

2.3 Statistical Approach Using Students T-Test

The statistical approach is discussed in more detail in Uhlig and Rose (2015). The basic idea is to start with a small sample size of simulation runs to get an initial evaluation of two candidate solutions. Subsequently a paired Student's t-test is applied to the samples. If the test indicates a significant difference between the samples a comparison of the sample averages is returned. Whenever the test is inconclusive additional simulation runs are performed. Essentially, this approach reduces the number of simulation runs for cases were candidate solutions are sufficiently different and only uses more runs whenever the candidates have similar quality. Algorithm 1 summarises the employed technique.

Algorithm 1: *The statistical approach using Student's t-test.*

```
1.   For every model execute 3 simulation runs and
     evaluate the objective value for each run.
2.   Use the t-test for the samples of objective values
     using the null hypothesis (H0) that the two samples
     have equal means. We calculate the minimal
     significance level (p) at which we can reject the H0.
3.   If p > 0.5 and the maximum number of simulation runs
     (20 per model) is not reached go to step 4 otherwise
     go to step 5.
4.   Execute one more simulation run and add the result to
     the appropriate sample. Return to step 3.
5.   Compare the two models using the sample average of the
     respective objective values.
```

2.4 "Natural" Approach

The final approach we will discuss is inspired by biological evolution. Accordingly, we refer to it as the "Natural" Approach. Conceptually, it is very simple and requires in its current implementation no additional parameters. It relies on implicit sampling, but in contrast to simply using a large population it does so over time. This is achieved by limiting the use of simulation results to just one generation. During each generation candidate solutions are evaluated based on only one simulation run. Subsequently, the simulation results are discarded at the end of each generation and for the next generation the candidates are re-evaluated. This mimics selection processes of natural evolution. In nature, an organism does not have an assigned fixed fitness value, but rather has to repeatedly prove to be fit enough for survival and procreation. Similarly, a candidate solution in the "Natural" approach is compared repeatedly with other candidates. Only candidates that are successful consistently over multiple generations will "survive" the optimisation process. In contrast, statistical anomalies in the simulation will only let a candidate survive a small amount of generations.

3 Experimental Setup

To test and compare the proposed approaches we perform a series of experiments. Our experimental setup relies on the SEREIN (Uhlig 2013) presented in Uhlig (2015). The framework provides implementations of various evolutionary algorithms and supports a comparison-based fitness evaluation. With regard to the implementation of our "Natural" approach, it is important that the framework has a distinct fitness concept. This concept models fitness as an observation by a selecting agent rather than being an inherent feature of a candidate solution. This approach lets the selecting agent handle all fitness decisions and accordingly it can easily reset the values for each generation.

The complete setup we use is illustrated in figure 2. It shows the four implementations of our comparison approaches depicted in blue and the interfaces and classes provided by the SEREIN framework in white. Depicted in orange are classes used to model a simulation component of SBO. We do not use an actual simulation for our experiments, since the amount of computation time required to perform all experiments is too large. Instead of an actual simulation we use a benchmark function with comparable behaviour and complexity. This technique has already been used successfully in Frank et al. (2013). We use the deterministic Rastrigin function (Mühlbein et al. 1991) and add normal distributed noise to it. In addition to the reduced computation time, this approach has two advantages. On the one hand, we know the theoretical optimum of the fitness function, which usually is not attainable for an actual simulation. On the other hand, the added error term in conjunction with a deterministic function provides fine grained control of the observed uncertainty.

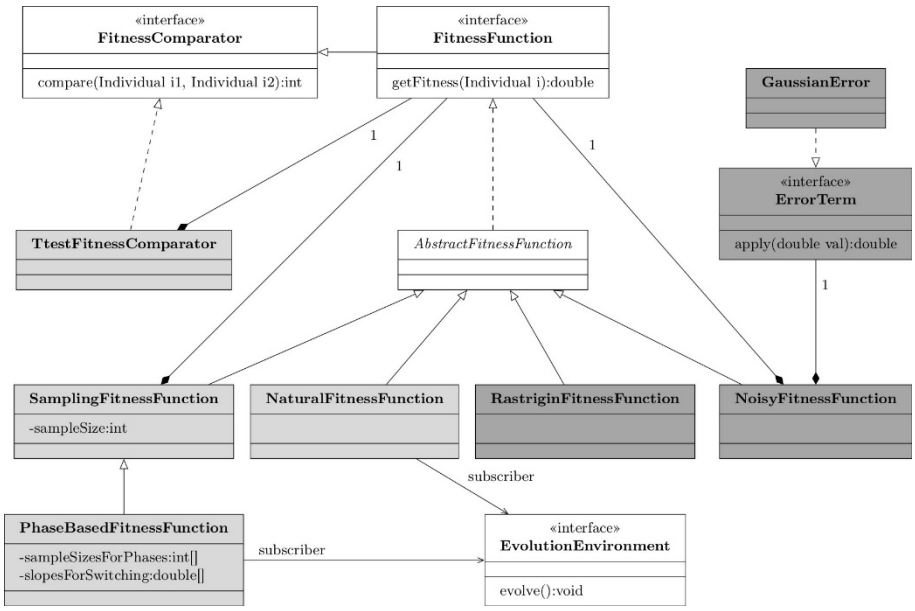

Figure 2: *UML class diagram for our experiment setup. Depicted white, classes provided by the SEREIN framework; in light gray, our implementations for evaluation methods that can be used in SBO; and depicted in dark gray, classes that are used to model a benchmark problem that mimics a simulation.*

For our experimental design, we use a standard evolutionary algorithm. Most parameters for this algorithm were determined using a meta-optimisation, e.g., mutation and selection operators. For our experiments, we mainly consider the three previously discussed parameters: number of generations, population size, and number of sample simulation runs see. We considered the following settings:

- Population size – 1 to 100
- Sample size – 1 to 20 simulation runs (where applicable)
- Generations – Implicitly given, each optimisation terminated after 36,000 simulation runs
- Error term – $N(0, \sigma^2)$, $\sigma = 0...10$

Each experiment is repeated 100 times to ensure statistical significance. In the following section we discuss the results of our experiments.

4 Experimental Results

The results of our experiments are divided into two parts. For the first part, we analyse the general impact of the three parameters – generations, population, and sample size – on the optimisation process. We evaluate them given different magnitudes of uncertainty using the classical approach with fixed parameter settings. We use the results to illustrate influence of these parameters and to determine a

reasonable reference setup. This reference is used in the second part of the experiments to compare the other approaches to the classic setup.

4.1 Influence of Parameters

Figures 3 and 4 exemplify the results of our initial experiments. Generally, a higher amount of noise causes the performance of SBO to decrease. Increasing the population size (fig. 3, on the left) or the number of samples reduces the performance decline. However, both values have to be balanced reasonably since too large values for both of them subtracts from the number of available optimisation steps (fig. 4).

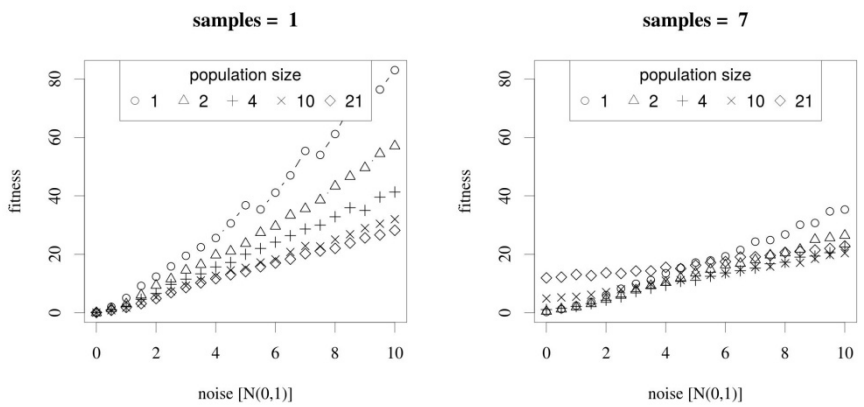

Figure 3: *Influence of population size and the magnitude of the error on SBO*

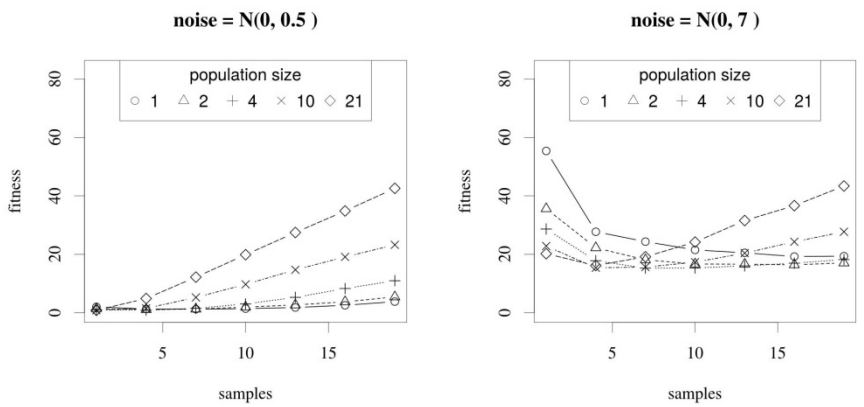

Figure 4: *Influence of sample size and population size on SBO.*

4.2 Comparison of Approaches

For our final comparison, we use the optimal setups determined in the initial experiments. The four compared approaches are:

- The classic approach implemented by the SamplingFitnessFunction (SFF)
- The phase-based approach implemented in the PhaseBasedFitnessFunction(PBF)
- The "Natural" approach (Natural)
- The statistical approach implemented in the TtestFitnessComparsion (TFC)

The results of our experiments are shown in figure 5. All alternative approaches exceeded the performance of the reference. However, the phase based approach shows only minimal improvements. Considering the added amount of parameterisation required (sample sizes for each phase and switching parameters) we do not recommend its use over a well set up classic approach. The Natural approach returns good results and needs only minimal parameterisation. Although the discarding of simulation results seems to be counterintuitive the results are promising. Finally, the statistical approach using Student's t-test outperformed all other techniques significantly. Using additional simulation runs only when they are actually required lead to the most efficient use of available resources. In the next section, we will discuss the results of the paper and put them into the right context.

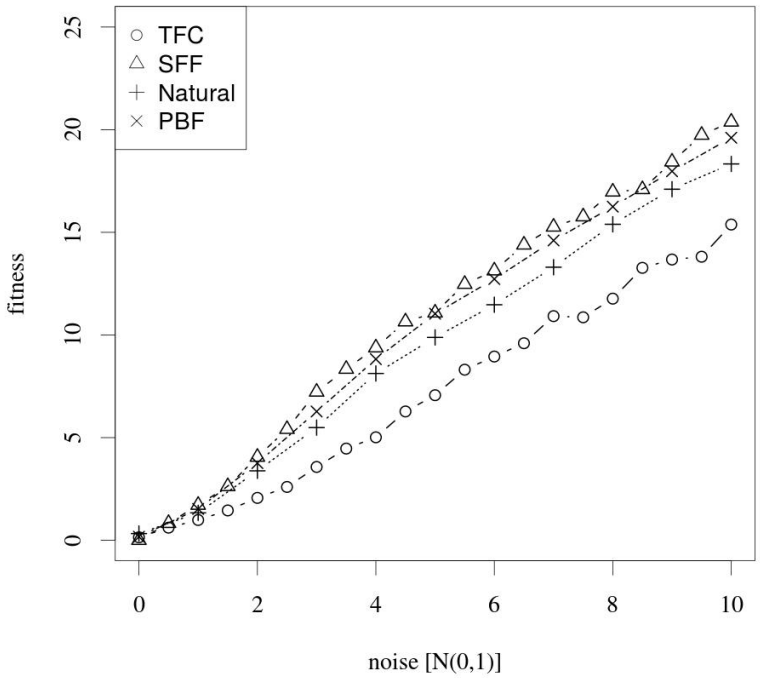

Figure 5: *Comparison of approaches (smaller fitness values are better).*

5 Discussion

Our experiments provide some interesting points for discussion. First, we need to address the obvious critique regarding the use of a surrogate function instead of an actual simulation. There is a strong indication that our theoretical results can be applied to actual SBO, since the employed function is comparable in behaviour and complexity. Furthermore, the work of Frank et al. (2013) suggests the transferability of the results. Initial experiments we performed using actual simulation supports our assumption. An additional point for discussion is the simple error model we employed. Granted, we cannot expect normally distributed errors sharing one standard deviation for all candidate solutions. Therefore, further experiments using different error models are required.

The discussed optimisation approaches were mainly compared to the classic reference approach. Currently, we have not tested the OCBA approach (Optimal Computing Budget Allocation) approach by Chen (2003). Therefore, we cannot provide any insights on its effectiveness in comparison to the discussed approaches.

One important result of our experiments is the large influence of the parameter setup. For example, the good performance of the phase based approach observed in Frank (2013) can probably be attributed to a suboptimal setup of the reference approach. Additionally, the phase-based approach is very hard to set up properly, since it needs a huge amount of parameters. Switching phases based on the slope is very sensitive. It might be beneficial to consider other measurements to determine the appropriate time of switching to the next phase. For example, the observed variance of the fitness of a given population can be used instead. A study evaluating this approach should also consider experiments with regard to the amount and setup of phases. For further use of the phase-based approach, it would be beneficial to develop a deeper understanding on the principles of its parameterisation. Otherwise, the amount of parameters makes the phase-based approach rather impracticable.

Regarding the other approaches, we can also consider further improvements. For a discussion on the t-test based approach, we refer you to Uhlig and Rose (2015). The natural approach can modified to keep the results of simulation runs instead of discarding them completely. For example, it could simply update the fitness for each generation instead of completely resetting it. Potential changes should probably try to preserve the simple nature of this approach, since the simple parameterisation of the approach is its biggest drawing point.

6 Summary and Outlook

In this paper, we compared three advanced approaches to the traditional comparison strategy used for simulation-based optimisation. All of them outperformed the classical approach that relies on a fixed amount of simulation runs. Nevertheless, our experiments show that the traditional approach can operate quite efficiently, if it is parameterised correctly. Determining adequate parameters, however, is challenging. We either have to rely on intuition and previous experience, which can be misleading, or we have to determine the parameters by performing extensive experiments, which is costly. This predicament is also true for the phase-based approach. With this in mind, it is important to point out that the other two approaches are much more robust with respect to their parameterisation. The “Natural” approach returns

good results, while only requiring a minimum of setup. The statistical approach shows the best performance and significantly exceeded all other approaches.

Future research in this area needs to evaluate other advanced approaches in comparison to the proposed techniques. Furthermore, the proposed methods have to be evaluated in other scenarios. For a final verdict, we have to consider the use of actual simulations in contrast to the surrogate benchmark function and we have to evaluate the influence of different kinds of uncertainty.

Acknowledgement

We want to acknowledge the work of our student Dominik Schwindling, who has performed initial experiments for our studies using actual simulation models. His results provide a strong indication for the transferability of the presented results.

References

Arnold, D. V.; Beyer, H.-G.: A comparison of evolution strategies with other direct search methods in the presence of noise. Computational Optimization and Applications 24(2003) 1, pp. 135-159.

Chen, C.-H.: Efficient sampling for simulation-based optimization under uncertainty. Fourth International Symposium on Uncertainty Modeling and Analysis, College Park, Maryland (USA), 2003, pp. 386-391.

Frank, M.; Laroque C.; Uhlig, T.: Reducing computation time in simulation-based optimization of manufacturing systems. In: Pasupathy, R.; Kim, S.-H.; Tolk, A.; Hill, R.; Kuhl, M. E. (eds.): Proceedings of the 2013 Winter Simulation Conference, Washington (USA), December 8th-11th2013, pp. 2710-2721.

Hagendorf, O.: Simulation based parameter and structure optimization of discrete systems. ARGESIM Report AR 67. Wien: ARGESIM / ASIM 2010.

Jin, Y.; Branke, J.: Evolutionary optimization in uncertain environments – A survey. IEEE Transactions on Evolutionary Computation 9 (2005) 3, pp. 303-317.

Law, A. M.; Kelton, D. M.: Simulation Modeling and Analysis, 3rd ed. Boston: McGraw-Hill Higher Education 1999.

Mühlenbein, H; Schomisch M.; Born J.: The parallel genetic algorithm as function optimizer. Parallel Computing Volume 17 (1991) 6, pp. 619-632

Uhlig, T.: Serein – a framework to Model Metaheuristics, website: http://sourceforge.net/projects/serein/, 2013.

Uhlig, T.: Self-replicating individuals – An approach to model metaheuristics and hyper-heuristics. Phd. Thesis, UniBw München, 2015.

Uhlig, T.; Rose, O.: Efficient use of simulation runs in simulation-based optimization. In: Yilmaz, L.; Chan, W. K. V.; Moon, V.; Roeder, T. M. K.; Macal, C.; Rossetti, M. D. (eds.): Proceedings of the 2015 Winter Simulation Conference, Savannah (USA), December 7th-10th 2015 [submitted].