Self-adaptive Scouting—Autonomous Experimentation for Systems Biology

Naoki Matsumaru[1], Florian Centler[1], Klaus-Peter Zauner[2], and Peter Dittrich[1]

[1] Bio Systems Analysis Group
Jena Centre for Bioinformatics (JCB) and
Department of Mathematics and Computer Science
Friedrich-Schiller-University Jena, D-07743 Jena, Germany
[2] School of Electronics and Computer Science
University of Southampton, SO17 1BJ, United Kingdom

Abstract. We introduce a new algorithm for autonomous experimentation. This algorithm uses evolution to drive exploration during scientific discovery. Population size and mutation strength are self-adaptive. The only variables remaining to be set are the limits and maximum resolution of the parameters in the experiment. In practice, these are determined by instrumentation. Aside from conducting physical experiments, the algorithm is a valuable tool for investigating simulation models of biological systems. We illustrate the operation of the algorithm on a model of HIV-immune system interaction. Finally, the difference between scouting and optimization is discussed.

1 Introduction

Perplexing complexity is prevalent in biology across the scale from single cells to ecosystems. Unraveling the intricate and manifold interplay of components in biological systems necessitates comprehensive information. Acquiring this information is challenging because the complexity of living systems entails extensive factor interaction. As an implication of these interactions the results of biological experiments have in general a narrow scope. Thus it is often impossible to synthesize quantitative system-level models form data in the existing literature, because measurements were obtained in disparate or unreported contexts.

Consequently, experiments are the limiting resource for quantitative systems biology. Automated high-throughput methods and recent sensor technologies are well suited to address this problem. To realize their potential, however, computational techniques have to be brought to bear not only to discover regularities in existing data, but rather the experimental procedure itself has to be embedded in a closed-loop discovery process. Only the latter affords the intervention of the computer during the experimental process required for full utilization of both the material subject to investigation and equipment time [1, 2].

With the advent of large-scale biological models the need to apply autonomous experimentation also to simulations became apparent. For example, a differential

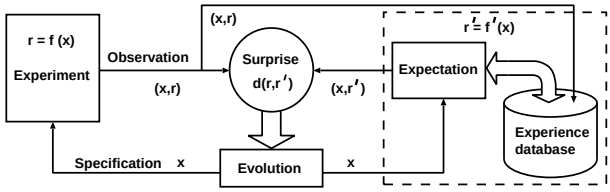

Fig. 1. Scouting combines the notion of information being equivalent to surprise value [11] with evolutionary computation for autonomous exploration. See text for details

equation model of the epidermal growth factor receptor signaling pathway contains 94 time varying state variables and 95 parameters [3]; or a recent model of E. coli contains more than 1200 metabolites and reactions [4]. Aside from the combinatorial explosion of the parameter space with increasing number of parameters, the analysis of the dynamic behavior of such systems is further complicated by the the rich interactions among the parameters that is typical for biological systems.

In the present context, the work by Kulkarni and Simon [5] is of particular interest. They developed a program that attempts to generate experiments, in which unexplained phenomena are enhanced. Notably, the program does not start out with a pre-set goal as is common in optimization experiments but decides on its objectives dynamically. This work demonstrated that an algorithm can successfully navigate an immense search space by emulating the interplay of adjusting hypotheses and modifying experiments, which is characteristic of human experimenters [6]. Recently, evolutionary computation has been applied to autonomous experimentation [7] and was employed in conjunction with computer-controlled fluidics to characterize protein response with regard to chemical signals [8]. This method, named *scouting*, has also been suggested for detecting and localizing unusual chemical signatures [9]. A drawback of the scouting algorithm so far was that several application-dependent parameters need to be set by the user to achieve good performance.

We developed *self-adaptive scouting* combining the scouting algorithm with two parameter adaptation strategies. In the following, first the improved scouting algorithm with self-adaptive mutation strength and population size is presented. Then, we illustrate the operation of the new algorithm on a simulation model of HIV-immune system interaction [10]. Finally, the crucial difference between scouting and optimization is discussed in Section 4.

2 The Self-adaptive Scouting Algorithm

The scouting algorithm is an evolutionary experimentation method for designing experiments dynamically, and experiments are scheduled to achieve maximal information gain at each step. In accordance with communication theory, information is quantified as the surprise value of arriving data [11]. No a priori knowledge about the system under investigation is required. An overview of the algorithm is depicted in Fig. 1. Given an experiment $\mathbf{f}$, the scouting algorithm

interacts with it by sending specifications $\mathbf{x}$ of the experiment and receiving observed responses $\mathbf{r}$. During the scouting run, every experiment performed contributes to an experience database, which stores the observations (pairs of $\mathbf{x}$ and $\mathbf{r}$). This database together with a prediction mechanics (labeled as expectation) forms an empirical model $\mathbf{f}'$ of the experiment given, and this model is used to formulate an expected response $\mathbf{r}'$ from the experiment. The deviation $\mathbf{d}(\mathbf{r}, \mathbf{r}')$ between the expected and the real response constitutes the *surprise* for the specification. In the evolution step, the specification is an individual offspring and the surprise value is used as the fitness in generating the next offsprings. As a result of this algorithm, experiments are performed densely in regions where unexpected observations occur.

2.1 Adaptive Mutation Strength

In the evolution step of the algorithm, an offspring is created from the parent by adding a normally distributed value with mean 0 and standard deviation σ. Varying σ controls the strength of the mutation. Given the current surprise value s and the current mutation strength σ, the mutation strength is adapted as follows:

$$\sigma \leftarrow \sigma \cdot e^{(\bar{s}-s)/\bar{s}}, \tag{1}$$

where $\bar{s}$ is the average surprise value over all past experiments. As a result of this adaptation, the region from which offspring are chosen shrinks if the current surprise is above the average surprise. A surprise value below the average, on the other hand, causes the region to expand—eventually leading to random search.

We also investigated standard step size adaption methods from evolution strategies (ES), e.g., ref. [12, 13] and found that those adaptation methods fail to keep up with the dynamics of our fitness landscape[1].

2.2 Adaptive Population Size

Originally, the population size λ had to be given by the user and determines the number of offspring generated from one parent (i.e., a $(1, \lambda)$-strategy in ES-terminology). If the population size is constant, we found that some individuals with high fitness are discarded because there was one individual with even higher fitness selected as the parent for the next generation. Conversely, it also happened that individuals with low fitness were selected as a parent.

The second adaptation scheme is developed to avoid this situation. We introduced an adaptive generation change. Whenever the surprise value is higher

[1] In the scouting algorithm, the fitness is obtained as the deviation between an expectation computed from the experience database and an observation. Since every experiment is stored in the experience database, the expectation improves continuously and the fitness landscape changes rapidly. In a deterministic setting (e.g., where the experiment is a simulation without randomness) even the individual with the highest fitness will have zero fitness when it is evaluated a second time (cf. [14, 15]).

than a threshold, the specification of the experiment is selected as a parent. The threshold at generation g is denoted as $\Theta^{(g)}$ and defined as the average surprise value of the second-best individuals in the past generations:

$$\Theta^{(g)} := \frac{1}{g-1} \sum_{k=1}^{g-1} s_{2;\lambda_k}^{(k)} \quad \text{for } g > 1, \qquad \Theta^{(1)} := 0. \tag{2}$$

Following Beyer and Schwefel [16], we use the notation of *order statistics* (e.g., ref. [17]) by identifying the surprise value of the second-best individual out of λ_k individuals of generation k by $s_{2;\lambda_k}^{(k)}$. The population size at generation k is λ_k. For a generation with only one member, we define this individual to be the "second-best": $s_{2;1}^{(k)} := s_1^{(k)}$, where $s_i^{(k)}$ is the surprise value of the ith individual of generation k (in the order of experiments performed).

The threshold is calculated as described above because the second-best surprise value separates the best, which is selected as the parent for the next generation, from the other offspring. The second-best surprise value works implicitly as a threshold in each generation. Furthermore, this method guarantees the threshold to be above the average surprise value $\bar{s}$, so that the mutation strength σ can become both bigger and smaller using the scheme of Sec. 2.1.

2.3 Pseudo Code

The complete scouting algorithm is presented here in more detail as pseudo code. During initialization (line 1–5), the minimum mutation strength σ_{min} and the number of experiments to perform t_{max} is set by the user. The mutation strength σ is initialized with 1. The number of the current experiment t and the number of the current generation g is set to 0. In line 6–9, an initial experiment specification is randomly chosen, the experiment performed, and stored with response $\mathbf{r}$ in the experience database DB. $\mathbf{x}_i^{(g)}$ is the experiment specification of the ith individual of generation g. Within the **while**-loop, a new generation is started by choosing the parent $\mathbf{x}^{(g)}$ of generation g to be the last individual of the last generation (line 13). The new generation is then populated within the **repeat**-loop. In line 16, a new experiment specification is created as a mutated copy of the parent individual (see Sec. 2.1), and the expectation $\mathbf{r}'$ is computed from the experience database. This is done here, as in ref. [8], by averaging over the (up to) 5 closest experiment entries from the experience database with inverse cubic distance weighting. The response $\mathbf{r}$ is derived by performing the experiment, and the result is stored in the experience database. Finally, the surprise value is calculated in line 21, and the mutation strength is adapted in line 22 (see Sec. 2.1). In generation g, the mean surprise value over all experiments is calculated as follows:

$$\bar{s} := \frac{1}{g} \sum_{k=1}^{g} \frac{1}{\lambda_k} \sum_{i=1}^{\lambda_k} s_i^{(k)}. \tag{3}$$

The **repeat**-loop is left and a new generation started, once the surprise value of an experiment is above the threshold $\Theta^{(g)}$ (see Sec. 2.2).

Algorithm: Self-adaptive Scouting

$\sigma_{min} \leftarrow$ minimum_resolution `# minimum resolution`
$t_{max} \leftarrow$ maximum_experiments `# number of exp. to perform`
$\sigma \leftarrow 1$ `# mutation strength`
$t \leftarrow 0$ `# time (experiments)`
$g \leftarrow 0$ `# time (generations)`
$\mathbf{x}_{\lambda_0}^{(0)} \leftarrow$ random_specification `# choose initial experiment`
$\mathbf{r} \leftarrow f(\mathbf{x}_{\lambda_0}^{(0)})$ `# conduct experiment`
$t \leftarrow t+1$ `# increment time (experiments)`
DB $\leftarrow$ `InsertIntoDB(DB, (`$\mathbf{x}_1^{(0)}, \mathbf{r}$`))` `# initialize experience db`
while $t < t_{max}$ **do**
  $g \leftarrow g+1$ `# start a new generation`
  $\lambda_g \leftarrow 0$ `# number of individuals`
  $\mathbf{x}^{(g)} \leftarrow \mathbf{x}_{\lambda_{(g-1)}}^{(g-1)}$ `# choose the parent`
  repeat
    $\lambda_g \leftarrow \lambda_g + 1$ `# add a new individual by`
    $\mathbf{x}_{\lambda_g}^{(g)} \leftarrow$ `Mutate(`$\mathbf{x}^{(g)}, \sigma$`)` `# mutating the parent`
    $\mathbf{r}' \leftarrow$ `Predict(DB,`$\mathbf{x}_{\lambda_g}^{(g)}$`)` `# compute expectation`
    $\mathbf{r} \leftarrow f(\mathbf{x}_{\lambda_g}^{(g)})$ `# conduct experiment`
    $t \leftarrow t+1$ `# increment time (experiments)`
    DB $\leftarrow$ `InsertIntoDB(DB, (`$\mathbf{x}_{\lambda_g}^{(g)}, \mathbf{r}$`))` `# experience database`
    $s_{\lambda_g}^{(g)} \leftarrow$ `d(`$\mathbf{r}, \mathbf{r}'$`)` `# compute surprise`
    $\sigma \leftarrow$ `Max(`$\sigma_{min}, \sigma \cdot \exp(1 - s_{\lambda_g}^{(g)}/\bar{s})$`)` `# adapt mutation strength`
  until $s_{\lambda_g}^{(g)} > \Theta^{(g)}$ *or* $t \geq t_{max}$
end

3 Scouting an HIV-immune System Model

Now we demonstrate the behavior of our new algorithm by applying it to a concrete model of immunological control of HIV by Wodarz and Nowak [10]. The model is a 4-dimensional ordinary differential equation (ODE) system. A mathematical analysis reveals that the model has two asymptotically stable fixed points. Using scouting, we will now explore how the model behaves depending on its initial state given a fixed parameter setting.

3.1 Definition of the Experiment

The experiment is a dynamic simulation of the the immunological control model taken from [10], which contains 4 variables: uninfected $CD4^+$ T cells x, infected $CD4^+$ T cells y, cytotoxic T lymphocyte (CTL) precursors (CTLp) w, and CTL effectors z. The dynamics is given by the ODE system: $\dot{x} = \lambda - dx - \beta xy, \dot{y} = \beta xy - ay - pyz, \dot{w} = cxyw - cqyw - bw, \dot{z} = cqyw - hz$. Uninfected $CD4^+$ T

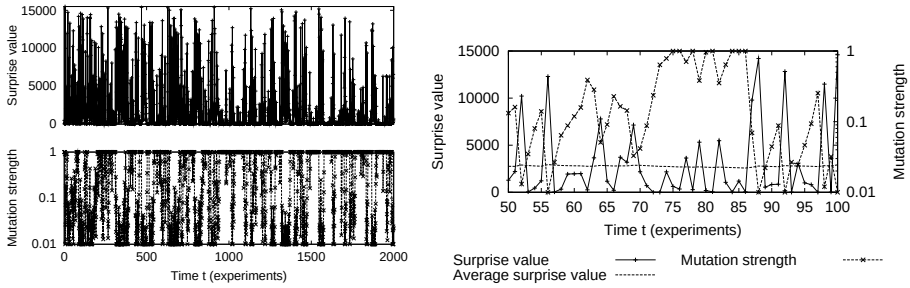

Fig. 2. Surprise value $s_i^{(g)}$ and mutation strength σ while exploring the behavior of the HIV immunology model with the self-adaptive scouting algorithm. A surprise value higher than the average decreases the mutation strength. The mutation strength is increased, when the surprise value is less than the average (see Eq. 1). This mutation strength adjustment helps the scouting algorithm to concentrate the samples on surprising regions

cells are produced at a rate λ, decay at a rate dx, and become infected at a rate βxy. Infected cells decay at a rate ay and are killed by CTL effectors at a rate pyz. The production of CTLp at rate $cxyw$ requires uninfected $CD4^+$ cells, virus load represented by y, and CTLp themselves. CTLp decay at a rate bw and differentiate in CTL effectors at a rate $cqyw$. CTL effectors decay at a rate hz. Here we set the parameters as in ref. [10]: $\lambda = 1, d = 0.1, \beta = 0.5, a = 0.2, p = 1, c = 0.1, b = 0.01, q = 0.5, h = 0.1$.

Given a specification $\mathbf{x} = (x_1, x_2, x_3, x_4)$, we perform the experiment and obtain the response $\mathbf{r} = (r_1, r_2, r_3, r_4)$ in the following way: (1) We set the initial state of the ODE system as follows: $x_{t=0} = x_1, y_{t=0} = x_2, w_{t=0} = x_3 \times 0.05, z_{t=0} = x_4$. (2) We integrate the ODE numerically (using *lsode* built in *octave* [18]) for a duration of t_1 (here, $t_1 = 500$) and obtain the response as the final state: $r_1 = x_{t=t_1}, r_2 = y_{t=t_1}, r_3 = w_{t=t_1}, r_4 = z_{t=t_1}$.

3.2 Scouting the Model

For scouting, we set the range of specification $\mathbf{x}$ of the experiment to $[0, 1]^4$ and the minimum mutation strength $\sigma_{min} = 0.01$. We allow a total number of $t_{max} = 2000$ experiments in a scouting run. Every experiment integrates numerically the ODE representing the HIV immunology model.

Figure 2 shows the progress of the surprise value $s_j^{(g)}$ (left top) and mutation strength σ (left bottom) of a typical run of (self-adaptive) scouting. On the right-hand side, experiments 50–100 are shown in detail. Surprise value and mutation strength are plotted together with the average surprise value $\bar{s}$ to illustrate the mutation strength adaptation. The difference between the current surprise value and the average surprise determines the adaptation of the mutation strength according to Eq. 1.

The time evolution of the population size λ_g is plotted on the lefthand side of Fig. 3. For the purpose of explaining the population size adaptation, the right-hand side of the graph shows in detail the surprise value of every individual

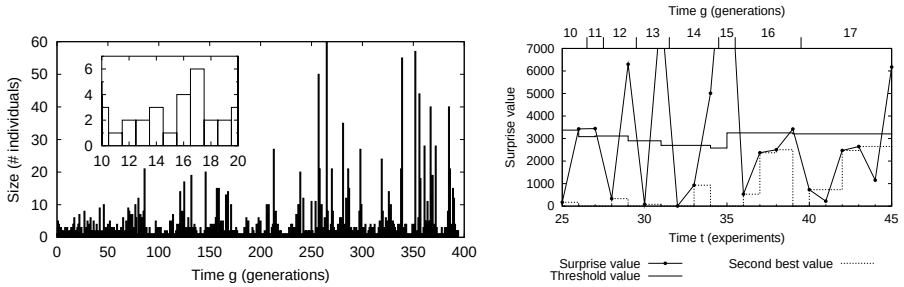

Fig. 3. Population size λ_g and threshold $\Theta^{(g)}$. See text for details

experiment from generation 10–17, the second-best surprise, and the threshold. Generation 17 consists of 6 individuals $\mathbf{x}_{40} \dots \mathbf{x}_{45}$ ($\mathbf{x}_i$ denotes the ith experiment conducted). Because the surprise of the 6th offspring is greater than the threshold, the individual $\mathbf{x}_{45}$ is selected to be the parent of generation 18. The surprise experienced by the experiment $\mathbf{x}_{43} = \mathbf{x}_4^{(17)}$ is the second-best surprise value in the generation. This value is used to calculate the threshold for the following generation. Since the first offspring of generation 15 ($\mathbf{x}_{35} = \mathbf{x}_1^{(15)}$) yields a higher surprise than the threshold, the population size of generation 15 is 1. The second-best surprise value for this generation is, in this case, the best one.

Figure 4 shows the 2000 specifications sampled by the scouting algorithm. The 4-dimensional data is projected on 6 graphs showing every possible combination of the four dimensions. Each point represents an initial state of the ODE model. The sampling points seem to spread equally except for the last graph with CTL precursors and effectors as axes. The pattern appearing in the last graph matches with the border of the two basins of attraction of the two asymptotically stable fixed points of the ODE model. The respective dynamical behavior of the model is shown in Fig. 5. As seen in Fig. 4, scouting has explored the borderline between the two modes of behavior more accurately, thus the borderline can be described much more precisely than for cases where systematic or random sampling would have been used. To illustrate this, a 2-dimensional projection of systematic sampling given by a full 7^4 factorial design is shown in Fig. 6. In this design of experiments, each of the 4 factors $x_1, \dots, x_4$ is explored equidistantly on 7 levels [19]. With approximately the same number of experiments as in the scouting method, the borderline can only roughly be approximated.

4 Scouting is not Optimization

It is important to note that scouting as described here is not a classical optimization method. Generally, the aim of optimization algorithms is finding the best solution among all possible solutions. More formally, given an objective function $F\colon Y \to Q$, which assigns to each solution from the search space Y a certain quality $\mathbf{q} \in Q$, an optimization algorithm tries to find the solution $\mathbf{y} \in Y$ such that $F(\mathbf{y})$ gets maximized. The result of optimization is a single best-so-far solution $\mathbf{y}$. For scouting, an experiment $f\colon X \to R$ is given, where X is the

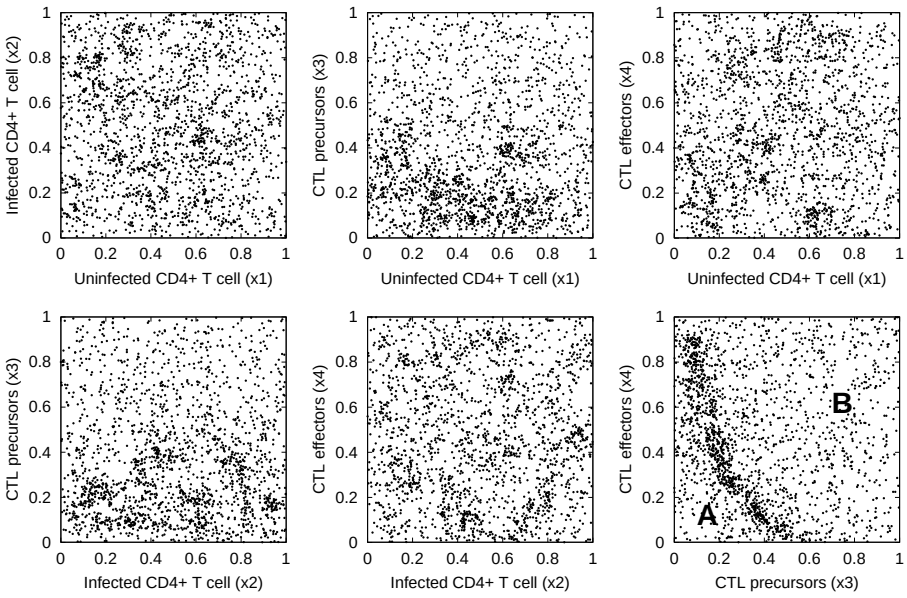

Fig. 4. Six different 2-dimensional projections of 2000 sampled locations (specifications) of a typical scouting run on the HIV immunology model developed by Woodarz and Nowak [10]. This model is described as an ODE with four variables. The scouting algorithm initializes the variables, which are plotted as dots, and observes the states of the model after 500 time steps as the response. When the locations are plotted with CTL precursors and effectors as axes (lower rightmost plot), a pattern of dense area shows up. The pattern matches with the border of two modes of behavior of the model, which are shown in Fig. 5

search space consisting of all possible experiment specifications and R the set of all possible responses. In contrast to optimization, the objective here is not to find a single best experiment $\mathbf{x} \in X$ (or a pareto set), but to gain as much information about f as possible by conducting experiments. The result of scouting is an experience database, which embodies the complete knowledge acquired about f and can be considered as an empirical model.

Trivially, every computational problem can be formulated as an optimization problem, e.g., by defining the objective function to be optimum when its argument is the solution of the problem. For example, a sorted sequence $\mathbf{y} = (y_1, \ldots, y_n)$ optimizes the objective function $F(\mathbf{y}) = \sum_{i<j}(y_i < y_j)$. Most sorting algorithms, however, contradict the typical picture of optimization where a sequence of evaluations of the objective function leads to a solution. *Bubble sort*, for instance, might be better regarded as a greedy strategy that seeks a local optimum to achieve the global optimum. The same is true for scouting with respect to the (implicit) aim of maximizing the total information about the experiment f. Every step (or every generation) of the scouting algorithm can be viewed as a step of a greedy strategy that tries to maximize the local information gain in terms of maximal surprise in the next experiment.

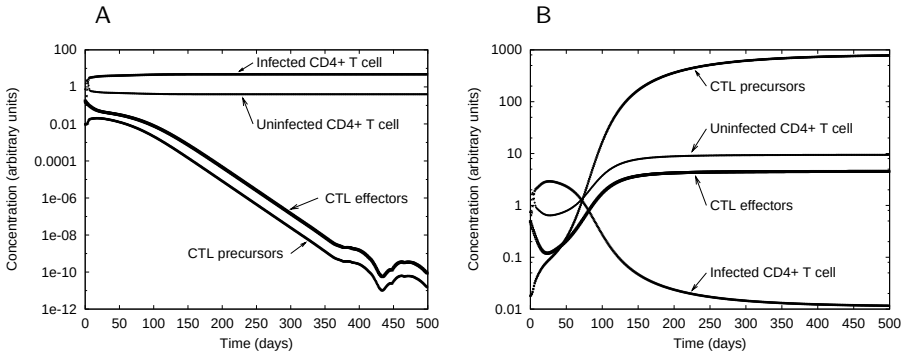

Fig. 5. The HIV immunology model has two modes of behavior: (A) immune system damaged and (B) CTL response established. The labels correspond to those in Fig. 4

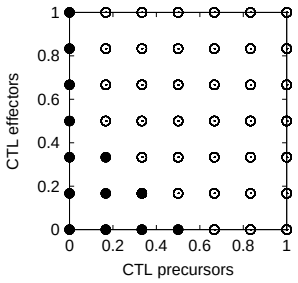

Fig. 6. Two-dimensional projection of systematic sampling. Initial state leading to a damaged immune system are marked by •; initial states establishing CTL response are indicated by ◦. Experiments with no initial infected $CD4^+$ T cells are excluded. Comparing to the lower rightmost panel in Fig. 4, the borderline of the two modes of behavior of the model can be approximated only roughly

5 Concluding Remarks

We introduced an algorithm capable of exploring unknown phenomena without the need for manual adjustments aimed at an application domain. We described how the two parameters crucial for the exploration, the mutation strength and the population size, can be adapted automatically, and why existing techniques for evolutionary optimization were not applicable. Our experience with the algorithm provides some evidence that it can be applied usefully for exploring complex systems. However, the next important step is to quantify the performance of scouting systematically. A suitable assessment measure may be the predicting strength of the experience database after a given number of samples. The process of evolution underlies the complexity observed throughout the realms of biology—it may also hold the key to tackle this complexity.

Acknowledgments: This article is based upon work supported by BMBF (Federal Ministry of Education and Research, Germany) under grant No. 0312704A to PD, and NASA under Grant No. NCC2-1189 to KPZ.

References

1. Langley, P., Simon, H.A., Bradshaw, G.L., Żytkow, J.M.: Scientific discovery: Computational exploration of the creative processes. MIT Press, Cambridge, MA (1987)
2. King, R.D., Whelan, K.E., Jones, F.M., Reiser, P.G.K., Bryant, C.H., Muggleton, S.H., Kell, D.B., Oliver, S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427** (2004) 247–252
3. Schoeberl, B., Eichler-Jonsson, C., Gilles, E.D., Müller, G.: Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. Nature Biotechnology **20** (2002) 370–375
4. Goryanin, I.: Progress in computational systems biology. Talk held on the Symposiumon Integrative Bioinformatics, Bielefeld, 4-5 August 2003 (2003)
5. Kulkarni, D., Simon, H.A.: Experimentation in machine discovery. In Shrager, J., Langley, P., eds.: Computational Models of Scientific Discovery and Theory Formation. Morgan Kaufmann Pubishers, San Mateo, CA (1990) 255–273
6. Gooding, D.: Experiment and the Making of Meaning. Kluwer Academic Publishers, Dordrecht (1990)
7. Pfaffmann, J.O., Zauner, K.P.: Scouting context-sensitive components. In Keymeulen, D., Stoica, A., Lohn, J., Zebulum, R.S., eds.: The Third NASA/DoD Workshop on Evolvable Hardware—EH-2001, Long Beach, 12-14 July 2001, IEEE Computer Society, Los Alamitos (2001) 14–20
8. Matsumaru, N., Colombano, S., Zauner, K.P.: Scouting enzyme behavior. In Fogel, D.B., El-Sharkawi, M.A., Yao, X., Greenwood, G., Iba, H., Marrow, P., Shackleton, M., eds.: 2002 World Congress on Computational Intelligence, May 12-17, Honolulu, Hawaii, IEEE, Piscataway, NJ (2002) CEC 19–24
9. Centler, F., Dittrich, P., Ku, L., Matsumaru, N., Pfaffmann, J., Zauner, K.P.: Artificial life as an aid to astrobiology: Testing life seeking techniques. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J.T., Ziegler, J., eds.: Advances in Artificial Life, ECAL 2003. Volume 2801 of LNAI., Berlin, Springer (2003) 31–40
10. Wodarz, D., Nowak, M.A.: Specific therapy regimes could lead to long-term immunological control of HIV. PNAS **96** (1999) 14464–9
11. Cherry, C.: Chapter 5. In: On Human Communication: A Review, a Survey, and a Criticism. 2nd edn. MIT Press, Cambridge, MA (1966)
12. Rechenberg, I.: Evolutionsstrategie'94. frommann-holzboog, Stuttgard (1994)
13. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9** (2001) 159–195
14. Weicker, K., Weicker, N.: On evolution strategy optimization in dynamic environments. In Angeline, P.J., Michalewicz, Z., Schoenauer, M., Yao, X., Zalzala, A., eds.: Proceedings of the Congress on Evolutionary Computation. Volume 3., Mayflower Hotel, Washington D.C., USA, IEEE Press (1999) 2039–2046
15. Arnold, D.V., Beyer, H.G.: Random dynamics optimum tracking with evolution strategies. In Guervós, J.J.M., Adamidis, P., Beyer, H.G., nas, J.L.F.V., Schwefel, H.P., eds.: Parallel Problem Solving from Nature VII (PPSN-2002). Volume 2439 of LNCS., Granada, Spain, Springer Verlag (2002) 3–12
16. Beyer, H.G., Schwefel, H.P.: Evolution strategies - a comprehensive introduction. Natural Computing **1** (2002) 3–52
17. Arnold, B., Balakrishnan, N., Nagaraja, H.: A First Course in Order Statistics. New York. Wiley (1992)
18. Eaton, J.W.: GNU Octave Manual. Network Theory Limited, Bristol, UK (2002)
19. Hinkelmann, K., Kempthorn, O.: Design and Analysis of Experiments. Wiley, New York (1994)