Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

On evolutionary optimization with approximate fitness functions

Yaochu Jin, Markus Olhofer, Bernhard Sendhoff

2000

Preprint:

This is an accepted article published in Proceedings of the Genetic and Evolutionary Computation Conference GECCO. The final authenticated version is available online at: https://doi.org/[DOI not available]

On Evolutionary Optimization with Approximate Fitness Functions

Yaochu Jin, Markus Olhofer and Bernhard Sendhoff
Future Technology Research
Honda R&D Europe (D) GmbH
63073 Offenbach/Main, Germany
Email: {yaochu.jin, markus.olhofer, bernhard.sendhoff}@hre-ftr.f.rd.honda.co.jp
Phone: +49-69-89011735/4/6

Abstract The evaluation of the quality of solutions is usually very time-consuming in design optimization. Therefore, time-efficient approximate models can be particularly beneficial for the evaluation when evolutionary algorithms are applied. In this paper, the convergence property of an evolution strategy (ES) with neural network based fitness evaluations is investigated. It is found that the evolutionary algorithm will converge incorrectly if the approximate model has false optima. To address this problem, two strategies to control the evolution process are introduced. In addition, methods to eliminate false minima in neural network training are proposed. The effectiveness of the methods are shown with simulation studies on the Ackley function and the Rosenbrock function.

1 Introduction

Evolutionary computation techniques are receiving increasing attention in design optimization. Compared with the conventional optimization methods, evolutionary optimization algorithms have the following characteristics:

- They do not need derivative information of the system;
- They are known to be robust (Goldberg 1989), which is substantial for optimization problems that have rugged, discontinuous and multi-modal objective functions;
- They are able to solve optimization problems with both real and integer parameters;
- They are inherently parallel and thus well suitable for multi-disciplinary design optimization (Coello 1999).

Due to these attracting properties, evolutionary algorithms have successfully been applied to mechanical and aerodynamic optimization problems, including preliminary turbine design (Tong and Gregory 1992), turbine blade design (Trigg et al. 1997; Olhofer et al. 2000), multi-disciplinary rotor blade design (Hajela and Lee 1998) and multi-disciplinary wing platform design (Obayashi et al. 1997).

One essential difficulty in generation-based approaches to aerodynamic optimization is the huge time consumption due to the high complexity of the aerodynamic analysis and the large number of evaluations needed in the evolutionary optimization. To cope with this problem, time efficient approximate models have to be used in optimization. Among other approximate models, Response Surface Methodology (Myers and Montgomery 1985), Krieging models, also referred as the Design and Analysis of Computer Experiments (DACE) model in the statistical literature (Sacks et al. 1989), and artificial neural network models (Bishop 1995) are most widely used. A combination of evolutionary computation and neural networks for aerodynamic design optimization was suggested in (Pierret 1999; Lee and Hajela 1996).

However, it is generally difficult to get a model with sufficient approximation accuracy. One of the problems is the lack of training data for the neural network, because data collection is a computationally expensive process. This is especially true when the dimension of the model is high. Due to this, the approximate model may be of low fidelity and even introduce false minima. In this case, the convergence of the evolutionary algorithm needs to be investigated. So far, most of the research work has concentrated on noisy fitness functions and very interesting results have been obtained (Beyer 1998). Generally, re-sampling during fitness evaluation and resizing the population (Fitzpatrick and Grefenstette 1988; Goldberg et al. 1992; Hammel and Bäck 1994) can reduce the influence of noise to a certain extent. Unfortunately, the convergence property of evolutionary algorithms with

approximate fitness evaluation has received much less attention. An approximate fitness function is combined with the original fitness function to accelerate the evolution process in (Ratle 1998; El-Beltagy et al. 1999). The problem of unstable and divergent optimization paths has been identified in (Ratle 1998), although it has not been addressed concretely.

In this paper, the convergence property of the evolutionary algorithms with approximate fitness evaluation is investigated empirically. We show that an incorrect convergence occurs when the approximate fitness function has false optima. To improve the convergence of the evolutionary algorithm, we then introduce the concept of *controlled evolution*, in which, the evolution proceeds using not only the approximate fitness function, but also the true fitness function. There are two possibilities to combine the true fitness function with the approximate fitness function. In the first approach, a certain number of individuals within a generation are evaluated with the true fitness function. Such individuals are called *controlled individuals*. The second approach is to introduce *controlled generation*, which means that in every M generations, $N(N \leq M)$ generations will be controlled. In a controlled generation, all the individuals are evaluated with the true fitness function. The most important question is how many individuals or how many generations should be controlled to guarantee the correct convergence of an evolutionary algorithm when false optima are present in the approximate fitness function. To answer this question, extensive simulations have been carried out on the Ackley function and the Rosenbrock function, which are widely studied in the field of evolutionary computation.

It is noticed that false optima in the approximate model pose the main problem in the evolutionary optimization. Accordingly, efforts are made to remove the false optima, which are usually located in the area where no training data are available. In this paper, the minimization problem is considered, therefore, our goal is to eliminate the false minima. Some preliminary methods are proposed to eliminate false minima in neural network training. They are applied to the Ackley function and encouraging results are obtained.

2 Convergence Properties of Evolution Strategies with Approximate Fitness Evaluations

All experiments in this paper have been carried out with a particular version of the evolution strategy (ES): the derandomized approach with the covariance matrix adaptation, which will very briefly be outlined in the next section.

2.1 The Evolution Strategy with Covariance Matrix Adaptation (CMA)

The ES can be described as follows:

$$\vec{x}(t) = \vec{x}(t-1) + \vec{\tilde{z}} \quad (1)$$

$$\sigma_i(t) = \sigma_i(t-1)\exp(\tau' z)\ \exp(\tau z_i) \quad (2)$$

$$z_i, z \sim N(0,1);\ \vec{\tilde{z}} \sim N(\vec{0}, \vec{\tilde{\sigma}}(t)^2) \quad (3)$$

where $\vec{x}$ is the parameter vector to be optimized and τ, τ' and σ_i are the strategy parameters. The σ_i are also called step-sizes and are subject to self-adaptation. The z_i, z and $\vec{\tilde{z}}$ are normally distributed random numbers and a random number vector, respectively, which characterize the mutation exercized on the strategy and the objective parameters.

The derandomized CMA differs from the standard ES mainly in three respects:

- In order to reduce the stochastic influence on the self-adaptation, one stochastic source for both the adaptation of the objective and of the strategy parameters is used. In the derandomized approach (Ostermeier 1994), the *actual* step length in the objective parameter space is used to adapt the strategy parameter. Therefore, the self-adaptation of the strategy parameters depends more directly on the local topology of the search space.
- The second method is the introduction of the cumulative step size adaptation. Whereas the standard evolution strategy extracts the necessary information for the adaptation of the strategy parameters from the population (ensemble approach), the cumulative step size adaptation relies on information collected during successive generations (time averaged approach). This leads to a reduction of the necessary population size.
- In the CMA algorithm the full covariance matrix of the probability density function

  $$f(\vec{z}) = \frac{\sqrt{\det(\vec{C}^{-1})}}{(2\pi)^{n/2}} \exp\left(-\frac{1}{2}(\vec{z}^T\, \vec{C}^{-1}\, \vec{z})\right). \quad (4)$$

  is adapted for the mutation of the objective parameter vector (B satisfies $\vec{C} = \vec{B}\vec{B}^T$ with $z_i \sim N(0,1)$, then $\vec{B}\,\vec{z} \sim N(0, \vec{C})$; $\delta(t-1)$ denotes the global step-size):

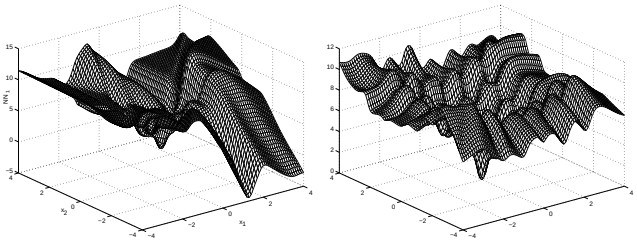
  $$\vec{x}(t) = \vec{x}(t-1) + \delta(t-1)\,\vec{B}(t-1)\,\vec{z}, \quad z_i \sim N(0,1) \quad (5)$$

Since $\vec{C}^{-1}$ has to be positive definite with $\det(\vec{C}^{-1}) > 0$, the different matrix entries cannot be determined independently and the detailed adaptation algorithm is a little more involved, see (Hansen and Ostermeier 1996; Kreutz et al. 1999) for a detailed description.

2.2 Convergence of the Evolution Strategy with Neural Networks for Fitness Evaluations

In this section, we investigate empirically the convergence property of the evolution strategy when a trained multilayer perceptron (MLP) neural network is used for fitness evaluations. The investigation is carried out on two benchmark problems, the Ackley function and the Rosenbrock function.

A neural network is trained to approximate the 2-D Ackley function. In this simulation, 600 training data are collected from one run of evolution with the Ackley function[1] as the objective function. The MLP network used has one hidden layer with 20 nodes. The rooted-mean-squared (RMS) error on the training data is 0.20 and the input-output mapping is shown in Fig. 1(a). It can be seen that a false minimum exists at the right side of the surface, near $(x_1, x_2) = (-4, 1)$. This

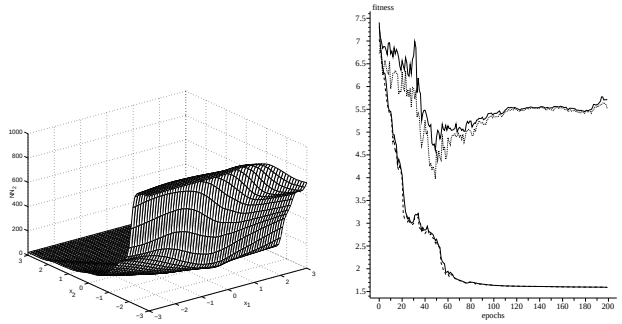


(a) (b)

Figure 1: Neural network approximation of the Ackley function; (a) with training data from optimization; (b) with additional random samples.

can be ascribed to the poor distribution of the training samples. We deliberately use such samples to simulate the situation frequently encountered in practice, i.e. sparse and poorly distributed training data. Recall that for the problems we are discussing, data collection is expensive. If we run the evolutionary algorithm with this neural network model, the algorithm converges to the false minimum, which is an expected result.

To show that the problem we identified above is a general one, simulations are also conducted on the 2-D Rosenbrock function. Similarly, we generate 150 samples from one run of evolution on the 2-D Rosenbrock function, which is much smoother compared to the Ackley function. The surface learned by the neural



(a) (b)

Figure 2: (a) The neural network approximation of the 2-D Rosenbrock function with training data from one evolution run. (b) The incorrect convergence of the ES on the 12-D Ackley function *with* random samples.

network is shown in Fig. 2(a). Comparing it with the true 2-D Rosenbrock function, it is seen that the left ramp of the 2D-Rosenbrock function becomes almost flat due to the poor distribution of the training data. No doubt, evolution with this neural network model is vulnerable to serious errors, because the global minimum of this approximate model lies in the area where the true value is very large. Therefore, an evolutionary optimization based on such an approximate model will converge to the false minimum.

2.3 Training the Neural Network with Random Samples

To illustrate that the false minima in the neural network model are caused by the poor distribution of the training data, we add some random samples to the data set. For the 2-D Ackley function, 150 data are generated randomly in addition to 450 data collected from one run of evolution. An MLP with 20 hidden nodes is trained and the resulting neural network mapping is provided in Fig. 1(b). It is noticed that the neural network has now correctly learned the main features of the Ackley function, *albeit* the RMS approximation error is larger. No false minima are introduced. This is confirmed by running the evolution process with the neural network model for fitness evaluations, in which a near-optimum is found.

Similar results are obtained for the 2-D Rosenbrock function. Therefore, we can conclude that false minima can be eliminated with some random samples for low-dimensional systems. Unfortunately, this is not

[1]The situation that training data is only available from earlier optimisation attemtpts is not uncommon in aerodynamic structure optimisation

an effective approach for high-dimensional systems. To illustrate this, simulations are conducted on the 12-D Ackley function. In the simulation, 450 random samples together with 1440 samples collected from one run of evolution are used to train the neural network. Despite that the neural network achieves a good approximation on the training data, the evolution process carried out with this neural network model still converges incorrectly. In Fig. 2(b), the solid and dashed lines in the lower part of the figure denote the mean and best fitness values during evolution, where unfortunately, the true mean and best fitness values that are denoted by the solid and dotted line in the upper part of the figure are much larger. That is to say, random samples cannot deal with the problem properly if the input dimension of the system is high. Similar results are observed on the 12-D Rosenbrock function.

3 Improvement of Convergence with Controlled Evolution

In the last section, we have seen that additional training samples are not sufficient to solve the problem of "incorrect" convergence of evolutionary algorithms with neural network models. Therefore, we introduce the concept of controlled evolution. Two methods are proposed:

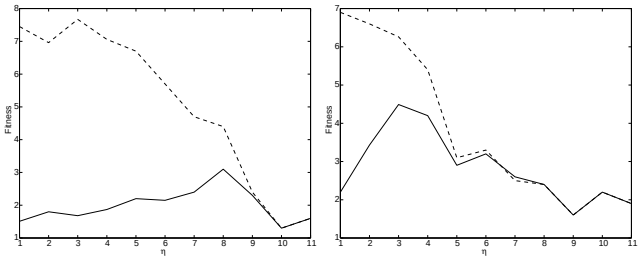
Controlled individuals In this approach, part of the individuals (η) in the population (λ in total) are chosen and evaluated with the real fitness function. If the controlled individuals are chosen randomly, we call it a **random strategy**. If we choose the best η individuals as the controlled individuals, we call it a **best strategy**.

Controlled generations In this approach, the whole population of η generations will be evaluated with the real fitness function in every λ generations, where $\eta \leq \lambda$.

Furthermore, when controlled individuals or controlled generations are introduced, new training data are available. Therefore, on-line learning of the neural network will be applied to improve the approximation of the network model in the region of optimisation and in turn to improve the convergence of the evolutionary algorithm.

3.1 Controlled individuals

We first consider the individual based methods for the 12-D Ackley function. To determine the number of individuals (η) that need to be evaluated using the original fitness function to guarantee a correct convergence, an η of 1 to 11 is tested for both the random strategy and the best strategy for a $(\mu, \lambda) = (3, 12)$-ES. The best fitness found by the ES with the random strategy reported by the neural model (solid line) and by the real fitness function (dashed line) for different η are plotted in Fig. 3(a). All the results are averaged over 10 runs. It can be seen that for the random strategy, convergence of the evolutionary algorithm is not achieved until $\eta \geq 9$. The result using the best strategy is better (see Fig. 3 (b)). When 5 out of 12 individuals are evaluated with the real fitness function, the reported best fitness is close to the true fitness. When η is larger than 7, a near-optimum is found. In Fig. 3, the x coordinate denotes the number of the controlled individuals in a population.

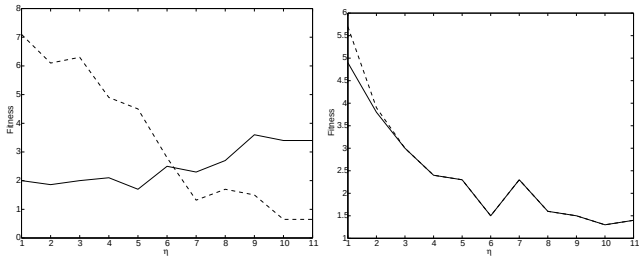


(a) (b)

Figure 3: (a) Convergence of ES with controlled individuals: (a) The random strategy; (b) the best strategy. Solid line: reported fitness; dashed line: true fitness.

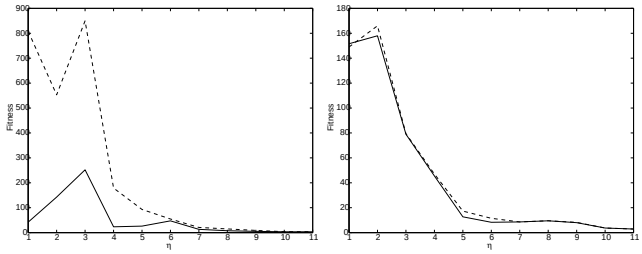
3.2 Controlled generations

The results of the generation based method are shown in Fig. 4(a), which are also averaged over 10 different runs. In the figure, the x-coordinate denotes the number of generations in which all the individuals are evaluated with the original fitness function in every 12 generations. For example, if η equals 8, then in every 12 generations, the true fitness function will be used in 8 of the 12 generations. In this way, we are able to compare the results obtained in the generation based method with the results in the individual based strategies, because in both cases, the same η entails the same computational overhead. Notice, that in which η generations out of 12 generation the original function is used is determined randomly. We see that with the increase of η, a near-optimal or optimal solution will be found, however, the average fitness reported by the neural model never converges to the true fitness.



(a) (b)

Figure 4: Convergence of ES: (a) with controlled generations; (b) with best strategy and on-line learning. Solid line: reported fitness; dashed line: true fitness.



(a) (b)

Figure 5: Convergence of ES for the Rosenbrock function: (a) with best strategy; (b) with best strategy and on-line learning. Solid line: reported fitness; dashed line: true fitness.

3.3 Controlled Evolution and On-line Learning of Neural Network Models

When some of the individuals are evaluated using the real fitness function, new data are available and on-line learning can be implemented. In the following, we investigate in which way the algorithm can benefit, if these newly available data are used to train the neural network on-line. The result for the "best strategy with on-line learning" is given in Fig. 4(b). The evolutionary algorithm reports a correct fitness when only 2 individuals are evaluated with the real fitness function. When about 50% of the whole population are evaluated with the real fitness function, a good near-optimal or optimal solution is found.

Similar results have been obtained for the 12-D Rosenbrock function. Fig. 5(a) shows the convergence of the evolutionary algorithm for different η values, where the best strategy is applied. It is seen that when $\eta \geq 6$, the algorithm converges to the correct fitness value and a near-optimal solution is found. When on-line learning is introduced, the convergence property improves considerably and the resulting solution is near-optimal when $\eta \geq 5$ (see Fig. 5(b)). In both cases, the results are based on an average of ten runs.

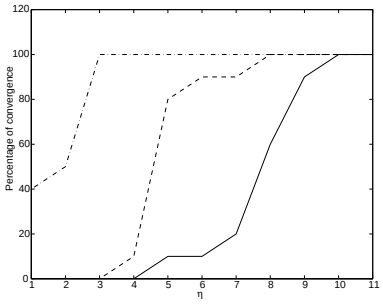
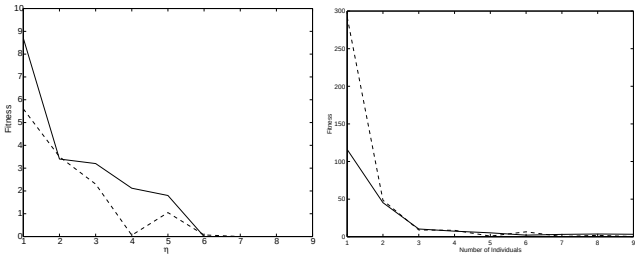


Figure 6: Percentage of correct convergence. Solid line: random strategy; dashed line: best strategy; dotted line: best + online learning.

In the following, we briefly compare the convergence property on the 12-D Ackley function for the different individual based strategies. In Fig. 6, the solid line denotes the percentage of correct convergence when the "random strategy" is adopted, while the dashed line and the dotted line represent the percentage of correct convergence when the "best strategy" is used and when on-line learning is applied. It is shown that the best strategy works much better than the random strategy and on-line learning can further improve the convergence property.

3.4 What Can One Benefit from an Approximate Model in Evolution

We have seen that when the neural network model has false minima (in particular if the global minimum is false), it is imperative to use it in conjunction with the original fitness function in order to guarantee the correct convergence of the evolutionary algorithm. Therefore, the question arises whether we can at all benefit from such a "potentially" erroneous model, or whether it is better to restrict the population size to the number of individuals which we would have to evaluate in any case with the original function in the combined approach; thus, to use a smaller population and to give up the approximate model all together. Suppose for a (μ, λ)-ES, at least η individuals have to be evaluated using the original function to guarantee the correct convergence of the algorithm, where $\mu \leq \eta \leq \lambda$. The question is whether we are better off with a (μ, η)-ES and all the individuals are evaluated with the original fitness function. Simulations are first run on the Ackley function. Fig. 7(a) shows the best result when η individuals are evaluated using the original function, where the solid line denotes the fitness from a combination of original function and the neural network



(a) (b)

Figure 7: Comparison of the best fitness when using (solid line) and not using the approximate models: (a) the Ackley function; (b) the Rosenbrock function.

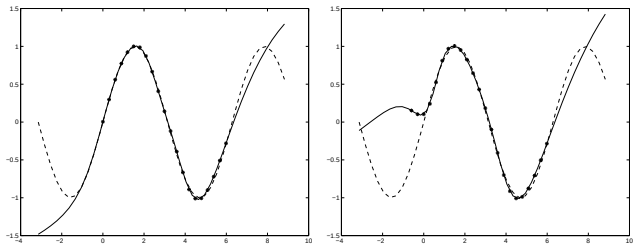
model with false minima, while the dashed line shows the best fitness using the original function only. The results shown in the figure are the best solution using different (μ, λ) combinations. Suppose an approximate model is used and 3 individuals need to be controlled. Then, we have the following possible combinations when only the original function is used: (1, 2), (1, 3) and (2, 3). Notice that when only the original function is used for evaluation, we have $\mu < \lambda \leq \eta$. In the case when both the original function and the neural network model are used, we have $\mu \leq \eta$ and $\lambda \leq 2\eta$, because $\lambda > 2\eta$ does not make much sense according to our aforementioned simulation results. For the Rosenbrock case, we made a similar comparison, which is given in Fig. 7(b).

From these results, we see that we neither benefit nor lose much from using an approximate model when the model has false minima. However, we do benefit in the following two cases.

- If there are no false minima in the approximate model, the model can be used for evolution in presence of certain range of errors. In this case, when on-line learning is introduced in the final stage of the evolution, better solutions could be obtained and much computation time can be saved.

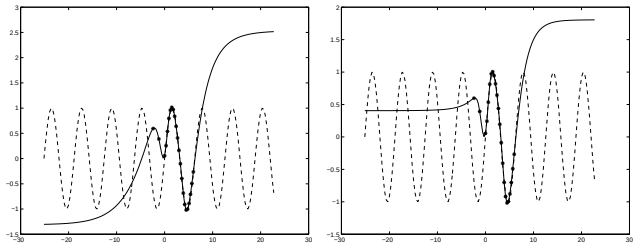
- When on-line learning is introduced, a better solution can also be achieved even if false minima are present, provided that the algorithm converges correctly.

However, to introduce large number of evaluations using the original function is undesirable. Therefore, we should additionally strive to achieve a better approximation quality of the network while minimizing the number of additional data needed.



(a) (b)

Figure 8: Network approximation of $sin(x)$: (a) Training samples only; (b) with additional artificial samples.



(a) (b)

Figure 9: Network approximation of $sin(x)$: (a) without regularization; (b) with regularization.

4 Elimination of False Minima in Neural Network Training with Regularization Techniques

Let us first consider a simple situation. Suppose a neural network is used to approximate the function $y = sin(x)$. The training data are uniformly collected in the range of $x \in [0, 2\pi]$. The approximation result is presented in Fig. 8(a). We notice that a false minimum has been introduced in the range of $x < 0$, which can be attributed to the distribution of the training data near the left bound of $[0, 2\pi]$. At first sight, Fig. 8(b), the use of artificial samples can remove the false minima. However, a closer inspection reveals, that it is merely shifted to lower values of x, which can be seen more clearly in Fig. 9(a), which shows the function for lower x values. To further investigate this phenomenon, let us have a look at the mathematics of an MLP network with one hidden layer:

$$f_{NN} = \sum_{i=1}^{H} w_i \theta(\sum_{j=1}^{n} v_{ji} x_j) \tag{6}$$

where w_i is the weight between i-th node of the hidden layer and the output layer (we call it output weight for short), H is the number of hidden nodes, v_{ji} is the weight connecting the jth input and the i-th hidden node, n is the number of inputs, and $\theta(\dot{})$ is the sigmoid

function:

$$\theta(z) = \frac{1}{1+e^{-z}}. \tag{7}$$

When all the hidden nodes of the neural network are in saturation, the output of the network can be estimated as follows:

$$f_{NN}^{\infty} = \sum_{i=1}^{H} w_i \delta_i, \tag{8}$$

where

$$\delta_i = \begin{cases} 1 & , \quad v_{ji}x_j \to +\infty \\ 0 & , \quad v_{ji}x_j \to -\infty \end{cases}. \tag{9}$$

This clearly shows that the amplitude of the network's output in saturation only depends on the amplitude of the output weight. Furthermore, we have

$$f_{NN}^{\infty} \geq -\sum_i |w_i|, \ \ \forall w_i < 0. \tag{10}$$

From the above equation, it is seen that the negative output weights should be as small as possible in order to avoid a false minimum. A straightforward way is to add a penalty term in the cost function to penalize a large negative output weight:

$$J = E + \lambda \sum_i |w_i|, \ \ w_i < 0, \tag{11}$$

where E is the conventional error function and λ is the regularization coefficient. We call it a biased weight decay, which is a special case of the weight decay method introduced in (Ishikawa 1996). The biased weight decay regularization is applied to the case discussed in Fig. 9(a). In Fig. 9(b) it can be seen that the false minimum is removed.

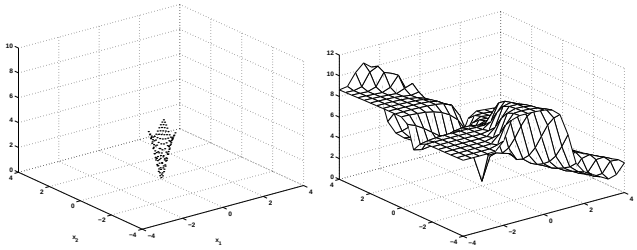
Since the idea here is to prevent the neural network from generating false minima when it is saturated, another straightforward way is to regularize the saturated output of the neural network directly. Suppose the saturated neural network output is f_{NN}^{∞}, then we can regularize the network in the following manner:

$$J = E + \lambda (y_{min} - f_{NN}^{\infty})^2, \tag{12}$$

where y_{min} is the prescribed value for the network output when it is saturated. Before this method can be applied, it is necessary to estimate f_{NN}^{∞}. When the sigmoid function in equation (7) is used, f_{NN}^{∞} can be estimated by randomly assign '0' or '1' to δ in equation (8). To get a better estimation, the final value of f_{NN}^{∞} can be obtained by averaging N random estimation:

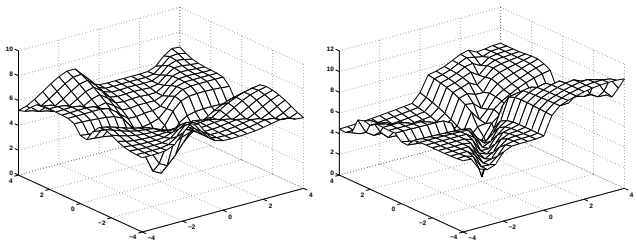
$$\tilde{f}_{NN}^{\infty} = \frac{1}{N} \sum_{i=1}^{N} f_{NN}^{\infty}(i), \tag{13}$$

where $f_{NN}^{\infty}(i)$ is the i-th random estimation of f_{NN}^{∞}.



(a) (b)

Figure 10: (a) Distribution of the training samples. (b) Trained neural network with training samples only.



(a) (b)

Figure 11: (a) Trained neural network with (a) biased weight decay and (b) regularization.

4.1 Simulation Examples

To demonstrate the feasibility of the proposed methods, first simulation studies have been carried out on the 2-D Ackley function. To simulate the situation in which the training data are poorly distributed, we use training data collected from a small area of the parameter space, as shown in Fig. 10(a). When only the training data are used, the input-output mapping of the neural network is given in Fig. 10(b). It is noticed that the learned surface is very irregular where no training data are presented and a false minimum occurs. We apply the two methods from the last section to improve the neural network approximation: 1) artificial samples plus the biased weight decay 2) artificial samples plus regularization of the saturated neural network output. The training results are provided in Fig. 11(a) and (b), respectively. We see that both methods remove the false minima.

5 Conclusions

The convergence of the evolution strategy with approximate fitness functions is empirically investigated and the occurence of false minima is identified as the main problem. Three methods are introduced to cope with it:

1. The introduction of *additional training data* for neural network learning.

2. The combination of network model and original evaluation function in the *controlled evolution approach* combined with on-line learning.

3. The introduction of approriate *regularization techniques* to avoid false minima.

Method (1) is the most straightforward approach, however in higher dimensions it is not feasible. Method (2) with on-line learning (and as such combined with 1) achieved the best results and is currently applied to evolutionary optimisation of turbine blades in a transonic compressor cascade (Olhofer et al. 2000). The main benefit of method (3) is that it does not rely on additional data and it should therefore be investigated further.

If the controlled individual method is used, the best strategy should be adopted and on-line learning is recommended. Besides, the number of the controlled individuals should be larger than 50% of the population size. If the controlled generation method is employed, there should be more than 50% of the generations in which all the individuals are controlled. It is also shown that with the best strategy and on-line learning, one is able to benefit from using an approximate fitness function even if there are false minima present in the model.

Acknowledgements The authors would like to thank T. Arima and E. Körner for their support.

References

Beyer, H.-G. (1998). Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. *Evolutionary Computation*. Submitted.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford Press.

Coello, C. (1999). An updated survey of evolutionary multiobjective optimization techniques: State of art and future trends. In *Proceedings of 1999 Congress on Evolutionary Computation*, Washington D. C.

El-Beltagy, M., P. Nair, and A. Keane (1999). Metamodeling techniques for evolutionary optimization of computationally expensive problems: promises and limitations. In *Proceedings of Genetic and Evolutionary Conference*, Orlando, Florida.

Fitzpatrick, J. and J. Grefenstette (1988). GA in noisy environments. *Machine Learning 3*.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Goldberg, D., K. Deb, and J. Clark (1992). Genetic algorithms, noise, and the sizing of the populations. *Complex Systems 6*.

Hajela, P. and J. Lee (1998). Genetic algorithms in multidisciplinary rotor blade design. In *Proceedings of 36th Structures, Structural Dynamics, and Material Conference*, New Orleans.

Hammel, U. and T. Bäck (1994). Evolution strategies on noisy functions: How to improve convergence properties. In *Proceedings of 3rd Conference on Parallel Problem Solving from Nature*, Jerusalem.

Hansen, N. and A. Ostermeier (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaption. In *Proc. 1996 IEEE Int. Conf. on Evolutionary Computation*, pp. 312–317. IEEE Press.

Ishikawa, M. (1996). Structural learning with forgetting. *Neural Networks 9*(3), 509–521.

Kreutz, M., B. Sendhoff, and C. Igel (1999, March). *EALib: A C++ class library for evolutinary algorithms* (1.4 ed.). Institut für Neuroinformatik, Ruhr-Universität Bochum.

Lee, J. and P. Hajela (1996). Parallel genetic algorithm implementation in multidisciplinary rotor blade design. *Journal of Aircraft 33*.

Myers, R. and D. Montgomery (1985). *Response Surface Methodology*. John Wiley & Sons.

Obayashi, S., Y. Yamaguchi, and T. Nakamura (1997). Multiobjective genetic algorithm for multidisciplinary design of transonic wing planform. *Journal of Aircraft 34*.

Olhofer, M., T. Arima, T. Sonoda, and B. Sendhoff (2000). Optimization of a stator blade used in a transonic compressor cascade with evolution strategies. In *Adaptive Computation in Design and Manufacture*. Submitted.

Ostermeier, A. (1994). A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation 2*(4), 369–380.

Pierret, S. (1999). Three-dimensional blade design by means of an artificial neural network and Navier-Stokes solver. In *Lecture Series at von Karman Institute for Fluid Dynamics, Belgium*.

Ratle, A. (1998). Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In *Proceedings of 5th Conference on Parallel Problem Solving from Nature*, Amsterdam.

Sacks, J., W. Welch, T. Mitchell, and H. Wynn (1989). Design and analysis of computer experiments. *Statistical Science 4*(4).

Tong, S. and B. Gregory (1992). Turbine preliminary design using artificial intelligence and numerical optimization techniques. *Journal of Turbomachinary 114*(1).

Trigg, M., G. Tubby, and A. Sheard (1997). Automatic genetic optimization approach to 2D blade profile design for steam turbine. In *Proceedings of International Gas Turbine and Aeroengine Congress*, Orlando, Florida.