Remarks on the Efficiency of Bionic Optimisation Strategies

Simon Gekeler\textsuperscript{1}, Julian Pandtle\textsuperscript{2}, Rolf Steinbuch\textsuperscript{1}, Christoph Widmann\textsuperscript{2}

\textsuperscript{1} Reutlingen University, Department of Engineering, Reutlingen Research Institute, 72762 Reutlingen, Germany
\textsuperscript{2} Now with Wafios AG, 72764 Reutlingen, Germany

Received: December 18, 2013 / Accepted: January 23, 2014 / Published: March 25, 2014.

Abstract: Bionic optimisation is one of the most popular and efficient applications of bionic engineering. As there are many different approaches and terms being used, we try to come up with a structuring of the strategies and compare the efficiency of the different methods. The methods mostly proposed in literature may be classified into evolutionary, particle swarm and artificial neural net optimisation. Some related classes have to be mentioned as the non-sexual fern optimisation and the response surfaces, which are close to the neuron nets. To come up with a measure of the efficiency that allows to take into account some of the published results the technical optimisation problems were derived from the ones given in literature. They deal with elastic studies of frame structures, as the computing time for each individual is very short. General proposals, which approach to use may not be given. It seems to be a good idea to learn about the applicability of the different methods at different problem classes and then do the optimisation according to these experiences. Furthermore in many cases there is some evidence that switching from one method to another improves the performance. Finally the identification of the exact position of the optimum by gradient methods is often more efficient than long random walks around local maxima.

Key words: Bionic optimisation, efficiency, evolutionary optimisation, Particle Swarm optimisation, artificial neural nets.

1. Introduction

Optimisation deals with modifying the free parameters of given problems to come up with better values of given objectives or goals. In structural mechanics optimisation is done by the variation of design data, e.g. the dimensions of the structure to improve the goal, e.g. to minimize the mass of a part or assembly.

The term of bionic optimisation covers all the ways to find better variants of a given design that may be related to phenomena observed in nature. This implies that many natural processes include optimisation in the sense of surviving and expanding at given environmental conditions by better adapting to the situation. As there are infinite variants of adapting to real problems a strong classification of the phenomena observed is inevitable to come up with an understanding of the optimisation histories. For this study we checked the most popular strategies, evolutionary optimisation, particle swarm optimisation and artificial neural nets. Furthermore we mention the classical methods like gradient search or response surfaces.

To decide, whether bionic optimisation strategies are an adequate tool for a given problem, the performance or efficiency of the different strategies has to be compared and checked, if they will yield acceptable results in a reasonable span of time. This measurement of the efficiency is not very well defined, as a large numbers of preliminary studies precede most optimisation done in the engineering practice.

2. Basics Terms on Bionic Optimisation

Before dealing with the different bionic optimisation strategies some terms and definitions help to come up
with a common language. Readers should take care when reading papers of different authors, as one term may be used for different subjects.

2.1 Terms and Definitions

Most people involved in optimisation accept that for an optimisation study:
- We need a given goal or objective \( z \).
- This objective \( z \) depends on a set of free parameters \( p_1, p_2, ..., p_n \).
- Limits and constraints are given for the parameter values.
- There are restrictions of the parameter combinations to avoid unacceptable solutions.
- We want to find the maximum (or minimum) of \( z(p_1, p_2, ..., p_n) \).

To better understand our ideas we propose the following conventions:
- The objective or goal has to be defined uniquely. It is not allowed to change the definition of the goal, as this poses a new question and requires a new optimisation process.
- We need to define all free parameters and their acceptable value ranges we might modify during the optimisation studies.
- This value ranges or parameter range is the span of the free parameters given by lower and upper limits.
- The less free parameters we have to take into account the faster the optimisation advances. Consequently accepting some parameters as fixed reduces the solution space and accelerates the process.
- Restrictions like unacceptable system responses or infeasible geometry must be taken into account. But restrictions limit the ranges of parameters to be searched. Such barriers have the potential to prevent the optimisation process from entering interesting regions.
- Finding the maximum of \( z(p_1, p_2, ..., p_n) \) is the same process as finding the minimum of the negative goal \( -z(p_1, p_2, ..., p_n) \). There is no need to distinguish between the search of maxima or minima.

2.2 Gradients, Response Surfaces: Derivative based Optimisations

Gradient based optimisation methods are the most popular ways to find improvements of given situations. From an initial position the derivatives of the objective \( z(p_1, p_2, ..., p_n) \) with respect to the free parameters are determined. As the derivatives are often not to be found by analytical ways they are approximated by
\[
\frac{\partial z}{\partial p_k} \approx \frac{z(p_1, p_2, ..., p_k + \Delta p, ..., p_n) - z(p_1, p_2, ..., p_k - \Delta p, ..., p_n)}{2\Delta p}
\]
(1)

The column of these derivatives defines the gradient:
\[
\nabla z = \left( \frac{\partial z}{\partial p_1}, ..., \frac{\partial z}{\partial p_n} \right)^T
\]
(2)

Jumping along this gradient, for example by using sequential quadratic programming (SQP, [1]) has the tendency to find the next local maximum in a small number of steps, as long as the search starts not too far away from this local maximum. Unfortunately the determination of the gradient requires \( 2n \) function evaluations per iteration, which may be not too small an effort if the number of parameters is large.

Response surfaces (RS) are defined by low order polynomial functions of the free parameters. For second order we may use the notation
\[
S = a_0 + a_1 p_1 + a_2 p_2 + ... + a_n p_n + a_{11} p_1^2 + a_{12} p_1 p_2 + ... + a_{n-1,n} p_{n-1} p_n + a_{nn} p_n^2
\]
\[
= a_0 + \sum_{k=1}^{n} a_k p_k + \sum_{k=1}^{n} \sum_{l=1}^{k} a_{kl} p_k p_l
\]
\[
= a_0 + P^T \nabla S + \frac{1}{2} P^T HP
\]
(3)
where \( P = (p_1, p_2, ..., p_n)^T \). Higher order schemes may be used, but have the tendency to show local oscillations or no unique maxima or minima so generally the degree of the polynomial approximation does not exceed the value of two. The coefficients \( a_0 \) and \( a_{ik} \) may be found by least squares from a sufficiently large set of function evaluations of \( z \) in the vicinity of an
Remarks on the Efficiency of Bionic Optimisation Strategies

interesting parameter set \( P_0 = (p_{10}, p_{20}, ..., p_{n0})^T \). The coefficients of the linear terms approximate the gradient \( \nabla S \) at the position of the function evaluation, the ones of the second order terms approximate the Hessian \( H \), the minimum (or maximum) of the response surface may be found by

\[
P_{\text{min}} = -H^{-1}\nabla S
\]

which is often a good guess for the position of the exact minimum of \( z(p_1, p_2, ..., p_n) \) as long as the function does not behave too irregularly. One condition is that all the second derivates of \( z \) wrt. the free parameters should have the same sign and the determinant \( H \) must not be singular. To determine the coefficients \( a_i \) and \( a_{ik} \) more than \( (n+1)^2/2 \) function evaluations are necessary, a number which may be rather large, especially in the case of an optimisation with many free parameters.

Both gradient and Response Surface approximation often help to locate an optimum faster than the bionic methods which locally do a random search and tend to miss the exact solution.

2.3 Bionic Optimisation Strategies

Bionic optimisation strategies are distinguished by different approaches. We here deal with some of the most commonly accepted classifications without taking into account all the many sub-classifications that might be found in the literature. The central approaches we want to compare are:

- Evolutionary Strategy (ES) \([2, 3, 4, 5]\), where paired or crossed parents have children by the combination and mutation of their properties. These children or a part of them are going to be parents in the next step (Fig. 1).

- Fern Strategy (FS), which may be regarded as a simplification of evolutionary optimisation. Individuals have offspring like spores by mutation only \([6]\) but not by crossing properties with other members of the parent generation (Fig. 2).

- Particle Swarm Optimisation (PSO) or simply Swarm Optimisation \([7, 8, 9, 10]\), where a population drifts through the space of possible solutions. The swarm’s coherence is given by simple rules about the velocity of the individuals (Fig. 3).

- Artificial neural nets (ANN) \([11, 12, 13, 14, 15, 16]\), where some training of the net yields an understanding of the solution space and allows the prediction of the response of a given input. Fig. 4 indicates how the surface of a trained ANN approximates a landscape with three hills. Further training would lead to a better approximation,

- Response surfaces (RS) \([17, 18]\) which may be regarded as very close to the neuron nets, but as they are not specific bionic strategies, we do not look in more depth at them in this paper. Fig. 5 demonstrates how a second order RS (cf. eq. (3)) approximates the landscape of Fig. 4. To improve the quality of the representation we need to use smaller patches to be modelled by a single RS.

Our restriction to four different methods tries to address the most important methods.

There have been strong debates between different schools whether evolutionary and genetic optimisation
Remarks on the Efficiency of Bionic Optimisation Strategies

Fig. 3  PSO: Definition of velocity components

Fig. 4  Approximation by ANN

Fig. 5  Approximation by low degree RS

are different approaches [2, 3, 4]. Genetic optimisation is related to ES but handles the genetic information in a different way. From our point of view and much experience of different working groups in engineering application this formally existing discrepancy has no central influence on the optimisation process and its results.

2.4 Measuring the Efficiency of Procedures

If we want to compare the efficiency of the different optimisation strategies, we have to introduce a measure that allows the unique definition of the amount of work required to achieve a predefined quality. This would be very easy, if the optimal solution of the test problems to be analysed (cf. section 4.1) is exactly known. Then the number of individuals required to come close to this best solution would be a measure of quality or speed of the process. But unfortunately even for relatively simple problems like the ones in sect. 4.1 there is no absolute guarantee that there are no better solutions hidden behind some sub-optima, to which our strategies tend to converge. Therefore we have to accept the best solution found by all strategies here and
the literature cited as the absolute best one. The measure of convergence quality will then be the number of individuals to be analysed before coming close to this best value. For a given and widely tested example this approach has some justification. As soon as we start studying new examples, we are not able to predict whether the absolute best has been found. Fig. 6 demonstrates this for an evolutionary optimisation.

After a certain number of generations, e.g. ten generations the best individuals are pretty close and no improvement is to be observed for some time. But after four more generations we observe a continuing improvement, which seems to go on at least to 20 generations. The search has to continue for an additional, unknown time period or number of generations.

For ANN, the measuring of the efficiency has to be done in a different way. As the training phase during which the weighting factors are determined takes very much time, the total computing time had to be taken into account. We add a number of fictive individuals which represent this learning time.

2.5 Violation of Boundary Conditions

In all sequences of parameter sets that are based on random input, the potential of violation of the restrictions or boundary conditions may occur. Simple examples are that wall thicknesses are smaller or larger than the specified limits. Other problems are related to the fact that some parameter combinations cause unfeasible geometries. Exceeding limits on physical responses, e.g. the maximum stress or displacement have to be taken into account as well. There are different ways to deal with these inacceptable parameter sets.

The easiest way is to remove all inacceptable children from the list and to continue to produce members of the respective set until the required number of acceptable individuals is found. There is no reason not to use this selection type unless the cost of a specific function evaluation is too high to produce a number of individuals that may be essentially larger than the number of usable individuals for the optimisation process. Typical examples for such expensive individuals are non-linear FE-studies, where it takes some hours of computing time to find each specific solution.

Another way to keep the population near to the feasible range is to punish all violations of the given restrictions (Fig.7a). A penalty number weights the intensity of violation. This penalty value is added to or subtracted from the objective of the individual [10]. In consequence it is less attractive for further reproduction, while the non-punished individuals have better chances to reproduce.

A third among many other ideas is to fix the parameters of the violating individual on the border of the allowable space (Fig.7b). This may be easily done for geometric input, but can be difficult if derived values like stresses or displacement have to be looked at. In such cases the reduction of the mutation from the good parents data may be used. If the parameters change less, the objective and the derived values will change less as well, so the violation may be avoided.

Many other proposals may be found in the literature cf. e.g. [8, 10]. We restrict our present study to the use of penalty functions for violations of the restrictions. The geometric input is set to the minimum or
maximum value, if the randomly produced data exceed the respective limits. For PSO, we invert the particles velocity, as soon as it violates given limits in addition to the penalty value. This combined approach has the advantage of simple applicability.

2.6 Hybrid Strategies

Many studies propose that different approaches tend to be preferable for different stages of some problems. So the idea comes up to use different strategies and switch between them during the optimisation process. This may be very efficient in many cases. We could start using evolutionary optimisation to cover a region of the solution space and change to particle swarms as soon as we feel we have reached the region of the best proposals [10, 14, 15]. But here as in other cases the question is when to use another approach. One idea could be to measure the rate of improvement. This rate may be defined as the relative improvement of the objective per individual studied. As soon as this rate drops essentially (cf. Fig. 6) another approach may be used as long as its rate of improvement is larger than the minimum accepted improvement velocity. Further changes may follow or the study terminates then.

If a bionic optimisation is not showing any further progress, it is always a good idea to use gradient or Response Surface methods to find the exact local optimum near the best solution of the bionic approach. This will yield a certain improvement after a finite number of steps, while the bionic search may randomly circle around the local optimum without hitting it exactly.

As the ways to combine strategies are unlimited, it is difficult to give consistent proposals when to do which switching. During this study, where the main concern is about measuring the rate of the different strategies, we avoid these hybrid methods, as their inclusion would lead to a large and confusing set of combinations and corresponding accelerations of the optimisation process.

3. Application of Bionic Optimisation Strategies

The sketch of the approaches in section 2.3 needs to be improved to allow a qualified understanding of the studies, so a list of definitions and explanations seems to be inevitable. We want to remember that there are different schools in bionic optimisation using different terminology. Nevertheless, the definitions given here are generally accepted ones.

- **Individuals** are the different elements of the parents and kids sets.
- **Generation** is one step in the evolutionary process. It is given by a set of parents or individuals. The creation of a new set of kids or new individuals defines the genesis of a new generation
- **Mutation** is the modification of the parameter values of an individual. Mutation may happen in many ways [2, 3, 4]. Every parameter may be changed by a random value, some parameters may be changed in
some correlated way or only a part of the parameters is changed in every generation. There are infinite possibilities to do mutations, so the different types of mutation have to be checked carefully, until some experience is collected. During this study we mutate all free parameters by the same mutation principle. We add the mutation radius of each parameter weighted by a random number \(-1 \leq r \leq 1\) to the initial value of the parameter.

- The mutation radius is the maximum amount that a parameter may be changed in a mutation step. This value may be the same for all parameters or different for some or all parameters. Often it is not given as an absolute value but as the percentage of the total allowed range of a parameter. We here use the same relative mutation radius for all parameters during a set of generations. During the course of a study we sometimes reduce this mutation radius to better approach the best solutions.

- There are many other terms used in conjunction with bionic optimisation. As there is no generally accepted vocabulary, users are advised to check carefully the meaning when reading papers from different authors [2, 3].

3.1 Evolutionary Optimisation (ES), Terms and Variables

Evolutionary optimisation [2, 3, 4, 5] copies the way bionic beings reproduce and adapt to a changing environment. Basically two parents have one offspring by combining the properties of their DNA. This crossed DNA is subject to some random modification, the mutation. As the children within a reproducing population are never identical, some of them will be better suited to adapt to the environmental challenges. Their chance to survive is superior to their siblings, so their genetic code may become dominant within the population. Therefore, these better children will be the next generation’s parents.

The most important terms in evolutionary optimisation are:

- The number of parents should be sufficiently large to cover some or many possible parameter combinations. Its value should generally not fall short 0.5-5 times the number of parameters.
- The number of kids covers the parameter space. So again a large number is to be preferred. Some experience, which may not be applicable in every case, proposes the number of kids to be 2-5 times the number of parents.
- Pairing is the selection of two individuals of the parent generation to produce one common child.
- Should the parents survive to be parents in the next generation as well or not? Dominant parents reduce the chances of weaker children to find their way, but excluding them removes the best solutions found up to now.
- Crossing, the way by which two parents define the properties of one common child, may happen in different ways. One of the first ideas is to average the parameter values of both parents. Another type of crossing could be taking randomly one parameter from one parent only. Some weighted average of the two parents’ data, e.g. preferring the better parent, would be a possibility too [3, 4].
- Selection determines which kids of the current generation (including their parents or not) should be the parents of the next generation. Often selection is done by only taking the best \(n_{parent}\) kids as new parents. Some experience leads to proposals that at least some not so good kids should be parents as well. For the sake of simplicity and comparability here we allow the parents to be parents in the next generation and take the best of the total of old parents and kids to be parents in the next generation.

One big advantage of ES is the fact that it has the tendency to converge to the best solution if there are sufficiently large numbers of parents, kids and generations and if the mutation radius is rather large. This advantage is relative, as the number of individuals to be studied may become very large, even if the values driving the process are set in a favourable range.
3.2 Ferns Optimisation (FS)

The fern optimisation (FS) is seldom mentioned as a special kind of bionic optimisation, but has the potential of producing very good results in very short times if good initial proposals are available. The term *fern optimisation* is derived from the fact that ferns and germane plants mostly do not reproduce by crossing of two parents but by spores of one single parent [6]. These spores do not have the same genetic properties as the parent. A certain mutation happens like in all duplication of DNA.

For bionic optimisation we use the term in the following way.

- We define a limited number of initial designs, the parents, given by specific sets of free parameters $p_1, p_2, ..., p_n$.
- Each parent has a certain number of kids, which are defined by a random mutation of their parents’ parameters.
- The way of defining this mutation like in ES is not unique. There might be a change of all parameters or only a subset using a constant mutation radius or a specific one. We use the same relative mutation radius for all parameters. This mutation radius again may decrease during the course of the study.
- From the set of kids of a parent including the parent as well, the best one in terms of the objective is supposed to be the next generation’s parent.
- This process is repeated for a certain number of generations.
- After some generations we check if the offspring of the different initial parents fail to come up with relatively good results. We then remove those from the total population who seem to be not so fast in the current of the optimisation process (Fig. 8). This accelerates the process essentially, as the number of individuals to be studied decreases strongly. On the other hand, it may happen, that a sequence of individuals has the potential to find very good solutions but is excluded as it does not perform well at the beginning of the process.

![Fig. 8 Exclude slowly improving ferns](image)

- The best individual of all offspring of all initial parents over the sequence of generations is the proposed optimum of the process.

The success of FS depends on the range of the solution space that the initial designs cover. With increasing dimension of the solution space it becomes less probable that the coverage of the initial parts is sufficient to find very good designs. But for smaller dimensions, FS is able to provide interesting results by a method that is very easy to implement.

3.3 Particle Swarm Optimisation (PSO)

Particle swarm optimisation (PSO, [7, 8, 9, 10]) follows the observation that groups of living beings sometimes have a tendency to behave like a complex being itself. Well known examples are the swarms of fish or birds that seem to be orchestrated by a common intellect. The assumption is that they would not do so, if there was not an advantage compared to the free individual swimming or flying, which may be observed as well. We don’t want to discuss, what this gain of coordinated behaviour might be, we want explain how to translate it to an optimisation strategy and check its efficiency. The basic assumption of particle swarm optimisation is that the individuals know their position and velocity. In addition they know where the individual’s best position during the process has been, and where the best position in the parameter space found up to now is located (Fig. 3). The process of optimisation may then be defined by the steps
Remarks on the Efficiency of Bionic Optimisation Strategies

- Each individual tends to continue the way it is travelling, given by its individual velocity: It has an inertial moment.
- Each individual tries to return to the best position it ever assumed, called the cognitive tendency.
- Each individual tries to approach the best position of all individuals found up to now, called the social tendency.
- Both of these tendencies are weighted by a constant $c$ and a random vector $|r| \leq 1$ and added to the inertia vector, which also is weighted by a scalar value

$$v_{\text{new}} = c_r v_{\text{old}} + c_c r_{\text{cognitive}} \cdot (x_{\text{best\_individual}} - x_{\text{old}}) + c_s r_{\text{social}} \cdot (x_{\text{best\_swarm}} - x_{\text{old}})$$

where the product $r \circ x$ is supposed to be the vector produced by the product of the 2 vectors components, the Hadamard product.

- This vector $v_{\text{new}}$ is going to be the individual’s velocity in the next step, which corresponds to the generations in the other approaches yielding

$$x_{\text{new}} = x_{\text{old}} + v_{\text{new}}.$$

PSO has proven to be very successful if an appropriate set of particles and velocity weighting factors $\{c_v, c_c, c_s\}$ has been selected. Unfortunately PSO has the tendency to stick to local minima, if the parameters are not well chosen. Fig. 9a compares the results found at test example F3 of section 4.1 for different weighting factors of the inertia and the social component of the new velocity. We see a valley of efficient combinations of the velocity weighting factors. This valley is limited by steep hills indicating less efficient progress and weaker goals achieved by the PSO studies. Limiting the particle’s velocity to a certain maximum value (Fig. 9b) has the potential to reduce the height of the valley’s side walls, but reduces the convergence of the optimisation process.

3.4 Artificial Neural net Optimisation

Artificial neural nets (ANNs) are combinations of neurons like those found in the brain of an animal or human being. The central function for our purpose is that each neuron has a number of inputs and one output ([11, 12, 13, 14, 15], Fig 10a). For each neuron, e.g. the neuron numbered $k$, the input signals $I_{i,k}$ are weighted by weights $w_{i,k}$ summed and then centred by an additional value $w_k$.

$$I_k = \sum_i w_{i,k} I_{i,k} - w_k$$

The output of this weighted, summed and centred input is given by a smooth transfer function which is zero for small values of the input and 1 for large input values (Fig. 10b), e.g. a variant of the sigmoid function

$$\text{out}_k = \text{sig}_{\nu_k}(I_k) = \frac{1}{1 + e^{-\nu_k I_k}}$$

Lagaros [14] discusses the efficiency of some
variants of the sigmoid function. The response of a neuron is given by

\[ out_k = \sigma(\sum_i w_{ik} I_{i,k} - w_k) \]  

or the corresponding value if we use another transfer function. Here again \( w_k \) is used to shift the function to an interesting region.

The neurons used in ANN are grouped in three different types (Fig. 10c),
- the input layer, which receives the information of the environment,
- one or more hidden layers, which combine the received input and their specific weighting factors to produce an output which will be part of the input of the next layer and
- the output layer, which is fed by the output of the last hidden layer. Its values present the response to the input signal.

In most applications of ANN only one or two hidden layers are used, as they suffice to represent even very non-regular outputs. Fig.10c presents the scheme. Here two inputs are read by the ten neurons of the first hidden layer weighted and transformed into an output signal for each neuron. The ten neurons of the second hidden layer collect the output of the first hidden layer, do another weighting and send a signal to the output layer. The output neuron does a third weighting and finally produces the single output value.

We only use such a simple design of an ANN. Many variants of it have been discussed and tested for their applicability in different fields [15].

The determination of the weighting coefficients of such an ANN often is called the training. For a sufficiently large set of \( m \) training examples \( m > |\{ w_i \}| \) with specified input and known corresponding output the minimum of the accumulated error is sought

\[ err = \sum_{n=1}^m (out_{exact,n} - out_{ANN,n})^2 \]  

where \( out_{exact,n} \) are the known results of the training cases and \( out_{ANN,n} \) depend on the weighting coefficients \( w_i \) and \( w_k \). So the minimisation of \( err \) by varying the weighting factors is an overlying optimisation or minimisation process. This training process may consume essential amounts of time and computer power. Following Widmann’s [15] proposals we use the quickprop method [16] to do this minimisation.

Even if the amount of computation to qualify a neural net may be large, it is a powerful tool if there are many function evaluations to be done and a single evaluation requires much computing power itself. The ANN then acts as a meta-model which is easily capable of producing the required output with a known exactness.

### 3.5 Response Surfaces

Response surfaces (RS) as already introduced in section 2.2 may be looked at as a simple modification...
Remarks on the Efficiency of Bionic Optimisation Strategies

of neural nets. Instead of training the net, the space of free parameters near a promising set of parameters is covered by a sufficient number of individuals. These individuals could be randomly placed, cover some regular grid or a sub-grid given for example by using a Latin hyper cube scheme. If we want to evaluate the objective at a distinct position, we build a low-degree polynomial surface by the data close to the point of interest (cf. eq. (3)) and find the position of the optimum by eq. (4).

Optimisation of larger problems then may be done by studying the regions where the objective shows promising values. One essential difference between ANN and RS consist of the fact, that the neural net covers the total of the solution space (Fig. 4), while the response surface approximates the objective only in some limited neighbourhood of the evaluation points (Fig. 5). Like neural nets, response surfaces prove to be very efficient if after an initial set of runs, the value of the objective function has to be found many times and each evaluation is relatively time consuming.

Applied to our test problems, RS might be used by many repeated runs using random or qualified initial data input. Then there is a chance that good proposals are found near this initial point. The performance of RS compares to that of the FS but for larger number of parameters it becomes slower as we have to do more function evaluations than to produce kids in the case of FS.

Gradient methods could be used in the same way. We use some starting points and find the next local optimum. If we are lucky and use a sufficient large number of initial points there is a real chance to hit the absolute optimum. As both of these approaches do not fit well into the class of bionic optimisation we do not discuss them in more detail.

4. Comparing the Efficiency of Bionic Optimisation Strategies

As optimisation is an expensive and time consuming process, the check of the efficiency may yield an understanding which procedure may lead to a good and acceptable result within a reasonable time. To compare the efficiency of the approaches we use five different structural problems.

4.1 Structural Test Examples

The five test examples are shown in Fig. 11. The first (F1) is a simple frame structure with 6 rods and 5 nodes loaded by a force at 2 nodes. The second one (F2) has been presented by Plevris [10] and others. 10 rods are given by 6 nodes. The system is loaded as indicated. Example F3 presents a bridge by 13 rods connected at 8 nodes and loaded by 3 different forces. For example F4 58 rods are joined at 27 nodes, for example F5 193 rods are joined at 80 nodes, the loading is indicated in Fig. 11 as well. The grid size of the examples is 1000 mm except for example 2 where the grid size is 360 inch. Table 1 summarises the input, the free parameters and the restrictions of the models. It should be mentioned that example F2 is defined in imperial units (in, kip) while the other 4 use mm and Newton. The goal of the

![Fig. 11 Frames used as test examples](image-url)
Remarks on the Efficiency of Bionic Optimisation Strategies

### Table 1  Data of test Problems

<table>
<thead>
<tr>
<th>Frame</th>
<th>params</th>
<th>grid size</th>
<th>Amax / Amin</th>
<th>E-Mod</th>
<th>σmax</th>
<th>d max</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>6</td>
<td>1000 mm</td>
<td>600 / 20 mm²</td>
<td>200 GPa</td>
<td>120 MPa</td>
<td>0.5 mm</td>
</tr>
<tr>
<td>F2</td>
<td>10</td>
<td>360 inch</td>
<td>35 / 0.1 inch²</td>
<td>10 Msi</td>
<td>25 ksi</td>
<td>2.0 inch</td>
</tr>
<tr>
<td>F3</td>
<td>13</td>
<td>1000 mm</td>
<td>400 / 20 mm²</td>
<td>200 GPa</td>
<td>50 MPa</td>
<td>0.5 mm</td>
</tr>
<tr>
<td>F4</td>
<td>58</td>
<td>1000 mm</td>
<td>400 / 20 mm²</td>
<td>200 GPa</td>
<td>100 MPa</td>
<td>2.0 mm</td>
</tr>
<tr>
<td>F5</td>
<td>193</td>
<td>1000 mm</td>
<td>600 / 20 mm²</td>
<td>200 GPa</td>
<td>450 MPa</td>
<td>20 mm</td>
</tr>
</tbody>
</table>

**params:** # of rods in frame  
**grid size:** horizontal or vertical distance between the nodes  
**Amax, Amin:** maximum and minimum allowed cross section area of the rods  
**E-Mod:** Young’s modulus  
**σmax:** maximum allowed stress in rod  
**d max:** maximum allowed displacement of nodes

optimisation was to minimise the mass by variation of the rods’ cross sections without violation of the restrictions on stress and displacements.

To come up with comparable results, we did a series of 20 loops to avoid having only one or few very good or very bad results. On the other hand, the parameters we use are based on some experience with the underlying problems, so the number of runs presented does not come from a naïve starting procedure, but includes some preliminary work which is impossible to quantify.

### 4.2 Input and Results of the Frame Test Examples

Table 2 lists the inputs of the test runs used. The input configurations are based on some experience, which reduces the power of the results. On the other hand, it is difficult to ask users without any experience to do such studies. So we are aware that the number of runs to achieve the data listed in the table is only one part of the total number of runs. For ANN (Table 2.4) and RS (Table 2.5) the results of the optimisation are given as well.

Table 3 summarizes the results of the test runs for ES, FS and PSO. In addition we add the results of random searches (Table 4), which do not perform very well as soon as the number of free parameters is not very small.

The most important data are the number of individuals analysed to find a sufficient good design. As all jobs had been repeated 20 times, the best of all 20 loops the average and the standard deviation are given. In Table 3 the ratio reldev of the difference between the best and the average number of individuals compared to the standard-deviation gives an idea of the stability of the strategy. Fig. 12a summarizes the data. Here we plot the number of individuals of the 20 jobs divided by 20 to give an idea of the number of individuals per study. ES, FS and PSO prove to be of a comparable efficiency when applied to the four smaller problems (F1, F2, F3, F4, Fig. 12b). For the largest problem F5 FS shows a performance that is essentially weaker than ES and PSO. RS and random perform well for the smallest problem F1, but need essentially larger numbers of individuals for the other problems. For random searches, the data plotted in Fig. 12 depict the number of individuals to find an acceptable result. For ANN we should remember that the performance we measure in number of individuals is not a sufficient value. An essential part of the computing time is spent on the determination of the weighting factors $w_{ij,k}$. We had to increase the number of the individuals by this computing time which we express in a corresponding number of individuals. So ANNs are not competitive even for the smaller problems. For the larger models we learned that ANN, RS and random searches do not perform sufficiently fast, so we do not list the results here.

### 4.3 Tendencies to be Observed

The primary result derived from Table 3 and Fig. 12 is that all the strategies except ANN and RS show comparable potential to solve optimisation problems –
Remarks on the Efficiency of Bionic Optimisation Strategies

Table 2 Input Parameters used

2.1 Evolutionary strategy

<table>
<thead>
<tr>
<th>model</th>
<th>parents</th>
<th>kids</th>
<th>mut.rad.max</th>
<th>mut.Rad.min</th>
<th>generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>10</td>
<td>20</td>
<td>.5</td>
<td>.05</td>
<td>60</td>
</tr>
<tr>
<td>F2</td>
<td>5</td>
<td>10</td>
<td>.5</td>
<td>.05</td>
<td>40</td>
</tr>
<tr>
<td>F3</td>
<td>5</td>
<td>10</td>
<td>.5</td>
<td>.05</td>
<td>50</td>
</tr>
<tr>
<td>F4</td>
<td>50</td>
<td>100</td>
<td>.5</td>
<td>.05</td>
<td>100</td>
</tr>
<tr>
<td>F5</td>
<td>100</td>
<td>200</td>
<td>.5</td>
<td>.05</td>
<td>200</td>
</tr>
</tbody>
</table>

2.2 Fern strategy

<table>
<thead>
<tr>
<th>model</th>
<th>parents</th>
<th>kids/par.</th>
<th>mut.rad.max</th>
<th>mut.Rad.min</th>
<th>generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>10</td>
<td>5</td>
<td>.5</td>
<td>.05</td>
<td>100</td>
</tr>
<tr>
<td>F2</td>
<td>10</td>
<td>4</td>
<td>.5</td>
<td>.05</td>
<td>50</td>
</tr>
<tr>
<td>F3</td>
<td>20</td>
<td>5</td>
<td>.5</td>
<td>.05</td>
<td>100</td>
</tr>
<tr>
<td>F4</td>
<td>100</td>
<td>5</td>
<td>.5</td>
<td>.05</td>
<td>200</td>
</tr>
<tr>
<td>F5</td>
<td>200</td>
<td>5</td>
<td>.5</td>
<td>.05</td>
<td>200</td>
</tr>
</tbody>
</table>

Mutation radius reduced

- 0% - 25% of generations: \( r_{mut} = .50 \)
- 25% - 50% of generations: \( r_{mut} = .20 \)
- 50% - 75% of generations: \( r_{mut} = .10 \)
- 75% - 100% of generations: \( r_{mut} = .05 \)

2.3 Particle Swarm Optimisation

<table>
<thead>
<tr>
<th>model</th>
<th>particles</th>
<th>generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>10</td>
<td>30</td>
</tr>
<tr>
<td>F2</td>
<td>10</td>
<td>40</td>
</tr>
<tr>
<td>F3</td>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>F4</td>
<td>50</td>
<td>200</td>
</tr>
<tr>
<td>F5</td>
<td>800</td>
<td>70</td>
</tr>
</tbody>
</table>

Weighing factors:

\( c_v = 0.08, c_{sign} = 0.005, c_{soc} = 2.0 \)

2.4 Artificial Neural nets

<table>
<thead>
<tr>
<th>ANN</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td># learn ind</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td># test ind</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Time training</td>
<td>600</td>
<td>925</td>
<td>830</td>
</tr>
<tr>
<td># train ind</td>
<td>800 000</td>
<td>1 233 000</td>
<td>1 107 000</td>
</tr>
</tbody>
</table>

Time training: training + testing time in sec

# train ind: corresponding # of individuals to training time divided by the 5 loops done

Time per individual was less than .15 msec

2.5 Response Surfaces

<table>
<thead>
<tr>
<th>RS</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td># of individuals / run</td>
<td>12</td>
<td>20</td>
<td>26</td>
</tr>
<tr>
<td># of loops</td>
<td>1000</td>
<td>4000</td>
<td>4000</td>
</tr>
<tr>
<td>best of loops</td>
<td>1.84E+06</td>
<td>5.26E+04</td>
<td>2.63E+06</td>
</tr>
<tr>
<td># of individuals to best</td>
<td>120 000</td>
<td>80 000</td>
<td>104 000</td>
</tr>
</tbody>
</table>

# of individuals / run: \( 2 \ast \# \) of free parameters

# of loops: \# of random initial designs

Goal measured in mm\(^3\) except F2: in\(^3\)

at least at studies with a small number of degrees of freedom. The most convincing result is that ES and PSO seem to be of comparable power when applied to the problem class we discuss. FS shows promising results if the number of parameters is not too large, but becomes less successful in higher dimensional spaces. ANNs lead to interesting results for smaller numbers of free parameters as well, if the results are used for many evaluations. Fig. 12 indicates that for ES, FS and PSO there might be a nearly linear relation between the number of free parameters and the individuals to be studied. Even if the number of runs becomes very large, the methods are superior to pure random search, as we learn from Table 4, where e.g. for example F4 20 loops with 100 000 individual have been producing proposals that are essentially inferior to the ones found by bionic approaches especially for the problems with larger numbers of free parameters.
Table 3  Results of 20 optimisation runs per problem

<table>
<thead>
<tr>
<th>strategy</th>
<th>model</th>
<th>mean</th>
<th>stddev</th>
<th>best</th>
<th>reldev</th>
<th>Individuals [1000]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES</td>
<td>F1</td>
<td>1.62e6</td>
<td>.716e3</td>
<td>1.62e6</td>
<td>1.50</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>6.33e4</td>
<td>4.50e3</td>
<td>5.47e4</td>
<td>1.90</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>2.56e6</td>
<td>6.99e4</td>
<td>2.48e6</td>
<td>1.11</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>F4</td>
<td>1.03e7</td>
<td>4.18e5</td>
<td>8.65e6</td>
<td>3.92</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>F5</td>
<td>1.98e7</td>
<td>1.07e6</td>
<td>1.58e7</td>
<td>3.65</td>
<td>800</td>
</tr>
<tr>
<td>FS</td>
<td>F1</td>
<td>1.66e6</td>
<td>4.49e4</td>
<td>1.62e6</td>
<td>0.81</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>6.39e4</td>
<td>4.43e3</td>
<td>5.45e4</td>
<td>2.09</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>2.50e6</td>
<td>2.29e4</td>
<td>2.47e6</td>
<td>1.19</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>F4</td>
<td>9.91e6</td>
<td>2.77e5</td>
<td>9.39e6</td>
<td>1.86</td>
<td>189</td>
</tr>
<tr>
<td>PSO</td>
<td>F1</td>
<td>1.65e6</td>
<td>1.71e4</td>
<td>1.62e6</td>
<td>1.61</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>5.87e4</td>
<td>5.61e3</td>
<td>5.15e4</td>
<td>1.27</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>2.50e6</td>
<td>2.53e4</td>
<td>2.48e6</td>
<td>1.02</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>F4</td>
<td>8.90e6</td>
<td>1.68e5</td>
<td>8.68e6</td>
<td>1.22</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>F5</td>
<td>1.54e7</td>
<td>0.18e4</td>
<td>1.53e7</td>
<td>1.70</td>
<td>1120</td>
</tr>
</tbody>
</table>

Goal measured in mm$^3$ except F2: in m$^3$

Table 4  Results of random jobs

<table>
<thead>
<tr>
<th>strategy</th>
<th>model</th>
<th>mean</th>
<th>stddev</th>
<th>best</th>
<th>reldev</th>
<th>Individuals [1000]</th>
</tr>
</thead>
<tbody>
<tr>
<td>rand</td>
<td>F1</td>
<td>1.74e6</td>
<td>3.26e4</td>
<td>1.66e6</td>
<td>2.58</td>
<td>2 000</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>6.42e4</td>
<td>2.47e3</td>
<td>6.03e4</td>
<td>1.61</td>
<td>2 000</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>2.86e6</td>
<td>4.28e4</td>
<td>2.77e6</td>
<td>2.12</td>
<td>2 000</td>
</tr>
<tr>
<td></td>
<td>F4</td>
<td>1.44e7</td>
<td>2.76e5</td>
<td>1.38e7</td>
<td>2.20</td>
<td>2 000</td>
</tr>
<tr>
<td></td>
<td>F5</td>
<td>4.12e7</td>
<td>5.50e5</td>
<td>4.02e7</td>
<td>1.96</td>
<td>2 000</td>
</tr>
</tbody>
</table>

Goal measured in mm$^3$ except F2: in m$^3$

4.4 Interpretation of the Results

Some knowledge may be gleaned from the results of this series of studies, foremost that optimisation, especially bionic optimisation, is a process that consumes large amounts of time and computing power. Furthermore, the results presented in section 4.2 and 4.3 would not have been found without a large number of preliminary studies which provided us with much experience in the field of optimisation.

Fig. 13 depicts the importance of good initial designs for example F1. If we start with proposals that are good, e.g. have an objective of $z = 1.75 \times 10^6$ we may save...
all the generations required to come down from the random designs to the good ones, which may be more than 50% of the total optimisation job.

All the input characteristics used in the test runs are derived from many preliminary studies. So e.g. the selection of the three weighting factors \((c_v, c_g, c_s)\) for the PSO required some 100 000 runs. The proposal to reduce the mutation range for ES and FS is the result of many studies done before the final loop which entered the results of section 4.2. The proposal to use a number of initial parents as large as the number of free variables for ES and PSO is based on many pre-studies, like the idea to use a large number of initial parents and a small number of kids in FS.

The fascinating strategy of ANN proves to be very efficient in some smaller examples. But as the number of weighting factors \(w_{ik}\) increases with the number of hidden neurons, the time to minimize the error (eq. 10) becomes larger than the time to evaluate the individuals. So ANN seems to be limited to cases smaller numbers of free parameters have to be taken into account. Successful applications are found if the response surface defined by the ANN is to be used many times after it has been initially established. These are applications where the trained configurations define a meta-model which helps to determine responses of parameter combinations without doing a new study.

Response surfaces prove to be a powerful tool if simple landscapes have to be studied many times. So if we are interested in the evaluation of a small region of the solution space (cf. Fig. 5), where the objective may be represented by a surface of low polynomial degree, RS may be more efficient than ANN and used successfully. The way we use RS is to define many starting points, evaluate the objective around these points like FS and with some luck find a good solution. The performance of this approach is only satisfactory in the case of small numbers of free parameters. Gradient methods may be used in a similar way and show a performance which is comparable with the one of RS.

5. Conclusions

As a general remark on the efficiency of optimisation strategies we want to state that the quality of the initial proposals is the most important component of any optimisation. If experienced and motivated designers propose designs that are close to optimal ones there are chances to find at least local optima which are not too far away from the best solution possible. If we are close to good proposals, gradient methods, fern studies with small mutation radii or response surfaces will improve the parameters in short time and at reasonable effort.

As soon as we doubt that our initial designs are close to the optimal ones or that there are good solutions in parts of the parameter space we did not yet take into account, evolutionary optimisation or particle swarm optimisation have the potential to give reasonable tendencies about the possibilities of better proposals. Nevertheless the number of function evaluations becomes large, so that for computing power demanding problems only strong parallel processing is able to limit the total time to find interesting solutions. Which of the two, ES or PSO is to be preferred may not be decided without some preliminary tests. Often the particle swarm shows a faster tendency to the assumed best values, but some examples indicate that depending on the chosen parameters \(\{c_v, c_g, c_s\}\) PSO might have the tendency to stick to local maxima like gradient
Remarks on the Efficiency of Bionic Optimisation Strategies

Using hybrid methods or switching to gradient optimisation if a maximum is approached closely is always an interesting option. But experience has to be gathered there as well.

Neural nets and response surfaces are powerful tools, if a given problem has to be analysed many times and the approximation by the ANN or the RS yields acceptable values in acceptable time. This holds especially, if the function evaluation is expensive like in the case of a large non-linear finite element study of a complex problem. There ANN or RS may be used as meta-models that allow fast access to the desired function values. For problems where the optimisation is done and the result found will never be questioned again, the large number of training jobs seems to be somehow prohibitive.

In every case, optimisation of large problems is a time and resources-consuming process. There is no way to avoid the evaluation of many individual solutions, and there is no guarantee that the absolute best solution will be found at all.

Acknowledgment

Part of this work has been supported by the federal government and some companies we have been working with. We gratefully acknowledge their financial and engineering support.

References