# Understanding the Course and State of Evolutionary Optimizations using Visualization

**Hartmut Pohlheim**

DaimlerChrysler AG, Research and Technology
Alt-Moabit 96a, 10559 Berlin, Germany

Hartmut.Pohlheim@DaimlerChrysler.com

phone: +49 30 39982 456
fax: +49 30 39982 107

10 Years of Industry Experience with Evolutionary Algorithms

category: research paper

## Abstract

Evolutionary algorithms (EA) are widely employed to solve a broad range of optimization problems. Even though they work in an algorithmically simple manner, it is not always easy to understand what is going on during a particular optimization run. It is especially desirable to gain further insight into the state and course of the algorithm if the optimization does not yield the expected results or if we are not sure whether the result achieved is really the best result possible.

During an optimization run an evolutionary algorithm produces a vast amount of data. The extraction of useful information is a non-trivial task. In this paper, we review visualization methods used to extract this useful information. We shall also demonstrate the application of visualization techniques and explain how they help us to understand the course and state of the evolutionary algorithm.

This extra information gained by the use of visualization techniques is often the difference between a good result and a very good result. In complex real-world applications merely achieving a good result often means that the approach has failed. On the other hand, a success means large gains in productivity or safety or a decrease in costs.



Keywords: 2-D visualization, 3-D visualization, image plot, scaling, convergence, real-world application

# 1 Introduction

A new optimization problem must be solved. Not much is really known about the properties of the system under optimization. For several reasons the decision is made to use evolutionary algorithms (EA). The objective function implementing the optimization problem is written and one of the many toolboxes implementing evolutionary algorithms is selected. The operators and parameters of the evolutionary algorithm are defined (based on experience or using a standard set suitable for the system under optimization). The evolutionary algorithm is started and quite a few numbers scroll over the screen. After some time a result is presented. The result is checked and looked at in the context of the system. There might be a chance that the result is satisfactory first time round – that it is much better than anything known before or than anyone expected. What a stroke of luck! It seems to have been a simple problem that nobody had thought about or tried to optimize before.

But this is real life. We never get such simple problems to solve and one optimization run is never enough. The numbers presented on the screen are never sufficient to really understand the way the evolutionary algorithm "solved" the problem – never sufficient to even know the inner workings of the evolutionary algorithm used. In the beginning the optimization was simply restarted. Did we get better results, were they different or the same? We obtained some new information – but did it really help? There was such a vast amount of data produced by the evolutionary algorithm, that we had to find methods of extracting useful information from this data. This is a non-trivial task.

The visualization of the data produced by the evolutionary algorithm can provide the insight necessary to understand the result(s). However, which data types over which time frame and using which visualization methods should be plotted?

Over the last ten years we have employed a large number of different visualization techniques to present the data types produced by evolutionary algorithms. Some of them have proved indispensable nearly all the time (and will be presented in this paper). A few others were useful under special circumstances or to help understand a very special feature (some will be outlined in this paper). Many more produced colorful graphics, but did not provide additional or better insight compared to the visualization techniques already in use (some of these are discussed in [8]).

In this paper, we will present a set of techniques for the visualization of different data types of evolutionary algorithms. These data types and techniques give an instant visual impression of the evolutionary algorithm's progress or the actual state of the individuals within the population. In this way, we also gain insight into the problem which has to be solved. We understand the problem better, learn more about its properties relevant to optimization and gain more experience in the application of evolutionary algorithms for the solution of this type of problem. For each technique, examples are given and the information derived is described. Most of the visualization techniques are simple to calculate and easy to display using standard visualization tools. Nevertheless, we shall provide some additional information on the real-world application of the techniques (having learned these fiddly details the hard way).

Previous work on the visualization of evolutionary algorithms is still rare. Apart from standard techniques most of the early work was published by Routen and Collins ([11], [12], [2], [3]). A systematic approach to the visualization of evolutionary algorithms was set out in [13] and in [8], chapter 5, but further (systematic) work still has to be done. The first workshop on the visualization of evolutionary algorithms was organized at GECCO'1999 [5]. Another workshop with a broader range was held at ALife VIII [1]. However, these are singular events.

If we look at all these publications we can see that a lot of techniques for visualizing evolutionary algorithms have been published. But only very few of them are employed regularly by the users of evolutionary algorithms. Thus, we also need to describe the advantages of the visualization techniques for "every-day" and "always-on" use. In this paper we will provide some of these descriptions including easily understandable examples.

Section 2 provides a compact systematization of the data types and time frames in evolutionary algorithms. Each of the following sections describes one particular visualization technique. Section 9 ends with concluding remarks and some possible directions of further investigations.

A few conventions are used throughout the paper. All the optimizations use minimization. We mostly work with a real-valued, fixed length representation during optimization (most of the real-world problems we, at DaimlerChrysler, solved are real-valued and fixed length). Many of the example diagrams are generated by optimizing one of the many standard test functions of evolutionary algorithms (and other optimization areas).

In the examples we employ two types of evolutionary algorithm, one which is globally and one which is locally oriented. The first one employs multiple subpopulations (regional population model, also called island model or coarse grained model), migration, and the application of different strategies and competition between subpopulations. In this paper it is called the Multi-Population Evolutionary Algorithm MPEA1. The second type of evolutionary algorithm employs a small population and a mutation operator similar to an evolution strategy. This is called LEA2. The first type, MPEA1, performs a globally oriented search and is the optimization algorithm which we use first for each new problem. Sometimes we are given problems suitable for a locally oriented search (no multi modalities, no discontinuities, ...) and we employ LEA2 for this kind of problem. In this paper we shall only look at single-objective problems.

## 2 Data types and time frames of evolutionary algorithms for visualization

When we want to extract useful information from the data produced by the evolutionary algorithm we must first understand which data types are available. In this paper we mostly concentrate on directly available data types: individuals and populations (including subpopulations). Each individual contains multiple variables and one or multiple objective values. Populations contain multiple individuals each containing variables and objective values. In a population we can look at single individuals (best or mean individual of the population) or at multiple or all individuals. In addition, we may also include data available from the use of advanced techniques. As an example we shall use the application of multiple subpopulations (regional population model including migration and competition). Figure 1 presents an overview of the selection of data types available for the visualization of evolutionary algorithms.

- individuals (solution vector)
  - → variables (real-valued, integer-valued or binary representation)
  - → objective value(s)
- (sub) population
  - → individuals (variables of best or all individuals, objective values of best or all individuals)
  - → ranking / order and size of subpopulation
  - → derived data types, for instance distance between individuals

Figure 1: Data types of evolutionary algorithms for visualization (selection)

The data types are only one side of the equation. We must also look at the time frame of the available data. In evolutionary algorithms the time frame of one or multiple generations is directly available. We can use up to all the generations of one optimization run or extend the time frame to the data produced by multiple optimization runs. These different time frames are shown in Figure 2. When visualizing data produced during many generations, a picture of the progress or course of the evolutionary algorithm is presented. Techniques employing only data produced anew for every generation give a picture of the current state of the population or the evolutionary algorithm. Using data from multiple optimizations runs we can compare different evolutionary algorithms or changes in their parameters.

- different time frame
  - → one generation → state of EA
  - → multiple/all generations → course of EA
  - → multiple runs → comparison of EA

Figure 2: Time frames of evolutionary algorithms for visualization (selection)

## 3 How does the best result get even better during optimization?

If any one visualization method is used, it is almost always the convergence diagram. It presents the objective value of the best individual in the population over many generations (mostly all the generations of one optimization run). A simple example is given in Figure 3 left. This type of diagram gives a good impression of the convergence of the evolutionary algorithm.

The objective values are problem-specific and their range varies with every optimization problem. The clarity of the diagram depends largely on the scaling of the objective values. On the other hand, the user of the evolutionary algorithm "understands" these problem-specific values. What are the problems with the convergence diagram and how can we circumvent them?

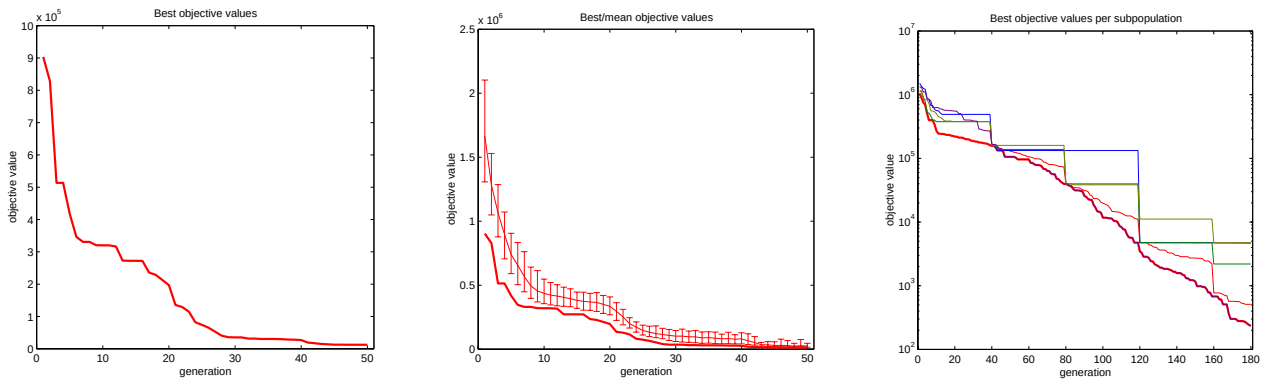


Figure 3: Convergence diagram; left: objective value of best individual, middle: best and mean objective value including standard deviation of all objective values, right: logarithmic scaling of best objective value (including best objective value of every subpopulation); [sphere function, MPEA1]

At the beginning of an optimization run the objective values (often) improve quickly. During the remaining generations the changes in the best objective values are not that large. Thus, the important changes at the end of the optimization cannot really be seen. **Logarithmic scaling** of the objective values (y-axis of the diagram) can provide a solution. When employing loga-

rithmic scaling, the objective values must always be positive (or to be more exact: not change their sign) during an optimization run. The diagram in Figure 3 right, presents a run similar to Figure 3 left. By using logarithmic scaling it is possible to clearly see changes in the best objective values over a much longer time frame (180 generations compared to 50 generations).

Many publications include the mean and worst objective value in the convergence diagram. However, including the worst objective value very often destroys the scaling of the diagram (especially when using globally oriented search algorithms such as MPEA1). More information can be extracted if the **mean objective value and the standard deviation** of all the objective values are included instead, Figure 3 middle. This provides a rough overview of the distribution of the objective values and is much better than using the worst objective value. However, we prefer other visualization techniques which provide a real picture of the distribution and values of (nearly) all the objective values (Section 5 and Section 7).

Another simple solution to the scaling problem is to visualize only a **limited time frame** of an optimization run. During an optimization we visualize only the last 20-50 generations (10%-20% of the maximal number of generations) presenting the most interesting time frame. This ensures useful scaling (even without logarithmic scaling). After an optimization, we review the optimization results by "zooming" into interesting parts of the optimization. Figure 3 left and middle are examples of this selection of a useful and understandable time frame.

At the end of the description of the convergence diagram we would like to present an extension which is useful when employing a special feature in the evolutionary algorithm. When the regional model is employed (MPEA1), the simple convergence plot can be extended to show the best objective value of each subpopulation, Figure 3 right. We can directly see the different behavior of the subpopulations. If all the subpopulations use the same parameters this is not really interesting. But if the subpopulations employ different strategies we are able to get a first impression of the success of the strategies during an optimization run. However, there is another data type which can display this information in a more direct way (see Section 6).

The convergence diagram is a powerful visualization technique for providing an overview of the problem-specific performance of the optimization run. Nevertheless, we only see the best of the objective values. This is only the "tip of the iceberg". The following visualization techniques start to look under the surface.

## 4 How does the best individual change during optimization?

By using the convergence diagram we are able to gain an impression of the objective value of the best individuals. On the other hand, the visualization of the variables of the best individual from every generation presents the background or basis for these objective values for the most representative diagrams of the progress of an optimization run. The user can actually see how the variables are changed and which combinations of variables are successful. During the run, the user is given a picture of good variable values. It is possible to gain a particularly good insight into the optimization results when this is combined with the convergence diagram. The diagrams in Figure 4 show this combination of variables and the objective value of the best individuals over the generations of an optimization run using three different example functions.

When examining these diagrams, we can directly see:

- whether or not large or small jumps in variable values occur during the optimization. These are an indication of existing local minima (Figure 4, top left: variables nearly always change in jumps, at the beginning in large jumps, from generation 40 onwards the jumps are quite small; Figure 4, top middle: around generation 50 and 150, variables change considerably but continuously; Figure 4, top right: only a few jumps occur, most of the variable value changes are quite small),
- where the "interesting" areas of variable values are (Figure 4, top left: all variables tend to 0; Figure 4, top middle: variables concentrate near zero but are not identical; Figure 4, top right: each variable has its own area),
- the correlation between variable values (Figure 5 left: all variables change at the same time together). This often makes an optimization much more difficult (or an evolutionary algorithm using variable mutation rates is needed, such as LEA2, see Section 1),
- the correlation between changes in variable values and changes in the objective value (Figure 4, right: the large variable change around generation 60 corresponds with a very large change in the objective value). This is especially useful, when the convergence becomes "slower". Visualizing the variable values can often provide information on whether the optimization is stuck at a local minima, or if the search is continuing.

All the diagrams in Figure 4 top used the **2-D line plot**. Each line represents the course of one variable. However, an assignment between variable number and line is not possible. We cannot see which line corresponds to a given variable. The use of a legend and different line styles and colors can solve this problem for a few variables (up to 10 variables), see Figure 4 top right.

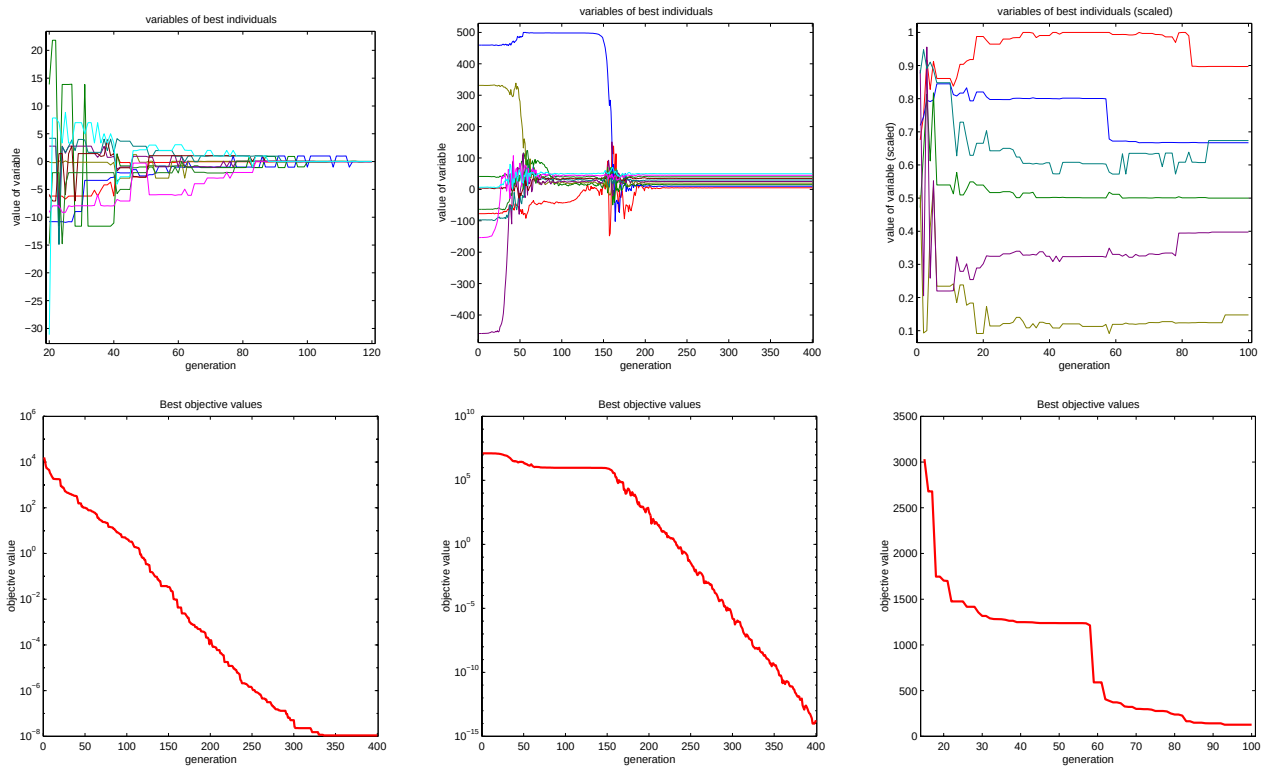


Figure 4: Combination of variables of the best individual (top row) and the respective objective values (bottom row) of many generations; left: RASTRIGIN's function 6, MPEA1; middle: moved axis parallel hyper ellipsoid, LEA2, right: lateral control of a vehicle, MPEA1

However, other techniques for presenting this data type also exist. A direct solution seems to be to use a **3-D line plot**. The 3-D line plot, Figure 5 middle, provides the extra dimension needed to visually assign every variable to a line. However, this technique works only for systems where adjacent variables have similar values, otherwise the overview becomes completely lost. In addition, it is difficult to derive a variable value from the diagram at a given generation.

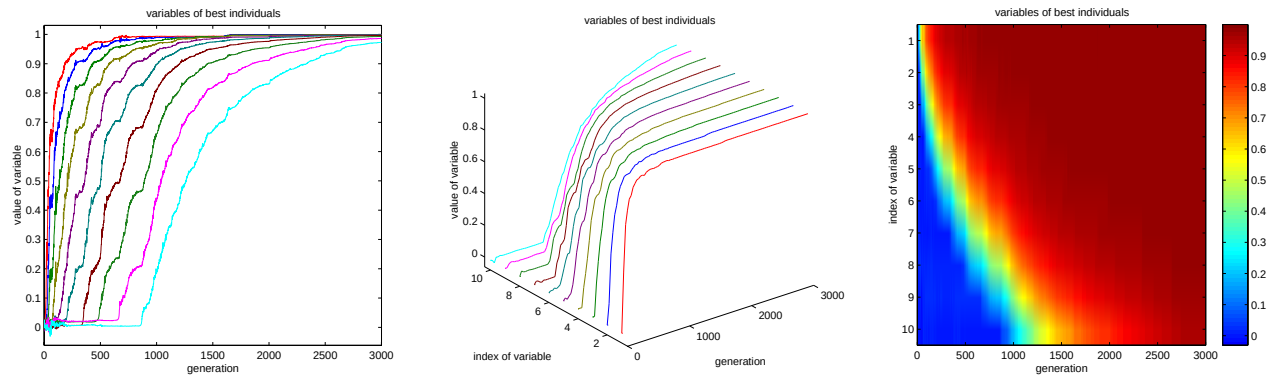


Figure 5: Visualization techniques for variables of the best individual from many generations; left: 2-D line plot; middle: 3-D line plot, right: 2-D image plot [ROSENBROOK's function 2, LEA2]

However, there is another technique which can extract all this information – the **2-D image plot**, Figure 5 right. Even for a large number of variables and different values of adjacent variables, a good impression of the distribution and change in the variables is possible. This type of diagram needs some time to get accustomed to. The use of color as an extra dimension is not common in everyday use. Nevertheless, in our opinion this is (nearly always) the best visualization technique for presenting the variables of the best individual during an optimization.

There is still one problem left. In most real-world problems each variable has its own range. To present all the variables in one diagram, we must scale the variables. Fortunately, the lower and upper variable boundaries are generally known (input of the optimization algorithm). Therefore, very different variable ranges can be circumvented by scaling the variables according to the

domain of each variable, thus visualizing the relative values of the variables. An example is presented in Figure 4 top right. This can also be used for the 2-D image plot. However, the direct information on the actual variable values is partly lost, we see just the relative value of the variable in its domain.

For all variable scaling it is important to keep a zero. Thus, scaling is performed to the range of [-1, 1]. In this way we keep the important information of a changing sign of the variables unchanged. However, if the sign of the variables is always the same, we can shrink the scaling range to the positive (or negative) part.

## 5 How can all objective values be shown at once?

The previous two data types involved only the best individual of the population. To obtain an overview of the whole population, one can visualize the objective value of all the individuals of every generation. This provides a further level of information about the progress (and state) of the optimization run.

Compared to the visualization of the objective value of "just" the best individual, we gain a real overview of the distribution of the objective values in the population. This includes the following aspects:

- which objective values are present in the population and at which time in the optimization run they are present,
- whether any "attraction" levels exist, in which many similar objective values are present,
- whether the objective values change continuously or by jumping from one level to the next.

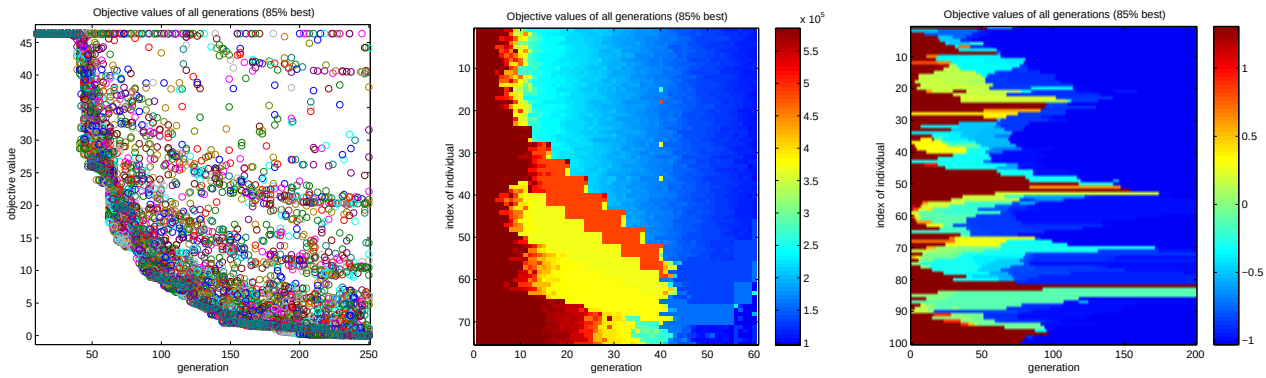


Figure 6: Objective values of all the individuals of many generations; left: 2-D point plot [RASTRIGIN's function, MPEA1], middle: 2-D image plot employing a regional model (multiple subpopulations with competition) [sphere function, MPEA1], right: 2-D image plot employing a one-dimensional local model [six-hump camel back function, MPEA1]

Two techniques are demonstrated for the visualization of this data type. The 2-D point plot, Figure 6 left, shows the distribution of the objective values. Attraction levels and the continuous changing of the objective values can be clearly seen. The 2-D image plot, Figure 6 middle and right, also shows the assignment between the objective value and the position of the individual in the population. In a panmictic population this is not really useful. However, when we employ the regional population model with or without competition between the subpopulations (MPEA1), Figure 6 middle, we can extract further information. We actually "see" the boundaries between the subpopulations here as well as the changing size of the subpopulations. Moreover, we also see the effect of migration. In generation 40 migration takes place, transferring a few good individuals between subpopulations. Try to understand this many numbers and effects in another way - you will be lost!

The 2-D image plot also comes in handy when analyzing the use of a local population model. A distance between individuals develops as selection is performed in a limited neighborhood only. Thus, information cannot spread very rapidly in the population. This effect of the "diffusion" of good objective values in the population during an optimization run is obvious in Figure 6 right. The visualization technique is exactly the same as the one used for the regional population model (or with a panmictic population). We obtain this extra information without having to carry out additional work.

Similar to Section 4 (especially Figure 5), a 3-D point or line plot is not very useful for the presentation of all objective values of the population during an optimization run. The relation between the objective value and the position of the individual in the population is not as easy to understand as in the 2-D image plot.

One problem remains - the scaling of the objective values. Normally, the range of possible objective values is not known. This range can be quite large. Because of the large changes at the beginning of an optimization run it is difficult to see the smaller changes at the end. Logarithmic scaling (similar to presenting the objective value of the best individuals in Section 3) is one possible solution to the problem. However, we chose a different one. For the scaling of the diagram we use only the better part of the population (85% of the individuals of the population proved useful). The objective values of all those individuals that are worse are set to this maximal value (just for visualization). In this way, we move the worst individuals out of the uninteresting area into the value range most interesting for the user. This kind of scaling was carried out in all the diagrams in Figure 6. For documentation purposes the percentage of the population used is shown in the title of the diagram.

This type of diagram is indispensable in our everyday work. Especially in many discussions with colleagues and application-specific engineers, it has proved very useful for explaining questions about the behavior of the optimization run. We have already been able to present many insights into the optimization by combining this data type with the previous two data types (objective value of the best individuals, see Section 3, and variables of the best individual from many generations, see Section 4).

## 6 How do the different subpopulation strategies compare?

For the solution of real-world applications it is often difficult to decide which of the available evolutionary algorithms are best suited and how the operators and parameters should be combined. The use of competing subpopulations which employ different strategies offers an elegant solution to these questions. The *application of different strategies* is the result of the definition of specific strategy parameters for every subpopulation (e.g. different mutation and recombination operators and/or parameters). In this way it is possible to specify different behavior for every subpopulation of the evolutionary algorithm.

The extension of *competing subpopulations* is based on the application of different strategies. However, it goes one step further than that. The different strategies are not only applied simultaneously but the successful ones also receive more resources than the less successful ones. This leads to a dynamic distribution of resources (changing number of individuals).

The success of a subpopulation is measured by its rank within the order of subpopulations. This approach is analogous to the ranking during the fitness assignment/selection process of the evolutionary algorithm. From the rank of a subpopulation we are able to assess the utility of a subpopulation in comparison to the other subpopulations. In this case, a low rank is better than a high rank. To calculate the order of subpopulations they have to be sorted according to one criterion. Since a subpopulation is made up of a number of individuals its quality results from the properties of its individuals. For a full description of both extensions refer to [8], chapter 4 or [10].

The main questions during and after optimization are:

- Which strategy / subpopulations are successful (or not successful)?
- When is a strategy / subpopulation successful?

Both questions can be answered by employing standard 2-D line plots of the available data on the size and/or order of the subpopulations. The example in Figure 7 (optimization of the hyper sphere function, also called DeJong function 1) presents the results for competing subpopulations.

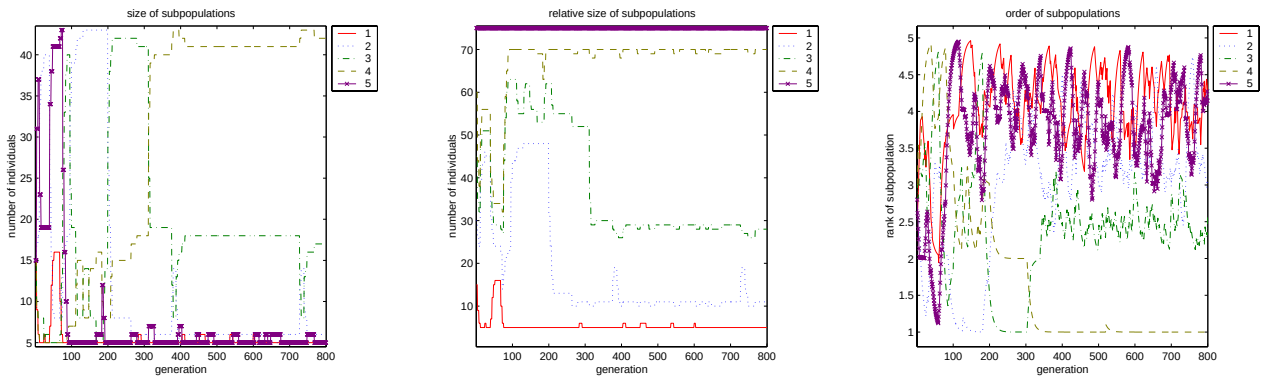


Figure 7: Order or size of subpopulations – 2-D line plot; left: direct visualization of the size of the subpopulations, middle: cumulative size of subpopulations, right: order/ranking of subpopulations (best subpopulation gets lowest rank) [hyper sphere function, MPEA1]

The left diagram in Figure 7 displays the absolute size of the single subpopulations while the middle diagram shows their relative size using a stacked line plot. At the beginning of the optimization, subpopulation 2 is the most successful, thus the size of subpopulation 2 increases. Around generation 50, the strategy of subpopulation 5 becomes more successful. Hence this subpopulation receives more resources; its size grows while subpopulation 2 decreases in size. This change between successful subpopulations occurs in generations 110, 210 and 340 for other strategies.

The right diagram in Figure 7 presents the order of the subpopulation. This can be seen as the raw data of the subpopulations' success rate. It offers an explicitly more detailed picture of the subpopulations' order. Here, short time changes in the order of the subpopulations can also be seen. At the same time it is definitely more difficult to derive a clear picture. However, the less filtered data may provide a more sophisticated answer, especially in cases which are not as clear-cut as the one presented.

Regarding the visualization, no special requirements need to be adhered to for these diagrams since all employ a standard 2-D line plot. The use of distinguishable line styles should be standard. The use of the stacked line plot in the middle diagram in

Figure 7 is a special peculiarity. The number of individuals in the middle diagram results from the distance between two contiguous lines. This adds the advantage that the index of each subpopulation can be identified without a legend. The first subpopulation is at the bottom of the diagram, the second is above and the last one is at the top. Even for 15 or more subpopulations (when a legend would get far too large) this diagram still provides the necessary information and we need only a few different line styles to distinguish between adjacent lines. Thus, this type of diagram can be scaled to a much higher number of subpopulations than the other two diagrams.

## 7 Objective values of all the individuals of one generation

All the previous visualizations presented the course of the optimization. Yet there is always a point at which these "overviews" deliver a picture which is too rough. For these areas more detailed diagrams are needed, which present the **state of the population** or the optimization in detail. The objective values of one population are the first stage of this deeper detailing.

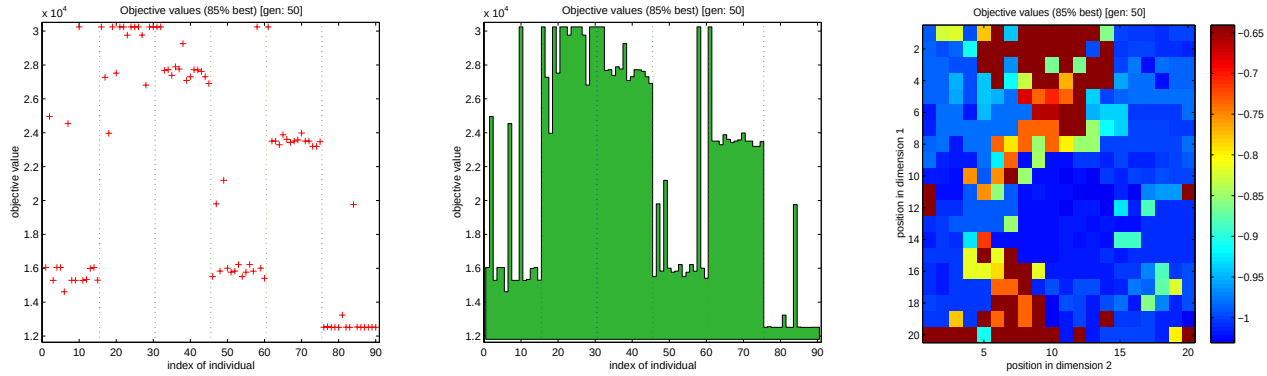


Figure 8: Objective values of all the individuals of a population in one generation; left: 2-D point plot, middle: 2-D filled stairs plot [hyper sphere function, MPEA1], right: 2-D image plot employing a two-dimensional local model [six-hump camel back function]

Figure 8 presents three techniques for visualizing the objective values. The 2-D point plot in Figure 8 left is straightforward. By using a filled 2-D stairs plot, Figure 8 middle, the differences within the population and between adjacent individuals become more apparent. Both diagrams display a population divided into subpopulations. The dotted lines mark the boundaries between the subpopulations. Thus, these diagrams also present information about the relative performance of the subpopulations in greater detail than the diagrams in Figure 7.

Figure 8 right employs a 2-D image plot to visualize the two-dimensional neighborhood of the objective values when using a two-dimensional local model. A sequence of such diagrams shows the diffusion of good objective values through the population during an optimization run.

## 8 Variables of all the individuals of one generation

The visualization of the variables of a population's individuals provides a very detailed insight into the distribution of the individuals, the distance between them and their concentration in one or more areas. However, because of the enormous amount of information involved, the technique chosen for visualization is important.

Figure 9 shows two techniques at three different stages of an optimization run. The upper row of diagrams shows a 2-D line plot of the variables of the best individual together with the mean and the standard derivation of the variables of all the individuals. The sequence of three diagrams displays the change in the variables' values and their distribution. The lower row shows the same data as a 2-D image plot. Due to the additional dimension of the image plot the data does not need to be compressed, but can be displayed directly. Comparing both diagram types we can see that the contents of the lower row can be retrieved more easily and in a more intuitive manner.

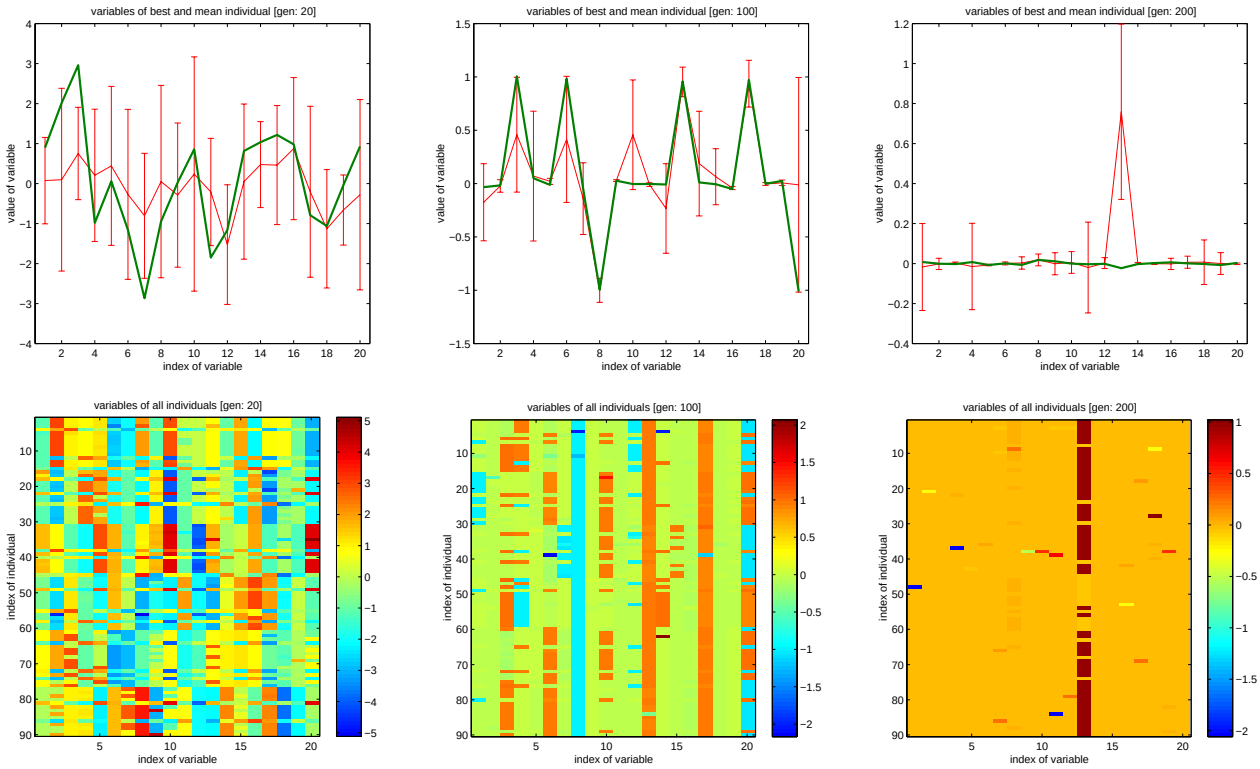


Figure 9: Variables of all the individuals of a population: 2-D line plot of the best (thick line) and the mean individual including standard derivation of all the variables at three stages of an optimization run; left: beginning of the run, middle: middle of the run, right: end of the run

All the data types and techniques visualizing the state of a population or the evolutionary algorithm should be used in a sequence of the respective diagrams during an optimization run or afterwards. The animation of these diagrams can reveal new and extended information and insights into the issue under investigation. The only problem is that these animations are hard to present in print. On the other hand, more and more papers are being published on-line, hence this restriction should diminish over the next years.

## 9 Concluding remarks

In this paper we have reviewed a number of visualization techniques for understanding the evolutionary optimization process. They provide a baseline for analyzing the behavior of the course and state of the evolutionary algorithm. Beginning with the convergence diagram, each of the data types presented provides more details of the optimization run. By selecting and combining diagrams of different data types the "necessary" level of detail can be defined.

The question remains of how data types and diagrams can be combined best. There is no simple answer to this question. The standard behavior in [7] is to use all 6 data types. This could be quite a lot of information, but the user gains an impression of the available data (and hopefully tries to understand it).

A good combination using a lower number of diagrams would be:

- the convergence diagram (Section 3),
- the visualization of the variables of the best individuals of all the generations (Section 4) and
- the objective values of all the individuals in the population of all the generations (Section 5).

The visualization techniques presented were applied in the course of solving real-world problems using evolutionary algorithms. Thus, these techniques are used in everyday work, have proved informative (see [8]), and could answer more than 95% of our questions regarding the behavior of the evolutionary algorithms employed. An implementation of all of these and additional techniques can be found in [7].

### References

[1] *Bird, J., Bullock, S., and Smith, T.*: Beyond Fitness: Visualising Evolution. Workshop at Artificial Life VIII: The 8th International Conference on the Simulation and Synthesis of Living Systems, University of New South Wales, Sydney, NSW, Australia, 2002. http://www.cogs.susx.ac.uk/users/toms/Visualization/Alife8Workshop.html

[2] *Collins, T. D.*: The Visualisation of Genetic Algorithms. Msc. Thesis, De Montfort University, Leicester, GB, 1993.

[3] *Collins, T. D.*: The Visualization of Genetic Algorithms – Related Work. Technical Report, KMI-TR-19, Knowledge Media Institute, The Open University, Milton Keynes, UK, 1995. http://kmi.open.ac.uk/kmi-abstracts/kmi-tr-19-abstract.html

[4] *Collins, T. D.*: Genotypic-Space Mapping: Population Visualization for Genetic Algorithms. Technical Report, KMI-TR-39, Knowledge Media Institute, The Open University, Milton Keynes, UK, 1997.

[5] *Collins, T. D.*: Visualization of Evolutionary Algorithms. Workshop at GECCO'99 - Genetic and Evolutionary Computation Conference, Orlando, Florida, USA, 1999.

[6] *Mathworks, The*: Matlab - UserGuide. Natick, Mass.: The Mathworks, Inc., 1994-1999. http://www.mathworks.com/

[7] *Pohlheim, H.*: GEATbx - Genetic and Evolutionary Algorithm Toolbox for Matlab. http://www.geatbx.com/, 1994-2003.

[8] *Pohlheim, H.*: Evolutionäre Algorithmen - Verfahren, Operatoren, Hinweise aus der Praxis. Berlin, Heidelberg: Springer-Verlag, 1999. http://www.pohlheim.com/eavoh/

[9] *Pohlheim, H.*: Visualization of Evolutionary Algorithms - Set of Standard Techniques and Multidimensional Visualization. In Banzhaf, W. (ed.): GECCO'99 - Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA: Morgan Kaufmann, pp. 533-540, 1999.

[10] *Pohlheim, H.*: Competition and Cooperation in Extended Evolutionary Algorithms. in Spector, L. (ed.): GECCO'2001 - Proceedings of the Genetic and Evolutionary Computation Conference – Late Breaking Papers. San Francisco, CA: Morgan Kaufmann Publishers, pp. 331-338, 2001. http://www.pohlheim.com/publications.html

[11] *Routen, T. W. and Collins, T. D.*: The Visualisation of AI Techniques. Proceedings of Third International Conference on Computational Graphics and Visualisation Techniques COMPUGRAPH'93, Alvor, Portugal, New York, USA: ACM Press, pp. 274-282, 1993

[12] *Routen, T. W.*: Techniques for the Visualisation of Genetic Algorithms. in Proceedings of The First IEEE Conference on Evolutionary Computation, Piscataway, New Jersey, USA: IEEE Service Center, Vol. II, pp. 846-851, 1994.

[13] *Schwehm, M.*: Globale Optimierung mit massiv parallelen genetischen Algorithmen. Dissertation, Universität Erlangen-Nürnberg, 1996.

[14] *Shine, W. and Eick, C.*: Visualizing the evolution of genetic algorithm search processes. In The Proceedings of the 1997 IEEE International Conference on Evolutionary Computation ICEC'97, pp. 367-372, 1997.