

this is a rather abstract position related to our topic of Bionic Optimization, we do not want to deepen this debate.

1.4 Terms and Definitions in Optimization

To discuss optimization effectively in the next chapters, we have to agree on a common language, i.e. the use of the same terms for the same phenomena. As optimization research is done by various groups within various and diverse scientific fields, and also in different regions of the earth, there is the real danger to get confused as the meanings of terms may diverge from group to group. Thus, we must clarify the set of terms used in this book. Most people involved in optimization accept that for an optimization study:

- We need a given *goal* or *objective* z .
- This objective z depends on a set of *free parameters* $p_1, p_2, \dots, p_n$.
- *Limits* and *constraints* are given for the parameters values.
- There are *restrictions* of the parameter combinations to avoid unacceptable solutions.
- We seek to find the *maximum* (or *minimum*) of $z(p_1, p_2, \dots, p_n)$.

To better define our terminology, we use the following conventions and findings:

- The *objective* or *goal* must be defined *à priori* and uniquely. Changing the definition of the goal is not allowed, as this poses a new question and requires a new optimization process.
- We need to define all free parameters and their acceptable *value ranges* we might modify during the optimization studies.
- This *value ranges* or *parameter range* is the span of the free parameters given by lower and upper limits. Generally it should be a continuous interval or a range of integer numbers.
- The fewer free parameters we must take into account, the faster the optimization advances. Consequently, accepting some parameters as fixed reduces the solution space and accelerates the process.
- Restrictions, such as unacceptable system responses or infeasible geometry, must be taken into account. But sometimes restrictions limit the ranges of parameters to be searched. Such barriers have the potential to prevent the optimization process from entering interesting regions.
- Finding the maximum of $z(p_1, p_2, \dots, p_n)$ is the same process as finding the minimum of the negative goal $-z(p_1, p_2, \dots, p_n)$. There is no need to distinguish between the search of maxima or minima.

Gradient based optimization methods are the most popular ways to find improvements of given situations. From an initial position, the derivatives of the objective

$z(p_1, p_2, \dots, p_n)$ with respect to the free parameters are determined. As the derivatives are often not to be found by analytical ways they are approximated by

$$\frac{\partial z}{\partial p_k} \approx \frac{z(p_1, p_2, \dots, p_k + \Delta p, \dots, p_n) - z(p_1, p_2, \dots, p_k - \Delta p, \dots, p_n)}{2\Delta p} \quad (1.1)$$

The column of these derivatives defines the gradient:

$$\nabla z = \left(\frac{\partial z}{\partial p_1}, \frac{\partial z}{\partial p_2}, \dots, \frac{\partial z}{\partial p_n} \right)^T \quad (1.2)$$

Jumping along this gradient, for example, by using a line search method such as Sequential Quadratic Programming (SQP) (Bonnans et al. 2006), or any related method, 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 (Fig. 1.4).

Optimization using this climbing of the ascent of the gradient is often labelled as the Gradient Method or included in the set of deterministic optimization methods. Here each step is determined by the selection of the starting point. Unfortunately, the determination of the gradient requires $2n + 1$ function evaluations per iteration, which may be an extended effort if the number of parameters is large and the hill not shaped nicely.

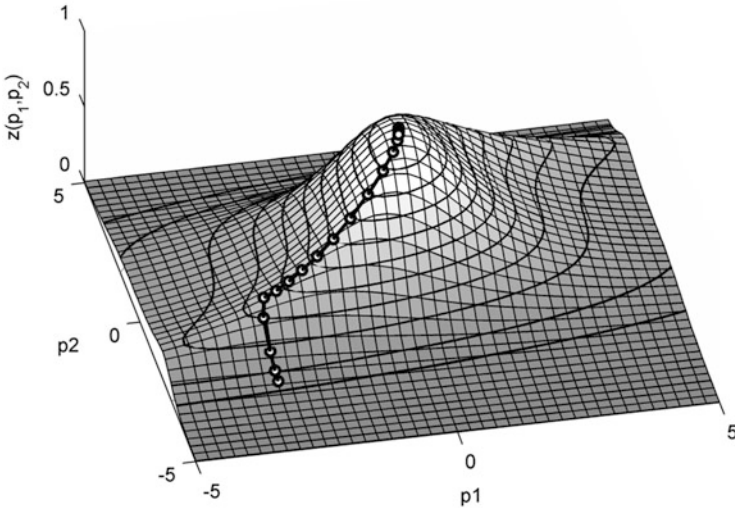


Fig. 1.4 Climbing up a hill using gradient methods

Pairing is the selection of two individuals of the parent generation to produce one common child.

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 (Fig. 2.1). 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, is also a possibility (Gen and Cheng 2000; Steinbuch 2010).

The quality of each individual is prescribed by its *fitness*. It can depend on one single value or be the weighted result of different optimization interests (cf. Sect. 6.2).

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 kids as new parents. Experience suggests that at least some lesser performing kids should be parents as well. For the sake of simplicity and comparability, here we allow the old parents to be parents in following generations and take the best of the total of old parents and kids to be parents in the next generation. Sometimes it proves efficient to define a restricted lifetime for such parents, given by a maximum number of generations.

2.1.2 Description of the Evolutionary Strategy

In the first step a number of individuals are randomly distributed within the design space. We determine their fitness, and they are the parents in the first generation.

For proliferation, the selected individuals are arranged in pairs. Every pair generates one kid only by combining the properties of their parameter sets. This crossed parameter set is subject to some random modification prescribed by the *mutation*. Then best kids and parents are chosen to be parents of the next generation. The poorer performing kids are removed from the gene pool. A pseudocode for a simple Evolutionary Optimization is shown in Table 2.1.

Variants

As there are different ways to cross, select and mutate individuals, we want to discuss selected modifications that are thought to be important in Evolutionary Optimization.

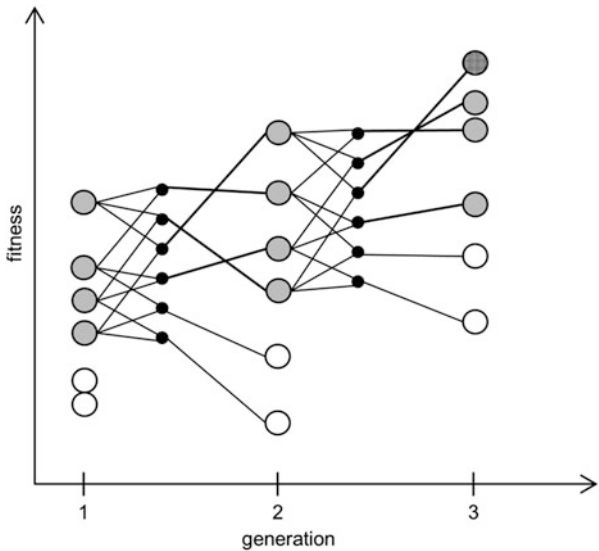
– Pairing Variants:

There are several ways of combining individuals to form a set of parents. Possibilities include randomly chosen pairs of individuals from the number of prospective parents, or the pairwise combination of individuals sorted by their fitness (combining the fittest pair, the second fittest pair, see Fig. 2.2).

Table 2.1 Pseudocode: Evolutionary Optimization

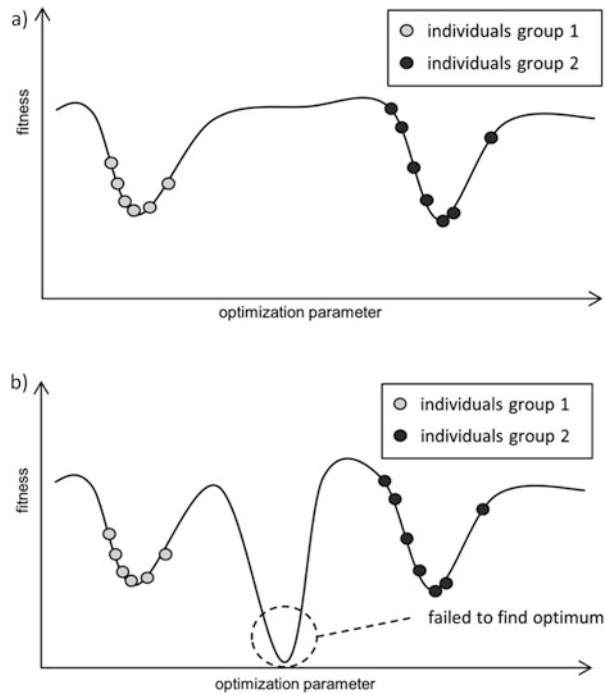
Initial: Define parameters: – Number of parents, kids, generations – Shall parents survive? – Crossing scheme, mutation radius – Define <i>nparents</i> parents; evaluate their performance and restriction violation
Start <i>ngenerations</i> loop: – Define <i>nkids</i> pairs of parents – Cross their properties – Mutate the properties to define kids – Evaluate the kids’ performance and restriction violation – Select the next generation’s parents from the kids (including parents?)
End loop
Stop

Fig. 2.2 Schematic diagram of Evolutionary Optimization: six initial individuals, four parents, six kids, over three generations



A possibility to affect the chosen sets of partners is combination based on the similarity of their properties, just as for many organisms in nature. The choice of partners depends strongly on similar interests, habitat, level of intelligence, attractiveness and so on. This kind of pairing can lead to a faster convergence because it concentrates the search only near optimal regions, and neglects the spaces in-between (Fig. 2.3a). However, any optima in the neglected spaces will not be found (Fig. 2.3b). Nevertheless, this method offers an interesting possibility to force the optimization to run in a specific direction. Users can include basic knowledge of the problem in the start of the optimization to affect the kinds of optima they find.

Fig. 2.3 Examples of pairing individuals with limited distance. (a) Pairing with limited distance lead to fast convergence. (b) Pairing with limited distance failed to find optimum



- Killing the parents:
Should the parents survive to be parents in the next generation as well? Dominant parents reduce the chances of weaker children to find their way, but excluding them removes previously found best solutions.
- Add new random parents:
Adding random parents in certain optimization steps increases the chance to cover the whole design space. Although this method contradicts the idea of the optimization algorithms, it prevents the population from converging to local maxima.

2.1.3 Evolutionary vs. Genetic Strategy

Rolf Steinbuch

From the beginning of Evolutionary Optimization until now, there is a schism between two schools. The first of them, often called evolutionary direction, uses floating point representation of the parameters' values. Crossing is done by weighted averaging (cf. Fig. 2.1), where mutation means randomly changing values. The other direction, the Genetic Algorithms school, as it often labels itself, has a different view on the information stored in the genes of the parameter values representation. We all know that DNA which carries genetic information uses a

Bionic Optimization in Structural Design
Stochastically Based Methods to Improve the
Performance of Parts and Assemblies

Steinbuch, R.; Gekeler, S. (Eds.)

2016, XII, 160 p. 103 illus., 6 illus. in color., Hardcover

ISBN: 978-3-662-46595-0