Smooth genetic algorithm

1994 J. Phys. A: Math. Gen. 27 7893

(http://iopscience.iop.org/0305-4470/27/23/032)

J. Phys. A: Math. Gen. 27 (1994) 7893–7904. Printed in the UK

Smooth genetic algorithm

Marek W Gutowski[†]

Institute of Physics, ON-3.2, Polish Academy of Sciences, Al. Lotników 32/46, 02-668 Warsaw, Poland

Received 1 March 1994, in final form 26 September 1994

Abstract. An existing family of genetic algorithms, which were designed with discrete and binary variables in mind, has been extended in this paper to handle truly continuous variables. Its close relationships with Monte Carlo methods, the simplex method, simulated annealing and other direct, i.e. derivative-free global optimization algorithms creates a really versatile tool for various difficult optimization tasks. The main area of its application should be the reconstruction of unknown, continuous, and possibly smooth, distributions of various physical quantities derived from the experimental data. Among them might be: grain-size distribution for particulate magnetic materials derived from isothermal magnetization curves, distribution of relaxation times derived from luminescence experiments or chemical kinetics (inverse Laplace transform), and other large-scale numerically hard problems. One such problem, namely solving for the grain-size distribution for particulate magnetic materials, is presented as a working example and treated in detail. Applications of this algorithm should be stable deconvolution of various spectra with a variable window and non-parametric curve smoothing with a non-smooth objective function.

1. Introduction

The genetic algorithm, also known as evolutionary strategy, was probably first introduced in 1975 by Holland [1]. Since then it has gained much attention among researchers conducting large-scale optimization calculations. Its original form, designed only for binary variables, has been extended to handle variables which may take more than two discrete values. Recently, several papers, for example [2, 3, 6], were published utilizing the genetic algorithm working with a discretized form of continuous variables. More applications of this approach can be found in [3–12]. Other reported areas include: image processing, job scheduling, pattern recognition, design of integrated circuits, modelling and system identification, etc. A good tutorial on various aspects of genetic computing was recently given by Lucasius and Kateman [13].

Here we only briefly summarize the algorithm in question. We will not discuss many important, but rather technical, details of the implementation, since those are usually problem- and computer-language-dependent. Their discussion can be found in [14] and references therein. Our main goal is to introduce and familiarize the reader with the terminology used thoroughout the rest of paper.

The problem. Given a (real) function $f(x_1, \ldots, x_N)$ of N variables find the set

$$\{x_1, \ldots, x_N : (x_j = 0 \vee x_j = 1) \forall 1 \leqslant j \leqslant N\} \qquad (1)$$

[†] E-mail address: gutow@gamma1.ifpan.edu.pl

which minimizes a given objective function f.

We will seek candidates for the final solution in the form of strings similar to

$$\texttt{0001010101111100011110...11001110111} \qquad (2)$$

where each position assumes the value of the corresponding unknown (0 or 1). We will refer to such a string as a chromosome or animal, and to its single element as a gene. The objective function, f, is usually called the fitness function or just fitness. The first step of the genetic algorithm is the creation of several (usually 50–500) trial solutions of this form comprising the initial population ('farm'). Many authors recommend completely random strings as members of the initial population. Now we allow this population to evolve, generation after generation. Evolution is governed by the Darwinian rule 'survival of the fittest'.

The following two steps are repeated to create a new generation from the previous one:

- selection of two parents (the better fitted a given chromosome is, the higher chance it has to be selected as a parent);
- the decision of whether selected parents-to-be cross or not. If not, then they are simply copied to the new generation without any changes, otherwise they produce two offspring chromosomes which, after evaluation of their fitness, enter the new generation.

The decision on whether or not the given chromosome will be allowed to replicate, may be made after investigation of the inequality proposed in this paper, to the author's knowledge, for the first time:

$$r < \frac{1}{1 + \exp(f - M_f/S_f)} \qquad (3)$$

where r is a random number drawn from uniform distribution on $[0, 1]$, f is the value of fitness for a given chromosome, M_f is the median of fitness calculated for the entire population, and S_f is the standard deviation of the population's fitness from the median. The chromosome may become a parent, if the above inequality holds.

Let us make two remarks concerning the right-hand side of inequality (3). First, it is the familiar Fermi–Dirac distribution with the following correspondence relations:

f —fitness (or better: misfit, since usually we want to minimize f) of a given chromosome
—energy of individual particle from the ensemble
M_f—median of the population's fitness
—Fermi level (chemical potential)
S_f —standard deviation of the population's fitness from the median
—absolute temperature.

Every chromosome with fitness equal to the Fermi level has a 50% chance of being selected as a parent. We prefer to use the median value of fitness *in lieu* of the usual arithmetic average. There is a good reason for doing so: some offspring and mutants reach exceptionally high values of f, which very seriously affects the average value for the population, while the median, although more difficult to obtain, is insensitive to such outliers.

The concept of temperature, already considered by some authors [3, 6], in the context of Gibbs partition function, is also very useful. The introduction of this parameter makes the algorithm have some of the properties of the simulated annealing Monte Carlo procedure, which has proved to be very efficient in many global optimization problems, but without the need for a special cooling schedule. In our model, the temperature is a self-adjusting parameter.

The second remark concerns the shape of the Fermi–Dirac distribution. It is exactly the sigmoid response function modelling the behaviour of a single neuron, like those used in artificial neural networks. Perhaps it is the shape of the Fermi–Dirac distribution which makes the two very different entities, neural networks and genetic algorithms, so intelligent.

The need for stretching or contracting the fitness functions before they are used in the selection process is apparent. Some authors [3] argue, that it is necessary to exaggerate the differences between the fitness functions to help the process discriminate between trial solutions with slightly different fitness values. Here we are talking about well fitted chromosomes, with almost equal fitnesses. According to commonly held belief, this kind of transformation of fitnesses to probabilities should improve the performance of genetic algorithms. On the other hand, the purpose of mutations is to help create quite new trial solutions by making big jumps in the search space. For this purpose, it would be very desirable not to differentiate various poorly fitted mutants too much. Our proposal, given in inequality 3, although heuristic, seems to fulfill both requirements. The concept of a Fermi level, crucial here, comes from nature, namely from analogy with the BCS theory of superconductivity and transport properties of semiconductors.

After the new generation (with the same number of members as the old one) has been created, the time comes to introduce mutants. Mutation is a relatively rare event, in which a randomly chosen gene in a randomly chosen chromosome is inverted, i.e. becomes zero, if it was one, and vice versa. The fitness of a mutated chromosome has to be evaluated again, of course. The lack of mutations might lead to the population consisting of identical animals, and thus be unable to create new trial solutions.

The evolutionary process is continued until the average fitness reaches the desired value, or the prescribed number of generations has been explored or some other convergence criterion is fulfilled. The best chromosome so far is usually considered to be the final result of the calculations; in some cases, however, the (weighted) average of the entire population may be preferred.

2. Description of the continuous algorithm

There are two basic differences between discrete and continuous versions of the genetic algorithm. One is the way the chromosomes are combined to produce offspring and the other one concerns the mechanism of the mutations. In the simplest, binary, version we have (the vertical line marks the crossover point)

```
                |
 0010100111100|111011   –  first parent
 0111011011011|011001   –  second parent
                |
 0010100111100|011001   –  first offspring                 (4)
 0111011011011|111011   –  second offspring
                |
 0111011011011|111010   –  mutated second offspring
                |      ↑   –  mutated gene
```

i.e. the heads of the new chromosomes are identical with those of their respective parents, while the tails are interchanged. The crossover point is selected at random, individually for each pair of parents. The mutant differs in at least one gene from its originator. The position of this gene is also chosen at random.

The process of reproduction shown above may be described in terms of 'mixing' (characteristic, weighting) function, w. In the discrete case, the domain of the mixing function is a subset of natural numbers directly related to the ordering of unknowns. In the classical, binary version of the genetic algorithm there are only two allowed values for the mixing function, namely 0 and 1, and the function itself is monotonous, i.e. after a series of nulls a sequence of ones follows (or reversely). The reproduction process may then be formally written as

$$\begin{aligned} o_1(j) &= w(j)p_1(j) + [1 - w(j)]p_2(j) \\ o_2(j) &= [1 - w(j)]p_1(j) + w(j)p_2(j) \end{aligned} \tag{5}$$

where $p_1(\cdot)$ and $p_2(\cdot)$ are parents, $o_1(\cdot)$ and $o_2(\cdot)$ are offsprings and the index j numbers the unknowns (genes).

For the continuous case we can afford much more sophisticated mixing functions. The function w should be continuous and its values should cover the range $[0, 1]$, but monotonicity is not required. Moreover, if the mixing function is smooth, and the initial population consists of smooth animals only, then all the intermediate and final populations will contain only smooth animals too. This feature is extremely desirable when dealing with some large, ill-conditioned optimization problems. By such we mean here, most of all, but not only, finding unknown distributions, in discretized form, derived from experimental data *ab initio*, i.e. with only very general assumptions—such as the smoothness of solutions. Assuming complete independence of all unknowns in such a problem normally leads to 'noisy' or discontinuous results, or unstable behaviour of the minimization procedure, unless special precautions are taken. These are usually based on some kind of 'regularization', or extra constraints, introduced more or less *ad hoc*, and sometimes quite arbitrarily. Our approach allows us to think about solutions to this kind of problem as tabulated functions, treated as single objects, rather than sets of many unknown numbers. Our point will be explained more fully in the section 'an example'.

As an example of continuous numbering of unknowns consider the problem of finding the distribution of relaxation times in a luminescence experiment. Looking for solutions in the form of a histogram, we have to find the shares of many relaxation times in the entire process, i.e. our unknowns are the contents of the histogram's bins. The bins, in turn, are ordered in a natural way: every one is centred on a specific relaxation time. Of course, there is no necessity to have the bins of equal width (evenly spaced). They may be spaced logarithmically, or even quite irregularly, on the real axis. So the ordering variable, here called the 'relaxation time', τ, may be used to number the unknowns and constitute the domain of mixing functions for further calculations. Alternatively, one may look for solutions of this particular problem in terms of properly spaced decay constants. This problem (inverse Laplace transform) has been known, for more than one hundred years, as a very difficult and ill-conditioned one, even when constrained to non-negative amplitudes.

Suppose, we are looking for the histogram covering relaxation times in range $[\tau_{\min}, \tau_{\max}]$, where the bounds $\tau_{\min}$ and $\tau_{\max}$ should be estimated independently from available data.

A suitable mixing function might have the shape

$$w(\tau) = \tfrac{1}{2}\left\{1 + \tanh\left[A \sin\left(\xi + B\pi \frac{\tau - \tau_{\min}}{\tau_{\max} - \tau_{\min}}\right)\right]\right\} \tag{6}$$

where ξ is a random, but fixed, number evenly distributed over $[0, 2\pi]$, A and B are positive constants and tanh denotes the hyperbolic tangent function.

An example of continuous parents and offspring, together with the mixing function, with $A = 7.254$ and $B = 1$ is given in figure 1.

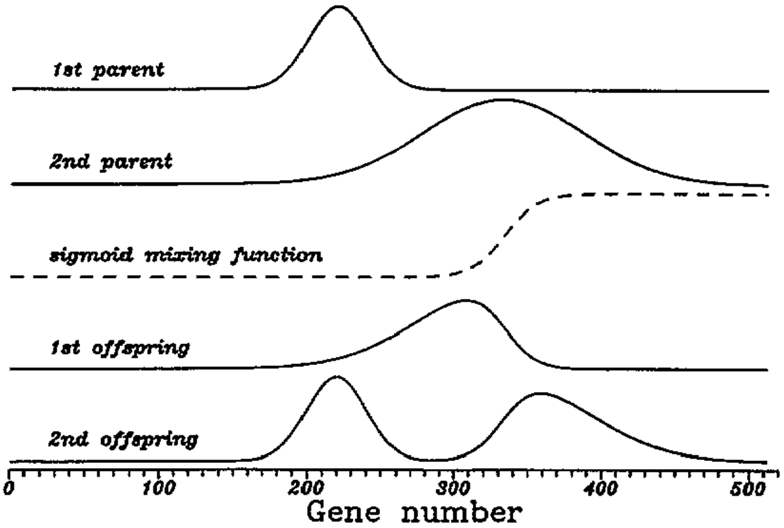

Figure 1. Two parent chromosomes and their offspring produced by the described algorithm. The animals consist of 512 genes each.

Other, simpler mixing functions may be preferable, but this one exhibits interesting properties: it becomes more and more similar to a square wave when the value of the parameter A is increased. This shows that a smooth transition is possible from the continuous to the discrete version of the algorithm. On the other hand, if constant $B \gg 1$, then w becomes highly oscillatory, and the algorithm resembles more and more the classical, memoryless Monte Carlo method. On the other hand, decreasing this factor below unity (so the weighting function is almost constant) makes the algorithm look similar to the well known simplex method. The classical, discrete case, presented earlier, corresponds to $B = 1$ and a very high value of A. Using a periodic function, like sin, has this effect, that all genes are treated equally, regardless of their location within the string of unknowns. The case of the integer parameter B ($B = 2$ is most often used) is well known in the literature as a 'multiple crossover point'. In the continuous case, however, there is no need to restrict the value of parameter B to integers only.

Note, that the proposed method of breeding new animals from the previous generation has a very desirable feature of generating smooth offspring from smooth parents.

The most difficult part in transforming the discrete algorithm into its continuous counterpart is the mutation mechanism. In the discrete version there is always a chance of reintroducing the gene, which accidentally got lost during the evolution, or was absent in the initial population. This feature is much more difficult to implement in the continuous version but—perhaps—it is not necessary. The continuous version of the mutation process is more difficult but also offers more flexibility. The difficulty lies in the simple fact, that any modification of a single gene will immediately destroy the continuity of the trial solution. The effect of mutation must somehow be distributed over neighbouring genes. Therefore one should consider a kind of 'plastic deformation' of an entire animal as a way of introducing the mutation process. An applied modification may be concentrated in the selected part of an animal or be visible in its whole body. This proposition is in full accordance with our experience from the biological world: some mutants have only the colour of their eyes changed, while others may lack the pigments in their entire skin.

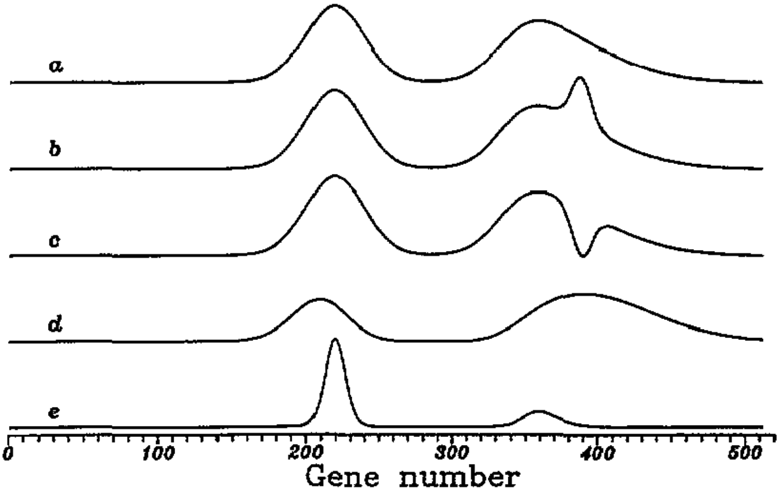

Figure 2. Various smooth mutations of the animal, marked as offspring no 2 in figure 1; presentation of local versus global effect of mutation (not to scale): (*a*) original animal before mutation, (*b*) multiplying every gene by a shifted Gaussian function with positive amplitude—an extra peak is generated, (*c*) multiplying every gene by a shifted Gaussian function with negative amplitude, localized as in case (*b*), splitting of existing peak occurs, (*d*) multiplying by a quadratic function—all peak parameters change, most notably their positions, (*e*) every gene nonlinearly transformed by an exp function—peak positions are preserved but widths and amplitudes are changed.

Generally we can describe the mutation of animal m as

$$m'(j) = T(m(j)) \tag{7}$$

where $m(j)$ and $m'(j)$ denote the gene j before and after mutation, respectively. Transformation T may be defined in a variety of ways provided it is smooth. As an example, consider the animals, which represent trial solutions of the spectral deconvolution problem, say in the visible part of the electromagnetic spectrum. Valuable 'pure' transformations could be

- decreasing (or increasing) the width of spectral peaks,
- shifting peak positions,
- changing the peak amplitudes,
- baseline shifts.

While the first, second and third transformation may be regarded as more or less localized, the last one has a global character. In practice, one will select transformations of a simple analytical shape but of mixed character, i.e. changing several features at once. A good representative of such a transformation is multiplication of the trial spectrum by a quadratic function. Examples of various kinds of mutations are presented in figure 2.

3. Applications

The algorithm is well suited for problems with many unknowns, which are ordered in some natural way, and therefore are *not* 'independent', but forming a continuous—and perhaps smooth—curve. All kinds of spectroscopical problems, as well as others not requiring highly precise solutions but rather the shape of the curve may be attacked with this method.

The main area of applications should be the reconstruction of unknown continuous, and/or smooth, distributions of various physical quantities derived from the experimental data. Among them are: grain-size distribution for particulate magnetic materials derived from isothermal magnetization curves (treated in detail in the next section, with early results presented in [15, 16]), distribution of relaxation times derived from luminescence experiments or chemical kinetics (inverse Laplace transform), distribution of hyperfine fields deduced from Mössbauer spectra to name a few. It should also be possible to find (complex) kernels of some integral, physically important, transformations like the one used in Kramers–Krönig analysis of optical spectra.

Other large scale numerically hard problems can be treated too. As an example we may take stable deconvolution of various wideband electromagnetic (NMR, ESR, IR, optical, UV, gamma ray), acoustic, chromatographic and other spectra with a window function varying along the spectrum, i.e. with a non-constant resolution. This kind of spectra cannot be processed correctly with the available deconvolution procedures utilizing the Fourier transform technique or equivalent iterative approaches based on the convolution theorem.

Perhaps, not least, application of this algorithm might be the non-parametric curve smoothing in an ordinary least-squares sense or with a non-smooth, robust objective function like, for example, the least (median of) absolute deviations.

4. An example

The described algorithm has been applied to the following problem: given the experimental data on isothermal magnetization of a nanocrystalline sample, find the distribution of magnetic nanocrystallites. The experimental data, selected from many sets, and consisting of 191 points, are presented in figure 3. The initially amorphous $Fe_{66}Cr_8Cu_1Nb_3Si_{13}B_9$ metallic glass, when annealead in a controlled way, develops an ultrafine crystalline structure and exhibits superparamagnetic [17] behaviour at elevated temperatures, in this case above 523 K [18]. The temperature and magnetic-field dependence of magnetization for a superparamagnetic sample can be written as a sum of the Langevin functions

$$M(H,T) = \sum_j n_j \mu_j [\coth(\mu_j H/(k_B T)) - k_B T/(\mu_j H)] \tag{8}$$

where n_j is a number of single domain particles with magnetic moment equal to μ_j per unit volume, k_B is the Boltzmann constant and H is the magnetic field strength.

The distribution $n_j(\mu_j)$ is the object of interest in this example. We started searching this distribution with determination of the maximum value of magnetic moment per single crystallite, $\mu = \mu_{max}$. It was calculated from $\partial M(H,T)/\partial H|_{H=0}$ and later increased 20%. The interval [0, μ_{max}] was then subsequently divided into 99 channels of equal width. The initial population was formed by the set of histograms, of bell-like shapes, given by the formula

$$(A/j)\exp\left[-(j-j_k)^2/(2\sigma^2)\right] \tag{9}$$

where j is the channel number $1\ldots99$, corresponding to μ_j, j_k is the centre of kth animal $(= k\frac{99}{64})$ and σ, closely related to the width of distribution, is a real number from the interval [1, 10]. The centres of the animals were evenly distributed in the space of magnetic moments in order to have a chance for *every* possible value to be used, and perhaps eliminated, by the algorithm. As will be seen later in this section, it is impossible to create new genes, either by the cross-over process or by the mutation mechanism, in positions where they were absent in the initial population.

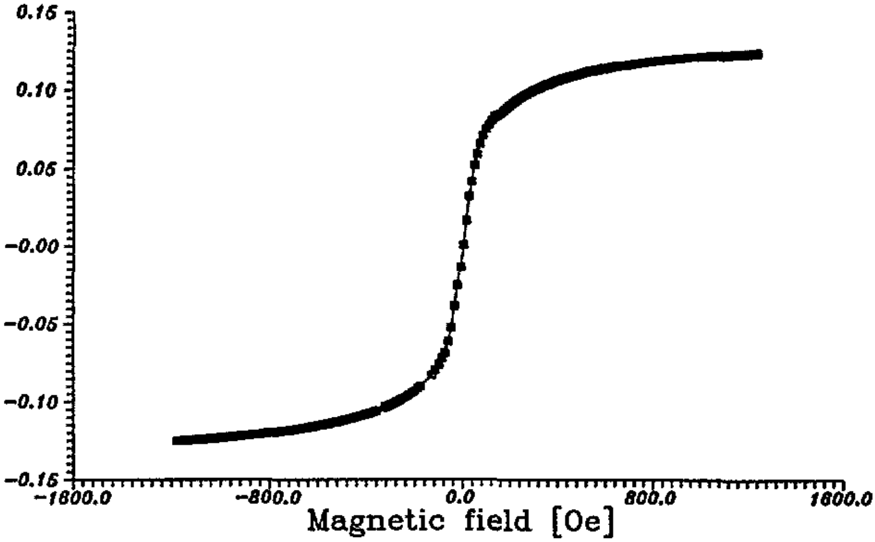

Figure 3. Magnetization curve (emu versus magnetic field strength) of a nanocrystalline sample ($Fe_{66}Cr_8Cu_1Nb_3Si_{13}B_9$) taken at 723 K using vibrating sample magnetometer. There is no visible hysteresis, since the sample is in the superparamagnetic state.

The best results (best fitted animals) were obtained with $\sigma = 3$, but for any value of σ the solutions had the same general shape, differing only with resolving power. There were only 64 members of the population. The normalization constant A was choosen in such a way to have (in given temperature)

$$\int_{H_{\min}}^{H_{\max}} |M_{\mathrm{calc}}(H, T)|\,\mathrm{d}H = \int_{H_{\min}}^{H_{\max}} |M_{\mathrm{exp}}(H, T)|\,\mathrm{d}H \tag{10}$$

where $H_{\min} < 0$ and $H_{\max} > 0$ are the minimum and maximum magnetic fields used during the experiment, respectively. The integral on the right-hand side was calculated only once for every data set to save processing time. The normalization was repeated for every new animal entering the population. Such a procedure considerably improves the ability of the algorithm to quickly identify and explore only the relevant parts of the search space—a property which is inherent to all genetic algorithms—thus significantly decreasing the computation time.

The objective function, F (after fitness), was constructed as

$$F = \sum_k |M_{\mathrm{calc}}(H_k, T) - M_{\mathrm{exp}}(H_k, T)|\,. \tag{11}$$

We prefer to use the sum of absolute differences between experimental and calculated (*simulated*) values, because it is more robust, i.e. more immune to random experimental errors, than the commonly utilized χ^2 criterion. Lack of smoothness in our objective function is meaningless, since the algorithm compares only the values of the function itself, not its derivatives. The results obtained for good quality (i.e. noiseless) data using our objective function and the χ^2 criterion are practically identical, while for noisy data our function performs better.

The result of calculations for the data set from figure 3 is shown in figure 4. Instead of presenting the results in the form of a histogram we have drawn a full curve connecting

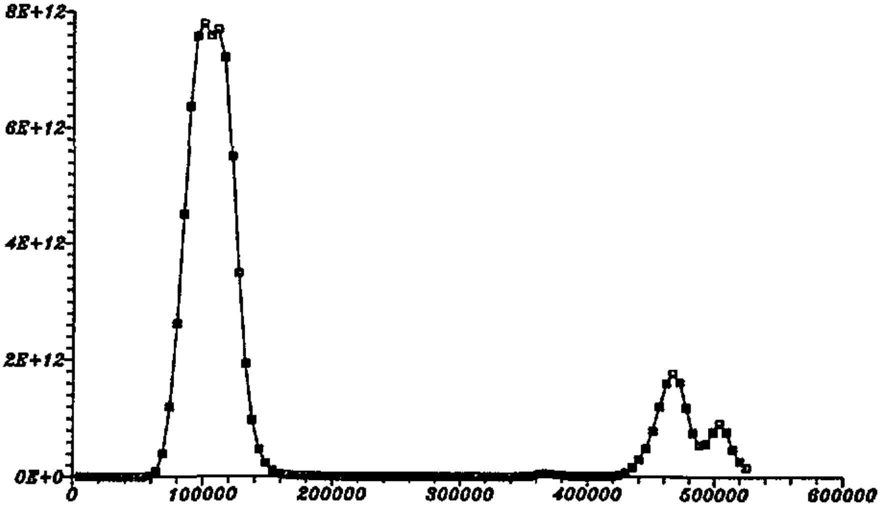

Figure 4. Computed distribution of nanocrystallites versus their 'magnetic' size expressed in Bohr magnetons.

the consecutive points, as a guide to the eye, in order to underline the smoothness of the distribution. Essentially, two peaks are visible. Their imperfections probably have no physical meaning—these should be attributed to the premature end of the calculations, or to unfortunate mutations, or both. The number of crystallites with various magnetic moments is reasonable (the sample mass was a few mg) and corresponds well to the so-called packing fraction, $p \approx 18\%$, determined independently from x-ray analysis. The packing fraction tells us how large the share of magnetic nanocrystalline material volume is within the sample. Two important conclusions can be drawn from these results.

- The distribution of crystallite sizes ('magnetic' or physical) is *not* unimodal, contrary to the common assumption, that it is log-normal. It is easy to check that, at least in our case, the shares of both groups of crystallites in the total magnetization of the sample are approximately equal to each other. It may appear that assuming only two kinds of magnetic objects are present in the sample will greatly simplify further calculations involving this distribution without sacrifice of agreement between theories and experiments.
- The ratio of volumes between two groups of crystallites is roughly 5:1, i.e. the ratio of their diameters is ~1.7:1 (we neglect here the possible differences in magnetic moments of Fe ions belonging to the surface of crystallites and those located inside the grains). This fact easily explains the high values of the packing fraction, in exceess of 70%, claimed by some authors (see, for example [19]). Having at hand two sizes of more or less spherical objects, we can fill the space even more densely than is possible in the regular HCP structure.

The probability of the cross-over between two selected parents was fixed at $\frac{2}{3}$. This means that, on average, a third part of the old population was simply transferred, without any changes, to the new generation. The probability of mutation was set to such a value as to expect one mutated animal in each generation. We use two kinds of mutations, with equal probability: local and global; both based on a scheme presented in (7). Local mutation *effectively* changes only a part of the chromosome in question, while a global one affects the entire animal. Local mutation is performed by multiplication of the animal by the function

$$1 \pm \tfrac{1}{2}\exp\left[-(j-j_0)^2/(2\sigma^2)\right]. \tag{12}$$

where j numbers genes, j_0 is the mutation centre, and σ is related to the mutation range. Both j_0 and σ are random numbers, with $\sigma \geqslant 5$. The upper sign corresponds to enhancing mutated genes, while the minus sign corresponds to their partial annihilation.

Global mutation is based on the factor

$$\exp\left[\pm\left(\frac{|m(j)|}{M}-\frac{1}{2}\right)\right] \tag{13}$$

where $M = \max_j |m(j)|$, applied to the entire animal, gene by gene. Similarly to the previous case, one of the signs (upper) corresponds to exponential stretching, while the other one (lower) is responsible for attenuation of the most pronounced genes.

We also decided to use the so-called *elitist strategy*—in its simplified form. An elitist strategy means nothing else but the warranty of survival for the best animal in case it is missing in the next generation *and* no better trial solution has been found. Our simplification of this strategy is based on comparisons between fitnesses of animals only, rather than on time-consuming immediate checking of their detailed shapes. The last approach is often used by those who deal with discrete problems.

The calculations are stopped when either of two conditions occurs:

- either the inequality:

$$|\text{median fitness} - \text{best fitness}| \leqslant \varepsilon|\text{best fitness}| \tag{14}$$

  is satisfied with $\varepsilon = 10^{-4}$ (preferred stopping criterion), or
- no improvement could be found after prescribed number of generations (64 in our case).

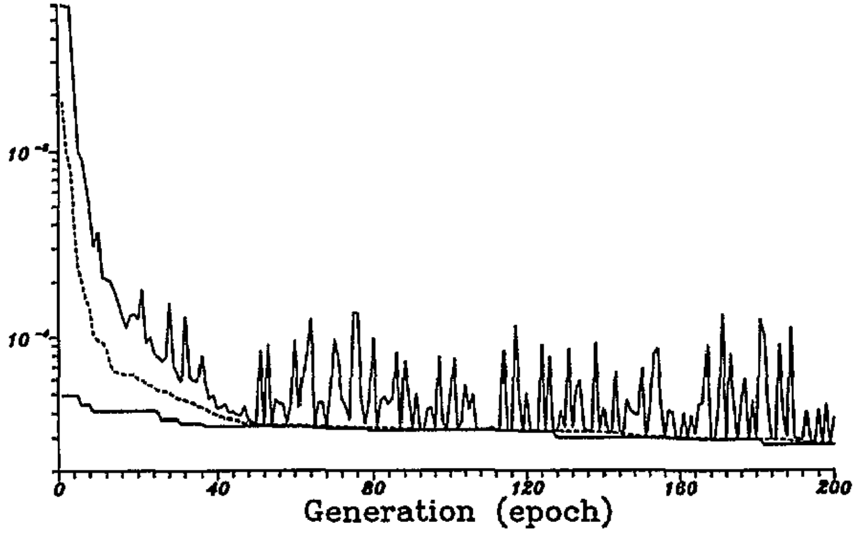

Figure 5. Typical course of calculations: full curves—best and poorest animals, broken curve—median value in the population. It is clear that, after some 50 generations, the population remains concentrated in a single domain (niche) of search space. The 'noise' is a manifestation of mutations. Note the logarithmic scale.

The first condition is satisfied when the population becomes homogeneous, in other words, when all the animals are almost identical, and the chances for significant improvement diminish. The second condition assures termination in all other cases.

Figure 5 shows typical history of a single run of algorithm. Usually 30–60 generations are sufficient to get quite satisfactory results. This may be expressed in terms of some 1300–2500 evaluations of the objective function. It is a far smaller number than that used by the blind Monte-Carlo-type algorithms and even by simulated-annealing-type algorithms. The number of objective function evaluations depends very weakly on the number of unknowns searched, contrary to many deterministic algorithms, which usually require $\sim N^2$ function evaluations. 1300–2500 evaluations is typical for these algorithms when dealing with problems involving only 5–15 independent unknowns. The memory requirements are of the same order of magnitude as for multiple start-point Monte-Carlo-type algorithms, deterministic simplex method or Newton/Marquardt minimization tools.

5. Conclusions

The presented algorithm is intuitively simple yet efficient, powerful and versatile. Its basis is formed using several well founded concepts taken from biology and statistical physics, as well as from other derivative-free optimization algorithms. Applications of the continuous evolutionary algorithm seem even broader and more important for physicists, than those of its discrete version. Although, at least in its present form, the algorithm cannot supply analytical solutions, nevertheless numerical solutions of high quality—like the one shown in the previous section—will be certainly appreciated by those who have to deal with noisy experimental data. Its ability to generate continuous and smooth solutions, for numerically hard problems, cannot be overemphasized. The algorithm thus provides a very attractive alternative to assess many variational problems. Also, some untractable problems may now be solved in an elegant and mathematically correct way.

Acknowledgments

This work was supported in part by the State Committee for Scientific Research under grant no 204 899 101. The author is indebted to Dr Anna Ślawska-Waniewska for the sample preparation and providing the results of her measurements used in this work. Sincere thanks go to two anonymous referees for their patience, words of encouragement and time they spent reading the first manuscript. Their comments and suggestions significantly influenced the final appearance of this paper.

References

[1] Holland J H 1975 *Adaptation in Natural and Artificial Systems* (Ann Arbor, MI: University of Michigan Press)

[2] Gallagher K, Sambridge M and Drijkoningen G 1991 Genetic Algorithms: an evolution from Monte Carlo methods for strongly non-linear geophysical optimization problems *Geophys. Res. Lett.* **18** 2177

[3] Stoffa P L and Sen M K 1991 Nonlinear multiparameter optimization using genetic algorithms: inversion of plane-wave seismograms *Geophys.* **56** 1794

[4] Lucasius C B and Kateman G 1991 Genetic algorithms for large-scale optimization in chemometrics: an application *Trends Analyt. Chem.* **10** 254

[5] Wilson W G and Vasudevan K 1991 Application of the genetic algorithm to residual statics estimation *Geophys. Res. Lett.* **18** 2181

[6] Sambridge M and Drijkoningen M 1992 Genetic algorithms in seismic waveform inversion *Geophys. J. Int.* **109** 323–42

[7] Hajima R, Takeda N, Ohashi H and Akiyama M 1992 Optimization of wiggler magnets ordering using a genetic algorithm *Nucl. Instrum. Meth. Phys. Res.* A **318** 822

[8] Li Tong-Hua, Lucasius C B and Kateman B 1992 Optimization of calibration data with the dynamic genetic algorithm *Analyt. Chim. Acta* **268** 123

[9] Global H B 1993 Geometry optimization of clusters using genetic algorithms *J. Phys. Chem.* **97** 9973

[10] Hibbert D B 1993 Generation and display of chemical structures by genetic algorithms *Chemom. Int. Lab. Sys.* **20** 35

[11] Diver D A 1993 Applications of genetic algorithms to the solution of ordinary differential equations *J. Phys. A: Math. Gen.* **26** 3503

[12] Cartwright H M and Harris S P 1993 Analysis of the distribution of airborne pollution using genetic algorithms *Atmosph. Environ.* A **27** 1783

[13] Lucasius C B and Kateman G 1993 Understanding and using genetic algorithms. Part 1: concepts, properties and context *Chemom. Int. Lab. Sys.* **19** 1

[14] See, for example,

Lucasius C B, de Weijer A P, Buydens L M C and Kateman G 1993 CFIT: a genetic algorithm for survival of the fitting *Chemom. Int. Lab. Sys.* **19** 337

Hibbert D B 1993 Genetic algorithms in chemistry *Chemom. Int. Lab. Sys.* **19** 277 (Turbo Pascal 6.0 routines to set up a simple genetic algorithm)

GENESIS—a complete set of genetic routines written in C, available from Dr John Grefenstette by E-mail: gref@aic.nrl.navy.mil, see also

Grefenstette J 1986 Optimization of control parameters for genetic algorithms *IEEE Trans. Sys. Man Cybern.* **16** 122

[15] Ślawska-Waniewska A, Gutowski M, Kuźmiński M and Lachowicz H K 1993 Particle sizes and magnetic properties of nanocrystalline fe-based metallic glasses *Conf. Proc. 13th General Conf. of the Condensed Matter Division, European Physical Society (Regensburg)* p 1207

[16] Ślawska-Waniewska A, Gutowski M, Kuźmiński M, Dynowska E and Lachowicz H K 1994 Microstructure and magnetic properties of nanocrystalline Fe–Cr-based metallic glasses *Nanophase Materials (NATO ASI Series* **E260***)* ed G C Hadjipanayis and R W Siegel (Deverter: Kluwer) p 721

[17] Bean C P and Livingston J D 1959 Superparamagnetism *J. Appl. Phys.* **30** 120S

[18] Ślawska-Waniewska A, Gutowski M and Lachowicz H K 1992 Superparamagnetism in a nanocrystalline Fe-based metallic glass *Phys. Rev.* B **46** 14594

[19] Yoshizawa Y, Oguma S and Yamauchi K 1988 New Fe-based soft magnetic alloys composed of ultrafine grain structure *J. Appl. Phys.* **64** 6044