Langevin granulometry of the particle size distribution

J. Phys. A: Math. Gen. **37** (2004) 6027–6041 PII: S0305-4470(04)73100-5

Langevin granulometry of the particle size distribution

Attila Kákay[1], M W Gutowski[2], L Takacs[3], V Franco[4] and L K Varga[1]

[1] Research Institute for Solid State Physics and Optics, 1525 Budapest PO Box 49, Hungary
[2] Institute of Physics, Polish Academy of Sciences, Al. Lotnikow 32/46, 02-668 Warsaw, Poland
[3] Department of Physics, University of Maryland, Baltimore County, Baltimore, MD 21250, USA
[4] Department of Condensed Matter Physics, Seville University, PO Box 1065, 41080 Seville, Spain

E-mail: attilak@szfki.hu

Received 4 December 2003, in final form 8 March 2004
Published 25 May 2004
Online at stacks.iop.org/JPhysA/37/6027
DOI: 10.1088/0305-4470/37/23/005

Abstract
The problem of deriving the particle size distribution directly from superparamagnetic magnetization curves is studied by three mathematical methods: (1) least-squares deviation with regularization procedure, (2) simulated annealing and (3) genetic algorithm. Software has been developed for the latest versions of all these methods and its performance compared for various models of underlying particle size distributions (Dirac δ-like, lognormal- and Gaussian-shaped). For single peak distributions all three methods give reasonable and similar results, but for bimodal distributions the genetic algorithm is the only acceptable one. The genetic algorithm is able to recover with the same precision both the lognormal and Gaussian single and double (mixed) model distributions. The sensitivity of the genetic algorithm—the most promising method—to uncertainty of measurements was also tested; correct peak position and its half width were recovered for Gaussian distributions, when the analysed data were contaminated with noise of up to 5% of M_S.

PACS numbers: 02.60.Pn, 02.70.−c, 75.20.−g, 75.50.Tt

1. Introduction

Modern demands have rendered magnetic nanoparticles increasingly important. Non-interacting nanoparticles are used in ferrofluids, high frequency inductive elements, hard disk recording media and biomedical applications. Interacting assemblies of nano-scale magnetic grains make up permanent magnets and nanocrystalline soft magnetic materials. In many cases the interaction between magnetic particles can be overcome by increasing the temperature well above the superparamagnetic transition temperature, T_{sp}, where the magnetization curves collapse to a single curve that is the superposition of Langevin functions. The shape of this curve depends only on the distribution of particle sizes and hence it can be used as a magnetic or 'Langevin granulometer' to determine this distribution. Although several other techniques are available for the determination of particle size distributions, such as transmission electron microscopy, small angle neutron scattering, etc, the magnetic method is easier to perform experimentally. Of course, this technique presents the same difficulty as many other problems in data evaluation, namely the physical system uniquely determines the measured data but the inverse problem of deriving the parameters of the physical system from the data is ambiguous. In this paper we study this ambiguity, applying three different algorithms in order to derive the particle size distribution from the superparamagnetic magnetization curves. As far as we know such a comparison has not been done before. The following optimization techniques were adapted for this problem and compared: (1) regularization method, (2) simulated annealing and (3) genetic algorithm.

2. Statement of the problem

Let us consider a ferrofluid or a nanocomposite in the superparamagnetic state. The magnetization curve of a single superparamagnetic particle is described by the Langevin function, reflecting the competition between the thermal and Zeeman energy terms:

$$m_i = \mu_i \,\mathrm{Lang}\left(\frac{\mu_0 \mu_i H}{k_B T}\right) \qquad i = 1, \ldots, N \tag{1}$$

(*the Langevin function* $\mathrm{Lang}(x) = \coth(x) - 1/x$), where N is the number of different types of particles, μ_i is the magnetization of a particle of type i, H is the magnetic field strength, μ_0 is the permeability of the vacuum, T is the absolute temperature and k_B is the Boltzmann constant. The particles are assumed to be single domain. The magnetization of a sample that contains a distribution of particle types is given by

$$M(H, T) = \sum_{i=1}^{N} w_i \mu_i \,\mathrm{Lang}\left(\frac{\mu_0 \mu_i H}{k_B T}\right) \tag{2}$$

where $w_i = w_i(\mu_i)$ is the number of particles of type i, per unit volume, each carrying a magnetic moment equal to μ_i.

The task is to determine the distribution of the magnetic moment, $w_i(\mu_i)$—directly related to the particle size—from the experimental magnetization curve, $M(H, T)$.

The fitting procedure can be initiated without assuming anything about the shape of $w_i(\mu_i)$, when using the regularization method. Alternatively, one can assume the particular shape of the searched distribution, parametrize it and then gradually improve the values of unknown parameters, initially estimated only very roughly or even just guessed, by—for example—simulated annealing (SA) or genetic algorithm (GA) method.

Each fitting procedure begins with determining the magnetic moment of the largest particles. Such an estimate can be obtained from the initial slope of the magnetization curve:

$$\mu_{\max} = \frac{3k_B T}{\mu_B M_S} \cdot \left.\frac{\partial M}{\partial H}\right|_{H\to 0}. \tag{3}$$

When defining the interval of the possible magnetic moments, the above value should be multiplied by a factor of 2 to 4, to make sure that even the largest particles are indeed presented in the sought distribution. This is because the largest particle predominantly, but not exclusively, determines the initial slope and therefore equation (3) underestimates the value of $\mu_{\max}$. The interval [0, $4\mu_{\max}$] is then divided into N bins of equal width, i.e. $\Delta\mu = 4\mu_{\max}/N$, and $\mu_i = i \cdot \Delta\mu$, $i = 1, \ldots, N$.

3. Description of the algorithms

3.1. Regularization method

This method is based on minimizing the sum of the squared deviations, defined as

$$\sigma = \sum_{m=1}^{L} \left[M_{\mu_i}^{w_i}(H_m) - M_m\right]^2 \tag{4}$$

where $M_{\mu_i}^{w_i}(H_m)$ is the calculated and M_m is the experimental magnetization ($m = 1, \dots, L$ being the numbers of experimental points). The necessary condition for extremum

$$\frac{\partial\sigma}{\partial w_j} = \sum_{m=1}^{L} 2\left[\sum_{i=1}^{N} w_i\mu_i \,\mathrm{Lang}\left(\frac{\mu_0\mu_i H_m}{k_B T}\right) - M_m\right]\mathrm{Lang}\left(\frac{\mu_0\mu_j H_m}{k_B T}\right) = 0 \tag{5}$$

$j = 1, \dots, N$, leads to a system of N linear equations for the unknowns w_i. Straightforward solution of this equation system leads to unacceptable values of w_i, with large negative and positive fluctuations in response to small variations of M_m ($m = 1, \dots, L$). Such a solution has no physical meaning and an additional constraint is necessary to stabilize the solutions. The method of Weser and Stierstadt [1] based on Tikhonov regularization [2] is adapted here. A 'regularization term' is added to the sum of the squared deviations, σ, and the modified sum of the squared deviations is minimized. The regularizing operator used by Weser and Stierstadt is

$$\rho = \sum_{i=1}^{N} w_i^2 \tag{6}$$

and the expression $\sigma' = \sigma + \alpha\rho$ is minimized, where $\alpha > 0$ is the regularizing parameter. By requiring a small value of ρ, the unwanted oscillations of w_i are suppressed, while the modified system of equations becomes

$$\frac{\partial\sigma'}{\partial w_j} = \sum_{m=1}^{L}\left[\sum_{i=1}^{N} w_i\mu_i \,\mathrm{Lang}\left(\frac{\mu_0\mu_i H_m}{k_B T}\right) - M_m\right]\mathrm{Lang}\left(\frac{\mu_0\mu_j H_m}{k_B T}\right) + \alpha w_j = 0 \qquad j = 1, \dots, N. \tag{7}$$

The random oscillations of w_i can also be limited by requiring small values of the first and second derivatives of the sum of the squared deviations, as suggested in [3]. The expression to minimize is $\sigma' = \sigma + \alpha p + \beta q + \gamma r$, where

$$p = \sum_{i=1}^{N} w_i^2 \qquad q = \sum_{i=1}^{N-1}(w_{i+1} - w_i)^2 \qquad r = \sum_{i=1}^{N-2}(w_{i+2} - 2w_{i+1} + w_i)^2 \tag{8}$$

and α, β and γ are regularizing parameters. The choice of these parameters is rather arbitrary as long as the $\sigma' < 2\sigma$ condition is satisfied and the best choice depends on the quality of the measured data, the shape of the particle size distribution, etc. Reasonable results were obtained by using terms with p and r only [3].

The $\alpha = \beta = \gamma = 0$ assumption takes us back to the ill-posed problem. For small values of the regularization parameters, one can obtain a precise representation of the magnetization curve, but the probability distribution is unphysical, whereas larger values of α, β and γ make the distribution smooth at the cost of increasing σ', i.e. the quality of the fit becomes worse. A reasonable compromise has to be found between these two extremes. The probability distribution is obtained by solving the set of linear equations provided by the minimizing process.

3.2. Simulated annealing method

As far as we know, the simulated annealing method has never been applied to Langevin granulometry. In principle, this is a Metropolis Monte Carlo algorithm, proposed by Metropolis and co-workers in 1953 [4] and modified by Kirkpatrick and co-workers in 1983 [5]. According to the principle of the method, the evolution of the distribution function, w_i, is imitated by the evolution of a thermodynamic system. The transition from the initial state with energy E_1 to another state with energy E_2 is governed by a probability $p = \exp[-(E_2 - E_1)/k_B T]$, where T is a controlling parameter and k_B is the Boltzmann constant. The system goes to state E_2 with the probability p if $E_1 \leqslant E_2$ and with probability 1 when $E_2 < E_1$.

The algorithm starts from an initial distribution assumed to be the uniform distribution in our calculation, i.e. $w_i = 1/N$ at the start of the algorithm. The object of interest is the distribution function $w_i\ (\mu_i)$. For this $w_i(\mu_i)$ we calculate the sum of absolute deviations, F^{ini}, with the following relation:

$$F = \sum_{m=1}^{L} \left| M_{\mu_i}^{w_i}(H_m) - M_m \right|. \tag{9}$$

Then, from the set of $w_i(\mu_i)$ we randomly choose one bin and with a probability of 1/2 we either increase or decrease the initial value by a preset fixed number (the smallest step in the w coordinate of the histogram). After this we calculate the sum of absolute deviations, F^{next} and then $\Delta F = F^{next} - F^{ini}$. The new distribution is always accepted, if $\Delta F < 0$. The new distribution can also be accepted when $\Delta F > 0$, if only $\exp(-\Delta F/T) > q$, where q is a random number between 0 and 1, and T is the 'annealing' parameter. For a given T, we make 1000–2000 Monte Carlo steps, then we decrease T by a factor of 0.98 and the procedure is repeated. Gradually decreasing T ensures that the algorithm converges to a stable solution. For the kernel of our program the subroutine given in [6] was used.

3.3. Genetic algorithm

Genetic algorithms (GAs) have recently been devised in the field of artificial intelligence and have been used for optimization problems. According to Gallagher, this algorithm is far superior in performance to Monte Carlo techniques [7]. The method was pioneered by John Holland in the early sixties [8]. The progress of the field was summarized by Grefenstette [9] and Goldberg [10]. Starting with the mid-eighties, genetic algorithms have received increased attention from problem-solving practitioners in numerous areas of applied sciences. For treating the problem of Langevin granulometry, this algorithm was first applied by Gutowski [11].

The genetic algorithm operates indirectly on the population of trial solutions to the given optimization problem. Every solution, called a *phenotype*, is represented by a string of symbols, usually of fixed length, called a *chromosome*, a *genotype* or just an *individual*. Chromosomes and pairs of chromosomes are the only objects which are explicitly manipulated by the algorithm. Therefore the genotypes should be thought of as the coded prescriptions for the trial solutions, not being the solutions *per se*. The coding of chromosomes is such that any given genotype determines uniquely the phenotype, which can be derived from it. A chromosome is an ordered sequence of smaller parts, assumed indivisible, and called *genes*. Usually, but not always, a single gene corresponds to a single unknown. The chromosomes may exchange their genetic material by means of the crossover operation: the two individuals (called the parents) exchange parts of their genetic material (substrings of symbols), thus creating offspring—new members of the population. Each single individual can also be subjected to *mutation*—the operation of random change of one or more of its genes (single-character substrings). The evolution of the population—the search for optimal solution—progresses as a sequence of crossover and mutation operations, applied repeatedly. It is essential to devise a means to evaluate every member of the population in order to decide which one is better and which one is worse. The quality of an individual is called *fitness*. According to the Darwinian law of *survival of the fittest*, the better fitted individuals have higher chances to mate with others and thus to transfer their genetic material to the next generations. The detailed mechanism of simulated evolution, at the lowest level, involves the following basic steps:

genotype $\xrightarrow{\text{decoding}}$ phenotype $\xrightarrow{\text{evaluation}}$ fitness $\xrightarrow{\text{selection}}$ probability to become a parent

The probability of mutation is kept low, since a high rate of mutation would turn the GA into an ordinary Monte Carlo procedure. At the highest level, the evolutionary process can be described as presented in algorithm 1. The algorithm is fairly insensitive to the particular choice of tuning parameters, thus their definitions and meaning will be given in the following text.

Algorithm 1 Genetic algorithm with fixed-size population and asexual reproduction

```
Require: environmental data (the function being optimized); few tuning parameters
Ensure: the set of near-optimal points in the search space
 1: create old population
 2: create empty new population
 3: evaluate all members of old population
 4: while termination criteria not met do
 5:   repeat
 6:     select two individuals as prospective parents
 7:     if mating successful then
 8:       perform crossover
 9:     end if
10:     try to mutate both chromosomes, crossed or not
11:     put both individuals into new population
12:   until new population is formed
13:   evaluate members of the new population
14:   replace old population with newly created one
15: end while
```

The low level mechanisms, except selection, are specific to the particular problem, while the general machinery (algorithm 1) is always the same. In the next section, the essential steps of GA are described as implemented in our problem.

3.3.1. Genotype coding. This coding is performed in a natural way: each chromosome consists of genes corresponding to the contents of histogram bins approximating the sought momentum distribution. Every gene is a positive integer number with the allowed number of bits (in binary counting system) being fixed, but not necessarily the same for every gene. To make sure that those numbers are always non-negative, we limit the number of bits assigned to every gene to 30, which is appropriate for 32-bit computers.

3.3.2. The transformation of genotypes into phenotypes. This corresponds to a smoothing tool, which produces smooth histograms from 'raw' genotypes, as proper candidates for solutions. The smoothing is done by convolution of the original, 'noisy' genotype with an appropriate chosen point-spread function $f(x)$, namely

$$f(x) = \begin{cases} \exp\left(\frac{-1}{1-(\|x\|/\lambda)^2}\right) & \text{for} \quad \|x\| < \lambda \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Here x is the distance between genes, i.e. the difference between the sequential numbers of histogram bins, and the value of the smoothing parameter, $\lambda > 0$, is chosen depending on the quality of experimental data. The double bar, $\|\cdot\|$, instead of just an absolute value, $|\cdot|$, is used here in order to indicate how smoothing is applied to multidimensional cases. After the convolution is executed, our phenotype becomes an ordered sequence of real numbers.

3.3.3. Evaluation of phenotypes. It takes two steps. In the first step, the phenotype is used as is to simulate the experimental data, following equation (2), with the ith gene playing the role of the unknown w_i. This produces the simulated magnetization curve, which is usually rather far from the collected experimental data. Fortunately one can scale all unknowns to satisfy (see equation (2))

$$\lim_{H\to\infty} M(H,T) = \sum_{i=1}^{N} w_i \mu_i = M_s \tag{11}$$

where M_s, the saturation magnetization, has to be estimated independently and only once. There is no need to recalculate the already simulated curve; it is sufficient to multiply it by the same scale factor. The second step is the calculation of the fitness according to formula (9).

3.3.4. Selection rules. The value of the fitness is a non-negative real number with unpredictable magnitude. A lower value indicates a better fit. To transform the fitness, F, of any given individual into probability, p, of being selected as a parent the formula

$$p(F) = \frac{1}{1+\exp\left(\frac{F-K_F}{S_F}\right)} \tag{12}$$

is used, with F given by equation (9). K_F is the median of the population fitness and S_F is, in our implementation, the standard deviation from the median of the population fitness. The positive scaling factor S_F may be chosen somewhat arbitrarily. Its precise value is not very important, but making it close to zero has the effect of 'hardening' the selection, i.e. individuals with fitness below the median are granted only negligible chances for reproduction, while those above the median will almost certainly become parents. On the other hand, increasing S_F results in forces $p(F) \to \frac{1}{2}$ for all members of the population, regardless of their individual fitness.

3.3.5. The initial population. It is created in an unbiased way. All researchers in the field agree that the initial population should be as random as possible. We build it bit by bit, except for chromosome number 1, which is set by the calling program and left unchanged at this stage. Each bit's value is set to either 0 or 1 with 50% probability. This is very important indeed, since a particular bit has the same value for every individual in the population, then 50% of search space will never be explored by the algorithm, unless a mutation changes this bit. Insufficient diversity of the population may be one of the reasons for the so-called *premature convergence*, when the algorithm becomes stuck in a certain region of the search space, not containing any extremal fitness at all.

There is also a question of how numerous should the evolving population be. For efficiency reasons, we would rather prefer a smaller population over a larger one. The lower bound for membership can be obtained in at least two ways [12]. Let n denote the number of individuals in the population and N_b the length of an individual chromosome, expressed in bits. Then either

- $n \geqslant \log_2 N_b - \log_2 p$, where p is the required probability that no bit is 'lost', after the construction of the initial population, or
- $n \geqslant \sqrt{2N_b}$.

Both numbers should be rounded up to at least the nearest even integer and the higher one should be used as the lower bound for the membership. The second number, usually higher, only says that it is impossible to explore the entire search space, using crossover operations exclusively, with a smaller number of individuals in the population. This is a necessary condition that may not be sufficient. In our case, with 256 unknowns, each being a 30-bit number (chromosome length is then 7680 bits), we need at least 124 individuals to reach the goal. We decided to establish a population of 256 members. This may seem in sharp contrast with many other papers on GAs, where larger populations are common, reaching sometimes up to 8192 members for problems involving only a few tens of unknowns. We never intentionally clone our chromosomes and therefore we do not see any reason to increase the size of the population significantly above the lower bound. This is not a very critical problem since a larger population may need fewer generations to reach the terminating criteria, leaving the total computational complexity, measured by the number of fitness evaluations, practically unaffected.

3.3.6. Parents and offspring. Whether any given chromosome can become a parent is decided by comparing a uniformly distributed random number on [0, 1] with the quantity p derived from an individual's fitness by equation (12). The second parent is selected the same way from the remaining candidates.

The probability of having offspring is one of the control parameters of GAs. Its value is usually chosen from between 20% and 95%, with no clear influence on the efficiency of the optimization procedure.

In our implementation this overall tuning parameter is set to 75%. To achieve this value in the statistical sense, the switching probability per bit is derived in the following way. The overall crossover attempt is divided into manipulations on corresponding genes, one from each parent. They are copied into the genetic material of two offspring straight or switched depending on a probability derived from the tuning. The switching occurs with a probability which can be derived from the value of our tuning parameter and the length of the gene. Since the genes may differ in length, it is convenient to precompute the switching probabilities at the beginning of the routine.

The above crossover operation returns a pair of unchanged chromosomes, a pair crossed at exactly one point, two points, etc. In a certain run we have obtained 408 pairs unchanged, 527 pairs crossed once, 411 at two points, 198 at three points, 65 at four points and 19 crossed at five or more points. As we can see, the fraction of 'childless couples' was 25.06% corresponding to the chosen 75% for the overall tuning parameter almost exactly.

3.3.7. Mutations. These are relatively rare events, in which a given gene is changed arbitrarily. The prevailing opinion is that mutation should be applied very carefully, since the majority of them are useless and they only waste computer resources. Yet there is another opinion saying that in reality mutations occur quite frequently, but remain unnoticed due to their small effect. Our mutation mechanism follows the second way of thinking.

There are two common sense limits for the mutation rate. The lower limit requires that at least one individual, on average, is mutated during one generation. Mutations are necessary to preserve the diversity of the population. They may also be considered as a kind of 'teleportation vehicle' for populations initially located very far from the optimum. The upper limit requires that no more than half of individuals can be mutated, otherwise the GA practically becomes a generic Monte Carlo routine.

Our implementation of mutation is such that statistically the number of mutated individuals is equal to the arithmetic average of both extremes. A routine computes the mutation probability per bit for each gene and stores the table of mutation probabilities per each gene. The chromosome subjected to mutation is processed on a gene by gene basis. First a decision is made, whether or not the given gene has to be mutated. If yes, then it is converted from its normal, binary form into Gray code. A single, randomly selected bit is then inverted and the result is transformed back into the usual form. Both transformations are fast in computer realization and do not increase the computational burden noticeably.

What are the advantages of such a complicated procedure? It turns out that mutations performed in this manner only rarely change the position of the chromosome in a search space significantly. The larger the jump the less frequently it occurs. This behaviour can be approximately described by the famous Zipf law [13]. Very small changes of location are the most frequent, thus ensuring the balance between large scale and local searching. On the other hand, the largest possible jump is nearly twice as large as that which could ever be achieved by mutating the gene in its original form. Of course, after the mutation attempt, we can retain the individual unchanged, with a single damaged gene, with two mutated genes, etc.

3.3.8. Termination criteria. The main loop of algorithm 1 is repeated until the median fitness of phenotypes no longer improves. This is recognized as a lack of progress during the last 15 generations (average fitness fluctuates around some value or even worsens).

3.3.9. Remaining comments. The best genotype from the previous generation is always saved, just in the unlucky event it were accidentally lost during the evolution process. If this ever happens, we forcibly re-introduce the saved genotype into the population in place of its worst member. We call this behaviour of the algorithm 'kingdom mode' in opposition to 'democracy mode', when the best individual is not protected. The best chromosome is always returned as a final result of the entire procedure.

3.3.10. Details of calculations. The calculation is usually started with a preliminary run using 4 as the expansion factor (section 2) and a large resolution λ (section 3.3.2). The objective is to derive the true support of the distribution. In the subsequent run, the expansion factor is adjusted to reliably cover the support of the distribution and the resolution is improved, i.e. the value of λ is decreased.

During the evaluation, the GA is executed several times. Each time chromosome number 1 is set to the smoothed version of the best individual obtained so far, while other members of the population are generated by the GA itself. The contents of each gene belonging to the best individual, except for the first and last gene, which are set to zero, is set to the arithmetic average of its two nearest neighbours. The number of bits assigned to each individual gene may change at this stage too. It is possible to exploit the available experimental information, expressed in bits, in such a way that it corresponded to the capacity of the histogram, also expressed in bits. To achieve this goal, we proceed as follows. The currently best phenotype (an ordered sequence of real numbers w_i) is converted back into its integer binary representation, the genotype number 1. This conversion requires a certain real multiplier ('*quantum*'), which has to be adjusted so that the sum of true lengths of all resulting genes does not exceed the experimental information. The true length of the *i*th gene is calculated as the integer part of the expression $\max(1, \lceil \log_2(w_i/\mathit{quantum}) \rceil)$, where $\lceil \cdot \rceil$ means the nearest integer not less than the argument. This procedure is by no means accurate, since the input information can be treated as a sum of contributions given by all experimental data, while this is not true for the information contained in the genotype because of its smoothing. The length assigned to each gene is increased by 2 before the next call to the GA routine. This trick allows for up to fourfold increase of (some) amplitudes w_i.

The results of a successful calculation cycle, i.e. when a new currently best genotype is discovered, are stored. Due to smoothing the best individual before each new iteration, the results cannot improve indefinitely. Three failures in a row (fitness worse than in the previous cycle), or lack of improvement in 12 consecutive cycles terminate execution.

It is worth mentioning that the GA does not need the derivatives of the objective function as the variants of the regularization methods do, and therefore it is easy to optimize with the GA non-smoothed objective functions, like the one given in equation (9). We prefer to work with absolute deviations rather than with their squares, because they are much less sensitive to outliers.

4. Applications to model distributions

In order to compare the performance of each method, we have to note that in the case of the regularization method, recovery of the unknown distribution curve relies on the optimization of three different parameters (α, β, γ) for each magnetization curve separately. For the regularization method, the best fit was obtained when the interval $[\mu_{\min}, \mu_{\max}]$ was divided into 30–50 channels. The support of searched distribution cannot start from 0 but instead at $\mu_{\min} > 0$, in order to get the best possible fit. When using a higher resolution for the recovery of the histogram, one obtains a noisy result, impossible to interpret from the experimental point of view, although the obtained fit in some cases is better. We have to mention that the regularization method often returns both negative and positive values for the unknown amplitudes w_i. In order to optimize the parameters α, β and γ we have to satisfy two requirements at the same time, namely (i) to find the global minimum for the least-squares deviation and (ii) the area contained in the bins with negative contents should be negligible compared to the positive contents (only the non-negative values have physical sense). Finding the proper choice of parameters requires several runs on the same experimental data and it is not efficient from the point of view of the recovering process. Moreover, the resolution obtainable by the regularization method is the worst among the methods treated in this paper.

For the SA method the cooling speed has to be adjusted. For a simple distribution the convergence is fast, but for a more complex case, a slower cooling process has to be chosen, increasing the calculation time. If the cooling process is too fast, the large-moment part of the distribution cannot be recovered properly.

4.1. Single and double peak Dirac δ-like model distributions

As the first examples, we considered samples containing only one or two discrete sizes of the magnetic moment, corresponding to single- and double-peak Dirac δ-like distributions. The corresponding 'experimental' magnetization (Langevin) curves of 1000 points were

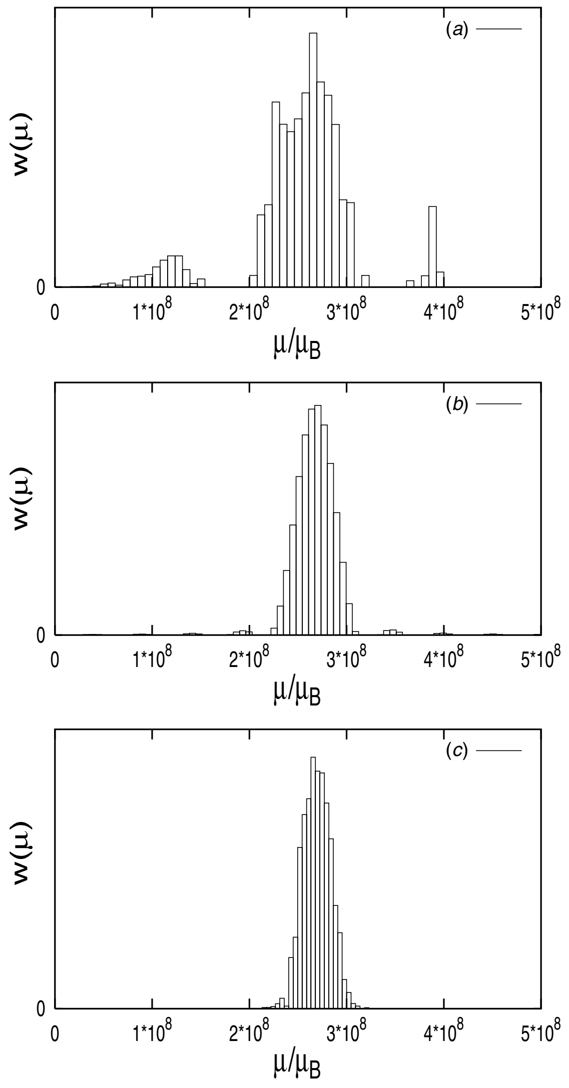

Figure 1. Distributions recovered from a simulated magnetization curve originating from a sample containing particles of equal moment by (*a*) regularization method, (*b*) simulated annealing and (*c*) genetic algorithm. The model peak position is at 2.7×10^8.

constructed, to be recovered later by the different algorithms. All three methods succeeded in recovering the initial distributions (results are shown in figures 1 and 2). None of the methods is able to provide an infinitely narrow peak, instead distributions with finite width are obtained.

The most promising method in these trials was the genetic algorithm; the results presented in the following were obtained using a thoroughly rewritten version of the earlier program [11].

4.2. Lognormal and Gaussian model distributions processed by genetic algorithm

We have generated superparamagnetic magnetization curves by using either a lognormal distribution or a Gaussian distribution of magnetic moments. In both cases, the number

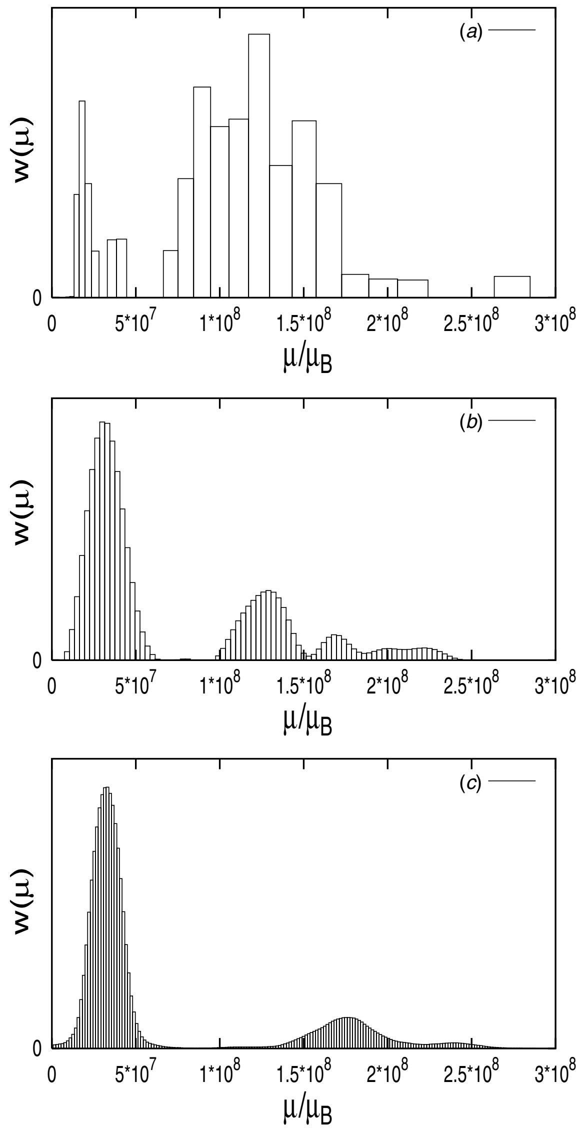

Figure 2. Recovered distributions from the magnetization curve of a sample containing a mixture of two magnetic moments with (*a*) regularization method, (*b*) simulated annealing and (*c*) genetic algorithm. Original peak positions were 3×10^7 and 1.8×10^8, with relative amplitudes of 1.0 and 0.3, respectively.

of points in the magnetization curve was 1000. The recovered distributions are shown in figures 3 and 4, respectively.

For further testing of the GA program, we have combined the previous two distributions in different proportions. The result of the computation is shown in figure 5.

Based on figures 3, 4 and 5, we can conclude that both the peak positions and half widths could be surprisingly well reproduced by the GA. For the composite distribution (lognormal plus Gaussian), the fit to the magnetization curve is shown in figure 6.

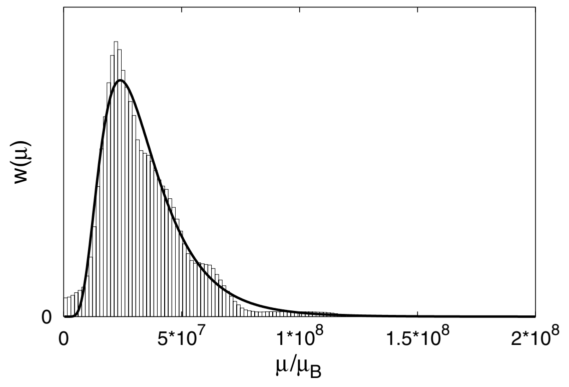

Figure 3. Simulated lognormal-like distribution of particles with small magnetic moment (solid line) used to generate a model magnetization curve and the distribution recovered by the genetic algorithm (histogram).

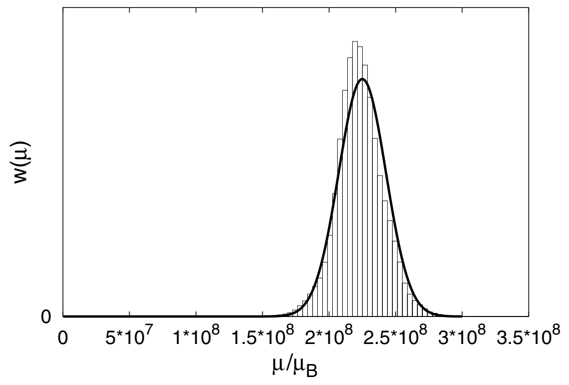

Figure 4. Simulated Gaussian-like distribution of particles with large magnetic moment (solid line) used to generate a model magnetization curve and the distribution recovered by the genetic algorithm (histogram).

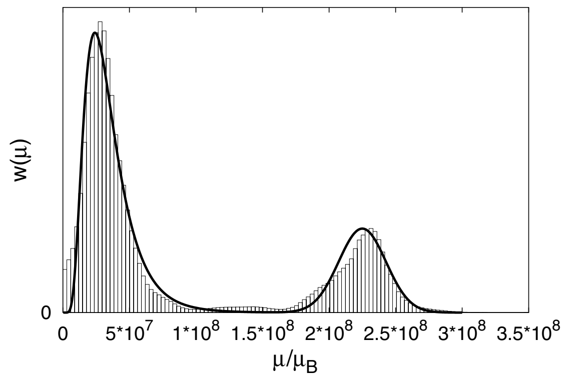

Figure 5. Simulated composite distribution, consisting of a lognormal and a Gaussian part (solid line), used for the generation of a model magnetization curve and the distribution recovered by the genetic algorithm (histogram).

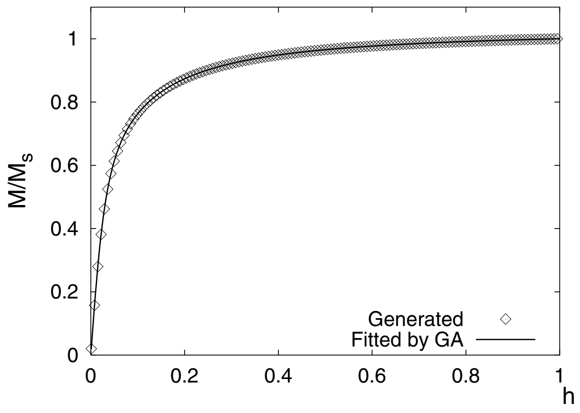

Figure 6. The simulated magnetization curve (diamonds), corresponding to the composite distribution consisting of a lognormal and a Gaussian part (shown in figure 5) together with the fit obtained by using the distribution recovered by the genetic algorithm (solid line).

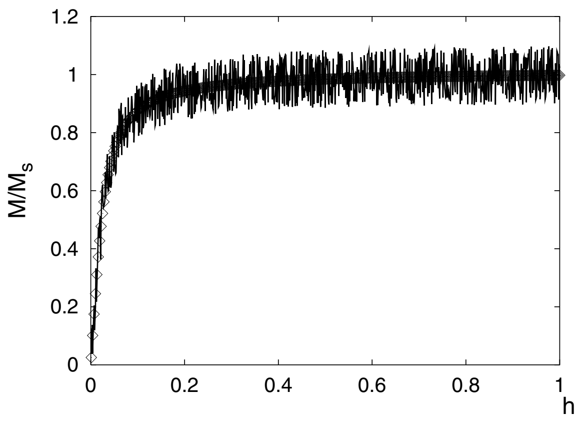

Figure 7. Generated magnetization curves with and without noise, assuming Gaussian distribution for the magnetic moments. The noise amplitude represents 20% of the M_S value.

The sensitivity of the recovered distribution to stochastic noise has also been studied. We have constructed magnetization curves—assuming Gaussian distribution for the magnetic moments—with the addition of a stochastic noise uniformly distributed in the $[-Amp, Amp]$ interval. The following numerical values have been considered for Amp: 0.1%, 0.2%, 0.5%, 1%, 2%, 5%, 10% and 20% of M_S. The noisiest magnetization curve is presented in figure 7.

In figure 8 the area of the two distributions, representing the saturation magnetization, is almost the same, the difference being less than 1%. One can conclude that in spite of the fact that the half width of the recovered peak is twice as large as the theoretical, the peak position is properly given. In practice the noise coming from measurements is always smaller than that in the extreme case (20% of M_S) considered above. Up to 5% noise (which is reasonable from an experimental point of view), both the peak position and the half width of the distribution have been properly recovered by GA. We can conclude that if the noise is smaller than 5%

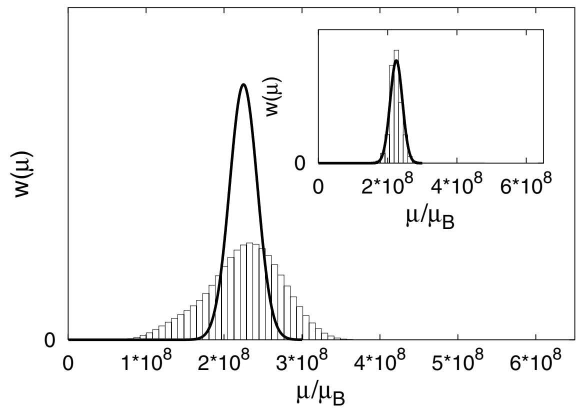

Figure 8. The histogram represents the distribution recovered by the genetic algorithm from the noisy magnetization data, which should be compared with the original Gaussian distribution (solid line), from which the noiseless magnetization curve was created. For comparison, the inset shows the distribution recovered from noiseless data (same as figure 4).

of M_S, the GA is still able to recover the initial distribution, but measurements with larger uncertainty are not suitable for the determination of the particle moment distribution.

Based on experience obtained during the recovery processes above, we conclude that selecting 2 as the expansion factor of the maximum moment and $\lambda = 20$ as the smoothing parameter allows for the recovery of a variety of model distributions with acceptable deviations.

This version of the GA combines the essential features of the two other methods described in this paper, namely the concept of regularization and the stochastic nature of SA. The advantages of GAs over other methods are that the recovered magnetic moment distributions are strictly non-negative; they have only one adjustable parameter, λ (resolution), with clear physical meaning, related to the signal to noise ratio of the experimental data.

All the presented calculations have been performed on two AMD ATHLON XP DUAL CPU computers with the clock rate of 1.2 GHz for each CPU. The typical execution time for a magnetization curve consisting of 1000 points, and using a 256-bin histogram for the recovered distribution, was around 1 h.

5. Conclusions

Comparing the three methods studied in this paper, the genetic algorithm is by far the best for extracting the magnetic moment distribution from the superparamagnetic magnetization curves. The comparison has been done using artificially created distributions: single-modal and multi-modal distributions with Gaussian and lognormal shapes. The genetic algorithm, operating on discrete variables, is able to reproduce the smooth, multi-modal distributions, recovering correctly and reliably all their essential features, such as peak positions and widths at half maximum. Exact recovery of model distributions, however, cannot be expected, because depending on the given smoothing parameter, some wiggles can appear distorting the results. By proper selection of the expansion factor for the domain of the magnetic moment distribution and by appropriate choice of the resolution, a variety of model distributions can be reproduced with acceptable deviations. Consequently, the genetic algorithm method can be recommended to process superparamagnetic magnetization curve data. Application for the evaluation of experimentally measured magnetization curves of nanostructured systems is in progress. The executable files are available on the corresponding author's Web page [14] and the source code will be sent on request.

Acknowledgments

This work was supported by the EU project G5RD-CT-2001-033009 and by the Hispano–Hungarian Bilateral Cooperation Program (TéT E-2/2001). MWG gratefully acknowledges the generous support of his one month stay in Budapest by the Centre of Excellence CELDIS, the European Commission program ICA1-CT-2000-70018.

References

[1] Weser T and Stierstadt K 1985 *Z. Phys.* B **59** 253
[2] Tikhonov A N and Arsenin V Y 1977 *Solutions of Ill-Posed Problem* (New York: Wiley)
[3] Takacs L 1988 *J. Appl. Phys.* **63** 4264
[4] Metropolis N *et al* 1953 *J. Chem. Phys.* **21** 1087
[5] Kirkpatrick S *et al* 1983 *Science* **220** 671
[6] Press W H *et al* 1992 *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge: Cambridge University Press)
[7] Gallagher K *et al* 1991 *Geophys. Res. Lett.* **18** 2177
[8] Holland J H 1975 *Adaptation in Natural and Artificial Systems* (Ann Arbor, MI: University of Michigan Press)
[9] Grefenstette J J *et al* 1985 *Proc. 1st Inst. Conf. Genetic Algorithms and Their Applications* vol 1 (Pittsburgh, PA: Lawrence Erlbaum Associates) p 160
[10] Goldberg D E 1989 *Genetic Algorithms in Search, Optimization and Machine Learning* (Reading MA: Addison-Wesley)
[11] Gutowski M W 1994 *J. Phys. A: Math. Gen.* **27** 7893
[12] Gutowski M W 1999 Genetic algorithms in matrix formulation *Proc. 3rd Domestic Conf. Evolutionary Algorithms and Global Optimization (Potok Złoty, Poland)* pp 123–30 (in Polish)
[13] Zipf G K 1949 *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology* (Cambridge, MA: Addison-Wesley)
[14] http://www.szfki.hu/~attilak/software.html