Multi-Level Approaches to Discrete-State and Stochastic Problems

Achi Brandt* and Dorit Ron∗

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, Israel

and

Daniel J. Amit**

Racah Institute of Physics
The Hebrew University
Jerusalem, Israel

Abstract

Fast multi-level techniques are developed for large-scale problems whose variables may assume only discrete values (such as spins with only "up" and "down" states), and/or where the relations between variables is probabilistic. Motivation and examples are taken from statistical mechanics and field theory. Detailed procedures are developed for the fast global minimization of discrete-state functionals, or other functionals with many local minima, using new principles of multi-level interactions. Tests with Ising spin models are reported. Of special interest to physicists are the Ising model in a random field and spin glasses, which are known to lead to difficulties in conventional Monte-Carlo algorithms.

Content

* Research supported by the Air Force Aeronautical Laboratories, Air Force Systems Command, United States Air force, under Grant AFOSR 84-0070.

** Research supported in part by the Fund of Basic Research of the Israel Academy of Science and Humanities.

1. Introduction

The present effort has been motivated by the recognition that while Monte-Carlo studies of the statistical properties of systems in statistical mechanics and in field theory have become a central methodological tool, on par with analytical and experimental investigations, there is an increasing number of systems for which this tool is too slow. In fact, for some problems of central interest in both fields, present day computers seem inadequate to obtain significant results in a reasonable amount of time. To mention some such systems one recalls the computation of masses in quantum chromodynamics [22]; the phase structure of spin-glasses or magnets in random magnetic fields [1]; or problems of combinatorial optimization, of which the travelling salesman is a simple representation [9].

Most of the current effort to deal with this difficulty is concentrated in the development of new hardware, more atune to the specific Monte-Carlo task [24]. Only modest attention has been devoted to the improvement of the algorithm. In fact, in the thirty years since the invention of the technique [23], almost all improvements have been concerned with the speeding up of the basic algorithm, improvements which have increased the specialization to particular problems. This line of development has culminated with the microcanonical technique [7].

Another line of development has been the shift from the original idea [23] of generating a sequence of configurations, with the correct probability distribution – a Fokker-Planck approach [29], to a Langevin approach, in which one solves stochastic time dependent differential equations for the various physical quantities [11], [26].

The closest, in spirit, to the class of algorithms to be presented here, has been the Monte-Carlo renormalization group approach [19], [30]. Yet, in this approach configurations are still generated in the original Monte-Carlo process, even though physical quantities are computed in each configuration on different scales (to produce renormalized coupling constants [19] or renormalized expectation values [30]).

All the techniques mentioned above suffer from two basic deficiencies: slow transitions and slow balancing of deviations (see Sec. 2.3 below). By a proper modification of the statistical technique (cf. Sec. 2.3) the latter deficiency can be reduced to the first, and the origin of both is traced to the purely local nature of the classical algorithms.

This situation strongly suggests the need to introduce multi-level processes. Such processes (multigrid methods) have in the past resolved quite similar difficulties in the solution of partial differential equations, by combining local processing on different scales, employing interactions between several grids of widely varying meshsizes (see [10] and references therein).

One cannot, however, readily use classical multigrid to solve the problems of statistical mechanics and field theory. These problems differ from the differential problems in two basic aspects:

(A) Discrete state. In differential problems each variable (the value of the unknown solution at each site) is a real number (or a real vector), hence the entire solution can be regarded as a linear combination of components of varying scale (e.g., varying wavelengths), each of which is mainly treated by processing on that scale (i.e., on a grid with a comparable meshsize), more or less independently of the processing on other levels. In statistical physics, by contrast, and in many other important fields such as combinatorial optimization and pattern recognition, the variables are often either discrete or at least have discrete components. The simplest examples are Ising spin problems, in which each variable can only assume the values 1 and -1, or optimization problems with various dichotomous decisions. In such problems the solution is no longer a combination of more or less independent scales. As a result the inter-scale interactions are more intricate, and a good deal more interesting.

Indeed, the technique of inter-scale interactions for discrete-state problems is the main finding of the present study from the point of view of multigrid methodology. This technique can also provide a better treatment for highly nonlinear *continuous*-state problems, such as minimization problems plagued with too many local minima (see Sec. 4).

(B) Statistical nature. The purpose of calculations in differential equations is to find an approximate solution for one particular set of data (forcing terms, boundary conditions, etc.). In statistical physics the data are random, and/or the solution is stochastic (governed by probabilistic relations), hence the aim is not only to obtain (probable) solutions, but mainly to calculate their statistical properties. Consequently, in addition to the traditional role of multigrid in accelerating numerical processes toward equilibrium (here meaning toward highly probable configurations), it should here also provide for fast statistics. At low temperatures (low stochasticity) this means providing for fast transition between approximate ground states. At higher temperatures this also means large-scale balancing of deviations, to allow fast reliable convergence of long-range correlations (necessary for example for renormalization calculations).

The easiest approach is first to deal with these statistical aspects in separation from the discrete-state feature, by studying a statistical model with continuous variables, such as the XY model, or the Heisenberg model, etc. In such models the discrete (or highly nonlinear) nature of the problem appears only at much coarser levels (e.g., at the scale of probable vortices or instantons), so that on finer levels multigrid processes similar to the classical ones can be used.

The present paper deals mainly with the discrete-state aspect, and chiefly at low temperatures, where it is clearly separated from statistical elements.

2. Model Problem and Classical Solution Techniques

2.1 The physical problem

We will discuss the algorithms in the context of nearest neighbor Ising models, with uniform or random interactions and possible site dependent external magnetic fields. The discussion will further be restricted to two dimensional lattices, for which the new algorithms have been tried. All these restrictions are not essential. The logic, as well as the effectiveness of the algorithms should not be much affected if the number of local states becomes greater than two; if the interactions involve more neighbors than just the nearest ones; or if the number of lattice dimensions becomes greater than two.

The framework is defined by an $L \times L$ lattice, whose sites are labelled by an index i. At each site i there is an *Ising spin*, or briefly a "*spin*", i.e., a variable S_i which can assume one of the two values: $+1$ and -1. Thus, the total number of spins is $N = L2$, and the lattice has $2N$ possible *configurations* (arrangements of spin values).

With each configuration $C = \{S_i\}$ we associate the "*energy*"

$$E(C) = -\sum_{<i,j>} J_{ij} S_i S_j - \sum_i h_i S_i \tag{2.1}$$

where $< i, j >$ denotes a "*bond*", i.e., a pair of nearest neighbor sites, so the first summation is over all such bonds, while the second sums over all the lattice sites. J_{ij} and h_i are given parameters, representing, respectively, the couplings and the external magnetic fields.

In principle, the energy (2.1) should be supplemented by boundary conditions. Instead, for convenience, periodic geometry is often used, namely one sets the lattice on a torus. Thus, if the coordinates of site i are denoted (x_i, y_i), where x_i and y_i are integers and $1 \leq x_i \leq L$ and $1 \leq y_i \leq L$, then i and j are called nearest neighbors iff there exist integers x_{ij} and y_{ij} such that $|x_i - x_j + x_{ij}L| + |y_i - y_j + y_{ij}L| = 1$.

At zero temperature, this physical system settles to a "*ground state*", that is, a configuration with the lowest possible energy. The computational task is to find such a ground state, or, if possible, all of them. More practically, what is usually given is some statistical properties of J_{ij} and h_j, and the goal of computations is to find statistical properties of the ground states.

At higher temperatures random fluctuations enter, and configurations other than ground states can be obtained, although the larger their energy the less probable they are. The probability $P(C)$ of obtaining a configuration C is given by the *Gibbs* (or *Boltzmann*) *distribution*

$$P(C) = e{-}\beta E(C)/Z(\beta) \tag{2.2}$$

where $1/\beta$ is the temperature (in suitable units), and $Z(\beta)$ is a normalization factor (the partition function) derived from the condition $\sum_C P(C) = 1$. Here the computational task is to calculate statistical properties of the system, such as the average magnetization

$$\bar{M} = \sum_C M(C) \cdot P(C)$$

where $M(\{S_i\}) = N{-}1 \sum_i S_i$. It is desired to calculate such averages as functions of temperature, with special interest in temperatures close to 0 (large β) and close to certain critical values (phase transitions).

Note the relation between the two computational tasks: at sufficiently low temperature (sufficiently high β), only ground states are probable; so to obtain the desired statistics, the ground states should effectively be found.

In most problems N is large. One is typically interested in N being anywhere between several hundreds to many many thousands. It is therefore impractical to gather the desired statistics, or locate the ground states, by scanning all the $2N$ configurations. Instead, Monte-Carlo methods are used.

2.2 Classical Monte-Carlo method

The Monte-Carlo method is a computational process which generates a sequence of configurations such that ultimately each configuration appears a number of times proportional to its physical probability (2.2). On some finite section of this sequence, the statistical calculations are done. The problem is of course to generate a feasible small section which is sufficiently representative in terms of measured statistics.

The classical Monte-Carlo generates the next configuration in the sequence from the previous one, by treating a single spin at a time. The spins are taken in some order; lexicographic, red-black and random orders are common. Each spin S_i, in its turn, is either flipped or left unchanged, according to one of the following two rules:

(1) *Heat bath rule*: Let C be the previous configuration and C' the one obtained from it by changing S_i to $-S_i$. The decision whether to choose C or C' as the next configuration is taken randomly (by some random number generator), with the "*transition probability*":

$$P(C \to C') = \frac{P(C')}{P(C) + P(C')} = \frac{1}{1 + e\beta \cdot \delta E_i} \tag{2.3}$$

where

$$\delta E_i = E(C') - E(C) = 2h_i S_i + 2\sum_j J_{ij} S_i S_j, \tag{2.4}$$

j running in the summation over the four nearest neighbors of i. Note, thus, that this transition probability is an easily calculable local quantity.

(2) *Metropolis rule* [23]: If $E(C\prime) < E(C)$ then flipping is performed; energy is lowered whenever possible. On the other hand, if $E(C\prime) \geq E(C)$ then flipping is decided randomly again, with the probability (2.3) assigned to it.

Any one of those rules is widely used. It is easy to show for each of them that, the only stationary distribution (i.e., the only distribution which stays unchanged by every Monte-Carlo step) is (2.2). Since any configuration can be reached from any other with positive probability, the elementary theory of Markov chains implies that ultimately (i.e., in a sufficiently long sequence) each configuration C will appear a number of times proportional to $P(C)$. Moreover, that theory actually implies that whatever configuration one starts with, the probability $P_n(C)$ of obtaining C at the n-th step tend to $P(C)$ as $n \to \infty$. When n is sufficiently large so that $P_n(C) \approx P(C)$, to the approximation needed in the desired statistics, we say that the system has reached *equilibrium*.

Since $P_n(C) \to P(C)$ for any large n, it is not really necessary to sample all the configurations in the sequence. Since each step treats another spin, it is indeed equivalent to sample (take into account) only the configurations obtained at the end of a full sweep (visiting once all sites).

2.3 Monte-Carlo slowness

To produce reasonable approximations to the desired statistics, the Monte-Carlo sequence of configurations usually needs to be very long. There are two intrinsic reasons for this slowness:

(A) Slow transition. This is a low temperature difficulty, which to an important degree also affects intermediate (e.g., critical) temperatures. At such temperatures, the important (i.e., more probable) configurations are locally stable; that is, in the space of configurations they lie near the bottom of extended "*attraction basins*". Each Monte-Carlo step leads just to a neighboring configuration, with higher probability of moving downhill (toward the bottom of the basin) than uphill. Hence, most probably, it will require a great number of steps to escape from a given attraction basin and visit a neighboring one. Moreover, all those neighboring attraction basins are likely themselves to be near the bottom of a larger-scale attraction basin, and so on. The process will therefore take a very long time both to equilibrate and to effectively sample the space, in order to produce reliable averages.

More specifically, for example in case of uniform ($J_{ij} \equiv 1$) Ising models not dominated by external fields (e.g., $|h_i| \ll 1$), at low temperatures the statistically important configurations are made of large blocks of mostly aligned spins. These blocks can only slowly be changed by single spin flips. At very low temperatures such flips cannot change such blocks at all – the familiar experience of "walls".

Near critical temperatures changes are still probable, but they only slowly affect the large-scale blocks, exactly those blocks whose statistics is important for studying critical phenomena. The equilibration time is proportional to $\xi 2$, where ξ, the correlation length, tends to infinity as the temperature approaches the critical point – the familiar "critical slowing down".

This difficulty is even more acute in disordered frustrated systems, such as spin-glasses ($J_{ij} = \pm 1$ in some random manner) and magnets in quenched random fields. A spin-glass has very large number of essentially degenerate low-lying states (this number diverges as $N \to \infty$), and crossing the barriers between them by single-spin flips can take anywhere from short to infinitely long times. This is the origin of such phenomena as the remenant magnetization and irreversible cooling [12], [13], [20], [21], [27], [28].

(B) Slow balancing of deviations. Even when equilibrium has been reached, accumulation of precise statistics is slowed down by the presence of random deviations, whose effect on the desired averages decreases only as $n-1/2$, where n is the number of sampled configurations. This slowness appears at any temperature, including very high ones. For example, even for uniform ($J_{ij} \equiv 1$) and homogeneous ($h_i \equiv 0$) Ising models at very high temperatures ($\beta \ll 1$), to obtain $\bar{M} = 0$ to an accuracy of $10-6$ would require some 1012 Monte-Carlo steps.

The high temperature aspect of this trouble is easy to overcome, by changing the order through which averages are calculated. For example, the average magnetization can be rewritten in the form

$$\bar{M} = N-1 \sum_{i} \sum_{C(i)} P(C(i))[P_{i+}(C(i)) - P_{i-}(C(i))]$$

where $C(i)$ are spin configurations on the lattice excluding site i, and $P_{i+}(C(i))$ is the conditional probability of getting $+1$ at site i given $C(i)$, and $P_{i-} = 1 - P_{i+}$. Since P_{i+} is readily calculable, and since each $C(i)$ appears in the Monte-Carlo process a number of times proportional to $P(C(i))$, $\bar{M}$ can be calculated by averaging, over the sampled configurations C, the quantity $\sum_i \bar{S}_i(C)/N$, instead of $\sum_i S_i(C)/N$, where $\bar{S}_i(C) = 2P_{i+}(C(i)) - 1$ and $C(i)$ here is the configuration C without its spin at i. It is easy to show that the standard deviation in averages calculated in this way is $O(\beta)$ times the standard deviation in the original Monte-Carlo averages. By excluding more sites in the definition of $C(i)$, this factor can be reduced to $O(\beta m)$, for any prescribed integer m, but the work of computing the conditional probabilities grows exponentially in m.

Thus, for high enough temperatures, the statistical deviations can instantly be eliminated. At lower temperatures the above device will no longer work: averaging each spin value in its given neighborhood eliminates deviations related to single spin fluctuations, but does little to eliminate fluctuations of *blocks* of spins. To eliminate those, blocks of spins should be weighted against their chance of being simultaneously flipped.

The question is which blocks to use. There are certainly too many to be all included. Only the more highly probable block flipping should therefore be considered. Those, however, yield configurations which lie at the bottom of different attraction basins, thus requiring long transition computations.

So, in a sense, the second problem (slow balancing of deviations) can be reduced to the first one (slow transition). Since this is basically a low temperature (high β) problem, we have decided to confront it directly by first studying the difficult case of zero temperature ($\beta \to \infty$). The next sections describe this study, returning to the case of finite β in the conclusion.

3. Approaches to the Minimization Problem

At zero temperature, the problem is to quickly find ground states, i.e., configurations C with minimal energy $E(C)$. If there is more than a single ground state, the problem is also to have fast transition from one to another, so as to be able to calculate any desired statistics.

Special cases of the minimization problem have been solved by showing their equivalence to certain graph-theoretic problems for which polynomial-time algorithms are known. The Ising spin-glass problem ($J_{ij} = \pm 1$ with equal probability; $h_i = 0$) in *two dimensions* has been shown to be equivalent to the Chinese Postman problem [2], [5], [6], [8], while the ferromagnetic Ising model ($J_{ij} = 1$) with arbitrary (e.g., randomly generated) magnetic field in *any* dimension has been reduced to the Min-cutset problem in graph theory [3], [4]. In both cases the resulting algorithms produce ground states in $O(N3)$ computer operations.

Although this complexity is not intractable, and allowed the production of many model results, it is still too large to be easily applied, especially to three and four dimensional problems. More importantly, this graph-theoretic approach is limited to very particular model problems; it does not provide the general computational tool needed in statistical mechanics and field theory. By contrast, the multi-level stochastic algorithm developed below is very general, and its typical complexity is $O(N3/2)$. Indeed, the main significance of the graph-theoretic approach is that its model results can be used in testing the more general algorithms (cf. Sec. 6.1).

3.1 Point-by-point minimization

A natural computational process that immediately suggests itself when one wants to minimize a function of many variables is the *point-by-point minimization*, or *point-by-point relaxation*, which is actually the zero temperature limit of the Monte-Carlo process described above. The process involves scanning the variables one by one in some prescribed order. Each variable in its turn is changed to the value which brings about the greatest reduction in the energy. For the spin systems discussed here, this means scanning the spins, flipping (changing the sign of) each one in its turn, if and only if that flip reduces the energy.

For *continuous*-state (continuous S_i) problems, this point-by-point relaxation converges very slowly. (Classical multigrid methods were developed to overcome exactly this slowness.) In the present case of *discrete* states ($S_i = \pm 1$), a much more acute difficulty is that this simple algorithm will in all probability get trapped in one or other "*local minimum*", that is, a configuration for which no single spin flip can reduce the energy, but which is not a ground state.

For example, consider the uniform ($J_{ij} \equiv 1$) and homogeneous ($h_i \equiv 0$) Ising model, with periodic boundary conditions. The only ground states are clearly the uniform spin up configuration $\{S_i \equiv 1\}$ and the uniform spin down configuration $\{S_i \equiv -1\}$. A clear *local* minimum is any configuration having some of its rows with spins up and the rest with spins down. Clearly in such a configuration, flipping any individual spin, or even any small group of spins, would only raise the energy.

3.2 Stochastic minimization. Annealing

One way to escape local minima is to introduce a certain "*artificial temperature*" $1/\beta$ into the system. Using the Metropolis algorithm with that β will still flip in its turn any spin S_i for which that flip reduces the energy (i.e., for which $\delta E_i < 0$; cf. (2.4)), but will assign the positive probability (2.3) for flipping, even when the energy is thereby increased ($\delta E_i > 0$). In the context of the minimization (zero temperature) problem such a process is called *point-by-point stochastic minimization* or *point-by-point stochastic relaxation.*

To approach the real minimum one cannot of course keep the fluctuations. Thus, the artificial temperature should gradually be decreased toward 0 (β gradually raised to infinity). This process is called *(simulated) annealing*, and has been introduced into the context of minimization problems by S. Kirkpatrick [16], [17]. The rate at which the temperature is lowered, or the *annealing schedule*, is crucial and sensitive. Although the algorithm will most likely escape bad local minima, it is still very likely to be trapped in larger-scale attraction basins.

The situation can more specifically be understood in terms of the lattice geometry. Suppose that a certain, rather large block of spins is *reversed*, i.e., its flipping would decrease the energy, while flipping any of its individual spins or any smaller sub-block, would increase the energy. In case the block is "narrow" (e.g., one or two spin wide), stochastic relaxation has a very good chance of breaking it into smaller blocks. On the other hand, if the block is many spin wide, it can be broken only by the simultaneous flipping of a large number of spins. The probability for this to happen in any stretch of point-by-point stochastic relaxation sweeps (where higher probability is assigned to the original spin values) decreases exponentially with the width of the block, and thus becomes extremely unlikely – unless the artificial temperature is so large that the entire block structure is broken into pieces. Since eventually the temperature must get down, and since wide reversed blocks are very likely to form exactly at that stage (on large grids, especially on such sub-regions where the average local field runs counter to the spins of the ground state), the process gets easily trapped (cf. Sec. 6.2).

3.3 Block relaxation

Having observed that the main difficulty is associated with the need to flip wide blocks, multi-level approaches readily suggest themselves. The general idea of such approaches is to supplement the point-by-point minimization process with *block-by-block minimization* sweeps, on various scales. In such sweeps, *blocks* of spins are scanned, flipping each block in its turn (i.e., flipping simultaneously all its spins) if this decreases the energy. The energy difference in flipping any block Q is easily calculated by the formula

$$\delta E(Q) = 2 \sum_{<i,j> \in \partial Q} J_{ij} S_i S_j + 2 \sum_{i \in Q} h_i S_i \tag{3.1}$$

where ∂Q is the set of *bonds* (pairs of nearest neighbors) $< i, j >$ *around the boundary of* Q (i.e., i being inside Q and j outside Q), and S_i and S_j are the spin values *before* flipping. The main question, of course, is which blocks to scan. It is impractical to try all possible blocks, since their number grows exponentially with the number of spins.

The first approach, a natural extension to familiar multigrid PDE solvers, could be to simply divide the grid into 2×2 (then 4×4, etc.) disjoints blocks, and scan them in some prescribed order (lexicographic, or red-black, etc.). Such a naive approach is doomed to failure. Experiments will immediately show that no significant improvement, in fact very little change at all, is contributed by the block sweeps once the processed configurations are highly evolved (i.e., have undergone many point-by-point minimization sweeps, reaching, for example, a local minimum). Indeed, in such highly evolved configurations most bonds $< i, j >$ are *unviolated* (i.e., $J_{ij}S_iS_j > 0$). Hence, the flipping of an *arbitrary* block of spins is likely to violate most of the bonds around its boundary, and thus almost certainly *increase* the energy. There are only very *special* blocks whose flipping would reduce the energy, and these blocks will seldom coincide with any of those arbitrary square blocks chosen for processing.

Nor would it change much if "annealing" were used with these arbitrary blocks, that is, if *block stochastic relaxation* were tried on them. Such stochastic relaxation is easy to produce: The flipping probability of any block Q at a given artificial temperature $1/\beta$ is

$$P_f(Q) = 1/(1 + e\beta\delta E(Q)) \tag{3.2}$$

with $\delta E(Q)$ given by (3.1). But again, since the blocks are arbitrary, flipping them would almost always increase the energy so much as to make them highly improbable ($P_f \ll 1$; unless the artificial temperature is raised so much that *any* block will be flipped with sizeable probability, rendering the procedure useless).

3.4 Revised block relaxation

These observations directly lead to the central algorithmic idea of this research. It is clear that before deciding whether or not to flip any given block, the block itself should be modified so as to become a better candidate for flipping. That is, the flipped region should undergo a certain optimization. This can be done by point-by-point (possibly stochastic) minimization in the vicinity of the flipped region. This optimization will reduce locally, as far as possible, the number of violated bonds around the boundary of the block, thus making its boundary less arbitrary. The new boundary is in fact likely to be the best possible one in the neighborhood of the original, highly artificial boundary. We call this optimization process the *block revision process*; its result is called the *revised block.*

Only *after* this revision, should δE of the (revised) block be examined, and the decision whether actually to adopt this flipping be accordingly made. The decision may be either stochastic (based on (3.2) with some finite β) or deterministic ($\beta = \infty$). In case it is decided to adopt the flip, it is the *revised* block, not the original square, which is flipped. If it is decided to reject the flip, the system returns to the configuration that existed before the whole process (the process of flipping the original square and then revising it) has started. For this purpose, the block revision process is done on a separate "*work area*". Only if and when it is decided to flip, the resulting configuration in the work area actually replaces the original configuration.

A full relaxation sweep on blocks of a linear scale of b is performed this way: The entire lattice is divided into squares of $b \times b$ sites. The squares are usually, but not necessarily, chosen so as to have no or minimal overlap. These squares are scanned in some prescribed order. Each one in its turn is transferred to the center of the work area, where it is flipped, then revised, and then either adopted or rejected, based on (3.2). (The choice of the artificial β, which may depend on the revised block, is discussed in Sec. 5.2.) Such a relaxation sweep is called *revised* $b \times b$ *block* (*stochastic* or *deterministic*) *relaxation sweep*, where the adjectives "stochastic" or "deterministic" stand for finite or infinite artificial β, respectively.

3.5 Multi-level recursion

The revised block relaxation, as described so far, works very well as long as the blocks one needs to flip are not too large. Revised 2×2, and then 4×4, block relaxation sweeps are likely to locate and flip all reversed blocks of comparable sizes. This is no longer true when the flipping of larger blocks is required. At such sizes the differences between the standard squares and the reversed blocks are no longer so local, and thus cannot efficiently be realized by point-by-point optimization. They can, however, be realized by supplementing the point-by-point optimization with a block-by-block optimization. Thus, the whole process should be *recursive*: The large-scale block relaxation employs, in its revision processes, relaxation sweeps over smaller-scale blocks, which in turn may employ still smaller scales, etc.

In the next section this idea is restated from more general points of view. Further important rules and algorithmic ideas are then described in Sec. 5.

4. Principle of Multi-Level Discrete-State Optimization

The optimization problems to which multi-level processes have traditionally been applied (e.g., discretized PDE problems in variational formulation) are *continuous-state* problems; that is, each of their unknowns (states) can assume any real value, or at least (in constrained optimization problems) any real value within certain bounds. In such problems the solution can be regarded as a linear combination of components, each having a specific scale; e.g., Fourier components, each having a specific wavelength. The point-by-point minimization relaxation is efficiently used to converge the small-scale components (those with wavelengths comparable to the meshsize), while block relaxation is used to effect larger-scale convergence: each relaxation sweep over blocks of a given size is efficient in converging components of comparable scale (e.g., Fourier components with wavelength comparable to the block width). Thus, upon completing the block relaxation, the remaining errors are small-scale errors, representing fine-grid details which are invisible to block processing. Owing to their local nature, these small-scale errors can effectively be reduced by subsequent point-by-point relaxation.

By contrast, in case of discrete-state problems, where each unknown can assume only some *discrete* values (such as +1 and −1 in Ising spin problems), the solution cannot be described as a sum of components of different scales. Hence, the fine-scale details cannot be *added* to coarse-level changes; at each point they either completely *cancel* those changes, or leave them exactly as they were. More importantly, in discrete-state relaxation the question at each point or block is not *how much* change to introduce, but *whether or not* to change. In this situation the finer-scale details accompanying a coarse-level change cannot be added *later*, *after* that change has been introduced, because they may well affect the very decision whether to introduce the change at all. Hence, in discrete-state minimization (or optimization) problems, multi-level processing should be governed by the following principle.

Decide on a large-scale change only after calculating its effects at all finer scales.

Employ this principle recursively.

This principle, we believe, can successfully be applied to many discrete-state optimization problems, including combinatorial problems such as the travelling salesman, problems in discrete-state physics, in electronic chip optimization, in image restoration and pattern recognition, and others. Moreover, this principle is likely to be very useful also to various *continuous*-state minimization problems, especially those which share with the discrete-state ones the property of having many local minima.

Indeed, the above multi-level approach can readily be motivated in terms of the typical topology of local minima. The role of large-scale changes is to take us out and away from a local minimum, sufficiently away that we do not return there; take us, in other words, out of the current attraction basin. This would bring us into a new basin. Now to decide whether to prefer the new basin to the old one, it is first necessary to reach its bottom, so that it can be compared to the bottom of the old basin. The large-scale change itself is not likely to land exactly at the bottom; in fact, it is most likely to land far from it, so arbitrarily high on the new basin's walls that comparison to the old basin, where the bottom had already been reached, is completely meaningless. By calculating the finer-scale effects of the large-scale change we basically attain the new bottom, where comparison to the old bottom is meaningful. Hence the principle as stated above. Furthermore, this principle should be employed recursively, because usually the energy function is made basins within basins within basins. The different scales of changes should, very roughly, correspond to moving between the different scales of basins.

One additional philosophical comment. Since the multi-level optimization described here proves to be a very powerful algorithm, it is likely to have been used in biological evolution. This suggests that evolution itself has not been accomplished by merely accumulating many small-scale improvements. Once in a while, a larger-scale (i.e., a more profound) change must have occured which, in all probability, did not immediately produce more viable individuals; but, if these individuals were just fit enough and lucky enough to survive several generations so as to evolve the smaller-scale (less profound) features needed to support the large-scale change, then the advantages (if any) of the overall change could come to play toward its preferred selection. Once in a much longer while a still more profound change could have emerged this way, lucky to survive enough generations to see the several recursive levels of improvements needed to show its profound advantages.

5. Further Multi-Level Rules

The central algorithmic concept introduced above can be implemented in various different ways and degrees of sophistication. Our experience with diversified types of examples has led us to several further observations and rules, which have subsequently been incorporated into the algorithm, correcting its performance in important ways. The main rules of multi-levelling are described below, while various other techniques are reported in the Appendix.

5.1 Level scheduling: the basic algorithm

5.1.1 General rule. An important rule, applicable recursively at all levels of the algorithm, is the following.

LEVEL SCHEDULING RULE: Do not relax on a coarser level (i.e., with larger blocks) before your configuration has been optimized (i.e., relaxed) on finer levels.

The reason for this rule is simple. In the block relaxation as described above, any flipped block is optimized on finer levels before its energy is being compared to the energy of the old (pre-flipping) configuration. Now clearly, this comparison can be meaningful only if the old configuration has itself previously been optimized on the finer levels.

Moreover, for the same reason, this rule should also be applied in scheduling levels *within each* *revision process.* This leads to a multi-level algorithm which should basically have the following structure.

5.1.2 Recursive algorithm. Let $1 = b_0 < b_1 < \cdots < b_M$ be the block sizes; that is, at level k, the unrevised blocks are $b_k \times b_k$ (mostly disjoint) squares of spins. We have usually chosen $b_k = 2k$ for $k = 0, 1, \ldots, M-1$, and $b_M = L \leq 2M$, where $L \times L$ is the entire lattice. (In many problems, some of the coarser levels can be omitted. Also, other choices of block sizes can as easily be implemented in the present program.) Denote by RELAX(k, D) a relaxation pass on level k over a domain D, where $D = \bigcup_{j=1} JD_j$ is a union of (usually disjoint) $b_k \times b_k$ squares D_j. For $k = 0$ these blocks are single spins, and the relaxation pass is the usual point-by-point relaxation described in Secs. 3.1 and 3.2 (the scheduling of the artificial β is discussed in Sec. 5.2). For $k > 0$ we recursively define RELAX(k, D) as performing, for each $j = 1, 2, \ldots, J$, the following steps A to C.

A. Transfer D_j and a sufficiently large neighborhood of it to the work area of level k, and there flip D_j.

B. Revise D_j by performing RELAX$(\ell, D_j\ell)$ $n_{\ell,k}$ times, for $\ell = 0, 1, \ldots, k-1$, *in that order.* (Sometimes a reversed sequence is added, so that the entire order is $\ell = 0, 1, \ldots, k-2, k-1, k-2, \ldots, 1, 0$.) Each of the domains $D_j\ell$ is usually a $b_{\ell,k} \times b_{\ell,k}$ square (sometimes a $\bar{b}_{\ell,k} \times \bar{b}_{\ell,k}$ square at its center is subtracted from it), whose center coincides with the center of D_j. Parameters like $n_{\ell,k}$, $b_{\ell,k}$ and $\bar{b}_{\ell,k}$ are chosen apriori.

C. Based on δE (cf. (3.1)) of the revised block, decide whether or not to accept its flipping (using the prescription detailed in Secs. 5.2.2 and A.3). In case of acceptance, replace the original D_j and its neighborhood by the content of the work area of level k.

Note that the algorithm uses only one work area per level, the size of which is comparable to the corresponding block size. The total storage requirement is therefore at most several times the size of the problem's lattice. In many problems the coarsest levels are not needed, in which case the required storage is just a fraction more than that of the given lattice.

A complete multigrid cycle consists of performing RELAX$(k$, the entire lattice$)$ n_k times, for $k = 0, 1, \ldots, M$, *in that order.*

5.1.3 Complexity considerations. The size of the relaxation areas should carefully be checked, otherwise the total computational work may grow too fast as a function of N, the total number of sites.

To see the essential relations, denote by w_k the work *per spin* involved in RELAX(k, D).

Denote also by $a_{\ell,k}$ the number of spins in $D_j\ell$ (in Step B of RELAX(k, D)) divided by b_k2; that is, $a_{\ell,k}$ is the size of the level-ℓ relaxation domain relative to the size of the square it serves to revise. Usually $a_{\ell,k} = b_{\ell,k}2/b_k2$. Neglecting the small work of Steps A and C, one gets

$$w_k = \sum_{\ell=1} k - 1 w_\ell \; n_{\ell,k} \; a_{\ell,k} \tag{5.1}$$

This implies that w_k grows geometrically with k, and is therefore best restrained by choosing $n_{k-1,k} = 0$, i.e., by skipping level $k-1$ altogether. Indeed, *revising level-k blocks by relaxation on level $k-1$ is not really needed*, especially when stochasticity (which takes care of blocks far from square-like shapes; cf. Sec. 5.2) rectangular enlargements (cf. Sec. A.6) and adaptivity (on level $k-2$; see Sec. A.4) are used. It may in fact even be *undesired* to relax on level $k-1$, because its scale is too close to the scale of the revised block, hence the relaxation may completely destroy the block (flip it back) instead of revising it.

The growth of w_k is further restrained by choosing $n_{k-2,k}$ and $a_{k-2,k}$ as small as possible. Our usual choice has been $n_{k-2,k} = 1$ and $a_{k-2,k} = 25/16$; namely we chose $b_{k-2,k} = 5b_{k-2}$, just enough to cover a little more than $b_k = 4b_{k-2}$. Necessary exceptions to these prescriptions are $n_{0,1} = 1$, $b_{0,1} = 4$, $b_{0,2} = 6$ and $b_{\ell,M} = L$, yielding $a_{0,1} = 4$, $a_{0,2} = 9/4$ and $a_{\ell,M} = 1$.

Note, on the other hand, that the growth of w_k is not much affected if somewhat larger values of $n_{\ell,k}$ are chosen for $k - \ell > 2$. Similarly, if the entire lattice is large, the work of a complete cycle does not significantly increase if many more sweeps are made on the finest levels: e.g., if $n_0 = 20$ and $n_1 = 5$ in problems with $L \geq 50$. *It is important to use such extra fine-level sweeps when L is large*, because it is exactly for large L that very special local situations (e.g., special snake-like reversed blocks of width 1) become more probable, which requires (e.g., for flipping those blocks) a longer sequence of point-by-point stochastic relaxation sweeps.

In our current algorithm, the work is somewhat larger than indicated by (5.1), due to adaptation processes (Sec. A.4). A typical relation in practice may for example turn out to be $w_k \approx 2\sum_{\ell=1} k - 2w_\ell$. This implies $w_k \sim 2k$, hence the overall work of a cycle is $W \sim 2MN = N3/2$. With other choices of the parameters a larger W may emerge, but it need never exceed $O(N2)$.

5.2 Artificial stochasticity: how much when

An important question of course is how much stochasticity (i.e., how large artificial annealing temperature $1/\beta$) should be used at various stages of the algorithm. Since the multi-level process is itself a mechanism for escaping from local minima, it is not clear that artificial temperature is at all required (especially when the approach of Sec. A.2 below is adopted). Indeed, in some cases we have found that strict minimization ($\beta = \infty$ at all stages) multi-level algorithms (especially when equipped with the devices explained in Secs. A.3 and A.4) converge to a ground state in less than one cycle, thus achieving best performance without *any* stochasticity.

Still, at least at the present stage of algorithmic development, stochasticity was found to be very helpful in most cases. More specifically, its usefulness is typically pronounced whenever a deterministic algorithm gets stuck with a reversed block which is *long and narrow.* Such a block is not approximable, even roughly, by any square at any level. Hence a deterministic algorithm may miss it, while a stochastic relaxation, on a scale comparable to the *width* of that block, can quite easily break it into shorter blocks.

5.2.1 Annealing rule. Since the goal is to reach strict ground states, not to just wander stochastically around them, the process cannot be left at finite artificial temperatures. It should always be terminated with a strict minimization ($\beta = \infty$) pass.

This, moreover, is again recursively true at all levels. That is, each block revision process which employs a stochastic relaxation pass on any finer level, should follow it up with strict minimization on that same finer level, before switching to a coarser level. This is because, as explained in Sec. 5.1.1, there is no point in going to a coarser level with a configuration which has not first been optimized on the finer level. Hence the following rule:

ANNEALING RULE: At every stage and every level of the algorithm any stochastic relaxation should be followed up by strict minimization on the same level.

Observe that strict minimization at a certain level may of course itself employ *stochastic* passes (followed by strict minimization passes) at *finer* levels.

Also note that the annealing could be *gradual*: at any level one could start with some lower β and gradually increase it. In practice, however, the best approach is usually to employ just one finite β (see Sec. 5.2.2) before increasing it to ∞. This is in sharp contrast to one-level annealing processes, which should be done in a careful, gradual manner.

5.2.2 Size of β. In principle there is a specific value of β for each block Q. That is that value which would make it probable to flip Q whenever Q is likely to be part of a longer (but not wider) reversed block. Such a value of β should be different at different levels. Moreover, it can actually be tuned to the very block Q whose flipping is currently to be (stochastically) decided.

Thus, more precisely, the decision whether or not to flip a revised block Q is done as follows. If $\delta E(Q) \leq 0$, the flipping is always accepted. If $\delta E(Q) > 0$ and relaxation is deterministic, it is rejected. If $\delta E(Q) > 0$ but relaxation is stochastic, then the algorithm measures the likelihood of Q to be part of a longer block whose flipping would lower the energy, and randomly decides accordingly. In our present algorithm, the flipping probability $P_f(Q)$ is based on the following two easily-measured quantities:

1) The energy difference $\delta E(Q)$ (see (3.1)).

2) $B_\lambda(Q)$, the bond violation, at size λ, around the boundary of Q, where λ is comparable to the linear size of Q. $B_\lambda(Q)$ is defined as follows. Let $\langle i_\nu, j_\nu \rangle$, for $\nu = 1, \ldots, \mu$, be the μ bonds around the boundary ∂Q, ordered in their (cyclic) geometrical order. Set $b_\nu = J_{i_\nu j_\nu} S_{i_\nu} S_{j_\nu}$, where S_i is the spin value at site i *before* flipping Q. Also, because of the cyclic ordering, for $\nu > \mu$ define $b_\nu = b_{\nu-\mu}$. Then

$$B_\lambda(Q) = \max \left[0, \max_{\substack{1 \leq \nu_1 \leq \mu \\ 0 \leq \nu_2 - \nu_1 < \lambda}} \sum_{\nu=\nu_1} \nu_2 b_\nu \right] . \tag{5.2}$$

Thus, $B_\lambda(Q)$ measures how much bond violation can be cancelled by flipping, together with Q, a neighboring block coupled to Q by at most λ consecutive bonds. We have usually chosen $\lambda =\mid Q \mid 1/d$, where $\mid Q \mid$ is the number of spins in Q, and d is the problem's dimension ($d = 2$ in our present description). The amount of work in calculating $B_\lambda(Q)$ is negligible compared with the revision work that produced Q.

With those two quantities, the block flipping probability $P_f(Q)$ is defined by (3.2), with

$$\beta = \gamma / B_\lambda(Q), \tag{5.3}$$

where γ is a global control constant; $\gamma = 1$ may be a general good value (when used together with the device explained in Appendix A.1).

Note that at the finest (single spin) level, $\lambda =\mid Q \mid= 1$, and almost always $B_1(Q) = 1$, hence normally $\beta = \gamma$ and the block stochastic relaxation described here reduces to the usual point-by-point stochastic relaxation described in Sec. 3.2. The only exception is when $\delta E(Q) \geq 0$ and $B_1(Q) = 0$, that is, when all the bonds of the unflipped spin are violated but in spite of that its flipping would not lower the energy. This is possible when a strong counter field exists at that site. In many problems the field cannot be that strong, so that stochastic relaxation on the finest grid can be programmed in the traditional way ($\beta \equiv \gamma$), saving the work of calculating B_1. Nevertheless, note that in case $\delta E(Q) \geq 0$ and $B_1(Q) = 0$ does occur, rejecting the flipping (as dictated by the present scheme, but not by the usual stochastic relaxation) is indeed the correct step, because there can be no ground state where this spin is flipped.

6. Numerical Minimization Tests

6.1 Test classes

The minimization algorithm defined above, with its supplementary techniques described in the Appendix, was trained and tested on the following five classes of two dimensional Ising spin problems.

(1) The uniform ($J_{ij} \equiv 1$) and homogeneous ($h_i \equiv 0$) model, with periodic boundary conditions on various sizes of square lattices. The two ground states are known ($S_i \equiv 1$ and $S_i \equiv -1$), and the purpose of testing is to see how fast these states are obtained from various initial configurations.

(2) The same, except that the field h_i is non-constant, but still constructed so that the ground states are known. For example, $h_i = H_1 > 0$ at all sites i inside some convex domain and $h_i = H_2 < 0$ outside that domain, yielding three possible ground states ($S_i \equiv 1$, $S_i \equiv -1$ and $S_i \equiv$ sign h_i), depending on the values of H_1 and H_2. Of particular interest are of course those special values that give two, or even all three, ground states; an efficient solver should then easily move back and forth between those states. More complicated geometries were also tried.

(3) The same, except that each h_i is a real random number, uniformly distributed in the interval $(-H, H)$. In this case the ground states are not apriori known, but we can compare the minima reached from different initial configurations, by different algorithms. In most experiments we have taken $H = 2$, which is small enough to produce long-range interactions, but not as small as to make them trivial: typically, the ground state has a sea of spins of one sign, with large randomly shaped islands of the opposite sign.

(4) $J_{ij} \equiv 1$, h_i is randomly either $-H$ or $+H$, with equal probabilities, and the boundary conditions are free (no periodicity). For one particular distribution of signs on a 50×50 lattice, and for four different values of H (73/26, 73/27, 73/30 and 73/32), exact ground states, calculated by a special graph-theoretic method (cf. Sec. 3.1), were supplied to us by the Grenoble group [3].

(5) Spin-glass models: $h_i \equiv 0$, J_{ij} is randomly $+1$ or -1, with probabilities p and $1-p$, respectively, and the boundary conditions are periodic. For three cases ($p = .12$, $p = .146$ and $p = .5$) on a 20×20 lattice, exact ground states are described in [6]. The difficult case here is $p = .146$, nearly the critical value, at which large blocks of aligned spins tend to form. For $p = .12$ and $p = .5$ the correlations are short range. Moreover, for $p = .5$ there are many ground states.

Note that in all cases $|J_{ij}| \equiv 1$. The present program is not developed for cases of *strong* local variations in $|J_{ij}|$ (see Sec. A.3), but in principle could handle general couplings.

6.2 Comparison to simple annealing

For each of these classes we have compared multi-level solutions with a single-level solution by simulated annealing. The latter turned out as effective as the former for Class 4 problems, at the above-cited values of H. These problems are indeed dominated by difficulties at the finest level. Namely, the reversed blocks that tend to form are thin, mostly one spin wide. Only for smaller H (e.g., $H \approx 1$) wider blocks would become likely, making multi-levelling necessary. But, even in multi-level processing, thin reversed blocks in isotropic problems should be flipped by relaxation at the single-spin level. In fact, these Class 4 problems – especially those with the lower H, which produce long snake-like reversed blocks (length 4 for $H = 73/30$ and length 7 for $H = 73/32$) – served as important test beds for our single level processing, leading to several of the techniques described in the Appendix.

For Class 1 problems, simple annealing still performed reasonably well. The reason is that reversed blocks are in a sense still local: even if a wide reversed block is formed (e.g., an extensive island of -1 spins in a sea of $+1$ spins), it is not necessary to flip this whole block to see a decrease in energy. It decreases each time one of the end rows or columns is flipped. Moreover, if an annealing process starts with a sufficiently small β – or equivalently: if one starts with a random first approximation – the chance is that the reversed blocks are not very wide, hence the rows to be flipped, one at a time, are not very long. Unlike the previous class, however, multi-levelling did accelerate convergence of Class 1 problems, typically reducing simple annealing solution times by one order of magnitude for moderate-size grids (e.g., 32×32).

The real strength of multi-levelling is shown in various problems of Classes 2, 3 and 5. Here, in many cases, simple annealing fails even to *approach* the ground energy, no matter how slowly β grows, how many iterations are made, or what supplementary techniques are tried. It is doomed to fail whenever local convergence contradicts global convergence; e.g., whenever there exist some wide subdomains where the magnetic field is mostly in a direction opposing the ground-state spins.

For example, taking a Class 2 problem with a sufficiently wide convex domain and values of H_1 and H_2 not far from the values that give two ground states (e.g., a 5×5 square with $H_1 \approx .8$ and $H_2 = -.1$), annealing could reach only the configuration $S_i \equiv$ sign h_i, which (e.g., for $H_1 < .8$) is not necessarily the true ground state. For several Class 3 problems (with $H = 2$) annealing never even approached the lowest energy produced by the multi-level algorithm, no matter how gradually the artificial temperature was decreased. In Class 5, annealing did reach ground states in the easier cases, but failed for $p = .146$.

6.3 Multi-level performance: current status

The current set of multi-level minimization programs is not fully streamlined. It is still a patchwork. Some of the supplementary techniques (see Appendix) are not yet fully implemented: some of them were introduced after most of the experiments were done, some are programmed only for the finest level, others only for coarser ones. (For necessary technical reasons, the finest level is treated by different routines, unlike the traditional multigrid practice.) Also, the programs are still far from being optimized with respect to CPU time.

Instead of timing, we have measured performance by counting *point decisions (PDs)*. One PD is consumed whenever a decision (to flip or not) is made at the single-spin level. This involves calculating (2.4) and, for a probabilistic decision, also (2.3). The main work at coarser levels is the block revision, which ultimately always leads to finest-level sweeps, hence can also be measured in PDs.

For any one of the test problems the algorithm produced the minimal energy in at most few cycles, always costing less than $3N2$ PDs. In some cases the algorithm jumps several times within one cycle back and forth between several approximate ground states. For example, in Class 2 problems with a 5×5 convex domain with $H_1 = .8$ and $H_2 = -.1$, the ground state $S_i \equiv$ sign h_i was produced in the first relaxation sweep over 4×4 blocks, the other ground state ($S_i \equiv -1$) was then produced in relaxing over 8×8 blocks, and two additional jumps between the two states occurred still within that 8×8 block relaxation (due to the adaptivity feature). A proper use of LCC (see § A.3) would in this case determine the existence of two equivalent ground states, or would choose the lower of them in case H_1 is slightly different from the transition value .8. Similarly, in one of Class 3 examples, there were two widely different configurations with almost the same energy, one of them minimal. Approximate transition between them always occurs at the coarsest scale (revising the entire grid flip), with exact transition seen upon using the suitable

LCC.

Many more tests are of course needed to establish the exact efficiency of the multi-level algorithms, especially since the present tests have not been entirely “fair”: they have been performed with the same classes of problems used in the training of the algorithm.

6.4 Summary. One-cycle algorithms

The present minimization algorithm is not “perfect”. To any one of its versions it may be possible to construct “counter examples” which would require exponential solution time, and this may remain true for future versions as well. But the more developed is the algorithm, the less likely are the counter examples. At the present state, the probability of such examples seem already to be small enough so that average solution times is $O(N2)$ or better. To maintain this efficiency in the future, with new *types* of problems, we may of course find, as we have found in the past whenever new types appeared, that some additional rules should be understood and implemented.

What is important to realize is that, for most purposes, it is not the ground states that are required. All that is needed are *approximate* ground states, which approximate ground state statistical properties. We have observed that the present algorithm easily yields such approximations in just one cycle.

Indeed, whenever the configuration obtained by one multi-level cycle is not itself a ground state, the difference turns out to be insignificant: the energy is very close to the true minimum, and, more importantly, slight changes of data (e.g., very small changes in the random magnetic field) can turn the obtained configuration into a ground state, hence its *apriori* chance of being itself a ground state is likely to have been about the same as that of the current ground states themselves.

We thus conjecture that *for many statistical purposes, one cycle of the multi-level minimization algorithm is enough for each set of data.* We further conjecture that even a relatively “light” cycle will often do, lighter even than the $O(N3/2)$ cycle mentioned in Sec. 5.1.3. Moreover, in each additional cycle many more approximate ground states may be encountered, which may similarly serve in calculating the desired statistics. We plan to test these conjectures on some classes of problems.

One should of course be careful in using this approach. It cannot be used when the desired statistics are strongly affected exactly by those special rare reversed blocks which the algorithm takes longer to flip.

7. Finite Temperature: Preliminary Observations

7.1 Continuous-state problems

For zero temperature (minimization problems), it has been shown above that discrete-state multi-level processing is considerably more involved than continuous-state multi-level processing. The same is expected at finite temperatures, as long as they are not as high as to have correlation lengths comparable to the meshsize. The first step in developing multi-level processes at positive temperatures is therefore made in the context of continuous-state problems, such as the XY model or the Heisenberg model. Multi-level Monte-Carlo processes can then be developed along lines similar to familiar multigrid techniques. In particular, since the problems are nonlinear, the inter-grid transfers are made in the *Full Approximation Scheme* (*FAS*; cf. [10, §8]), whose conventions will be used below.

For simplicity, assume first that only two levels are involved. The coarse level is typically made of every other column and every other row of the fine level. Two basic transfer operators should be defined between these two levels: the coarse-to-fine interpolation I_cf, and the fine-to-coarse local averaging I_fc. The interpolation can be simple bilinear (in terms of angles, in case of Heisenberg or XY models, so as to preserve the unit size of the spins), and the local averaging can be its transpose.

We assume that for every fine-grid configuration uf, an energy function $Ef(uf)$ is defined. This induces, for every coarse-grid configuration uc, the energy function $Ec(uc) = Ef(I_cfuc)$. Since this coarse-grid energy function will be defect-corrected below, we can actually replace it by any convenient function $\bar{E}c(uc)$, as long as $\bar{E}c(uc)$ approximates $Ec(uc)$ for smooth uc; i.e.,

$$\| \nabla c\bar{E}c(ei\omega x) - \nabla cEc(ei\omega x) \| \,/\, \| \nabla cEc(ei\omega x) \| \to 0 \quad \text{as} \quad |\omega| \to 0 \tag{7.1}$$

where $x = (x_1, \ldots, x_d)$ are the space coordinates, $\omega = (\omega_1, \ldots, \omega_d)$, $\omega x = \omega_1 x_1 + \cdots + \omega_d x_d$, $\nabla c = (\partial_1 c, \partial_2 c, \ldots)$, $\partial_i cEc(uc) = (\partial / \partial u_i c)Ec(uc)$, and $\| \cdot \|$, $| \cdot |$ are any finite norms. This normally allows using $\bar{E}c$ which has the same functional form as Ef; e.g., nearest neighbor couplings only.

For any given fine-grid configuration $\tilde{u}f$ we then define the corrected interpolation $\tilde{I}_cf$ (the FAS interpolation), given by

$$\tilde{I}_cfuc = \tilde{u}f + I_cf(uc - I_fc\tilde{u}f) \tag{7.2}$$

and the corrected coarse-grid energy

$$\tilde{E}c(uc) = \bar{E}c(uc) + (uc, \tilde{\tau}_f c) \tag{7.3}$$

where $(,)$ is the inner product and

$$\tilde{\tau}_f c = (I_c f)T \nabla f E f(\tilde{u}f) - \nabla c \bar{E}c(I_f c \tilde{u}f), \tag{7.4}$$

$(I_c f)T$ denoting the transpose of $I_c f$. These corrected forms interpret any coarse-grid configuration uc as describing only the change from $I_f c \tilde{u}f$, with $\tilde{u}f$ (the current fine-grid configuration) still describing its fine-grid details.

A typical cycle of the multi-level algorithm starts with several sweeps of a usual point-by-point Monte-Carlo process (see Sec. 2.2) on the fine grid, bringing the system to a *local* statistical equilibrium. The resulting configuration $\tilde{u}f$ is then used in a coarse-grid Monte-Carlo, based on $\tilde{E}c$ and on the starting configuration $I_f c \tilde{u}f$. This coarse-grid Monte-Carlo can (and for full efficiency usually should) itself use still coarser grids, in a similar manner. The cycle terminates with the final coarse-grid configuration uc being used to update the fine grid, replacing $\tilde{u}f$ by $\tilde{I}_c f uc$.

Note that in each cycle, throughout the coarse-grid processing, $\tilde{\tau}_f c$ is fixed, representing a fixed field-like fine-to-coarse defect correction. This allows the coarse-grid Monte-Carlo to be done without constantly using the fine grid, hence to consume relatively short CPU times. The main work per cycle is the few fine-grid sweeps. Their number is small since they need to equilibrate only on the smallest scale (the scale invisible to the coarse grid).

When enough grids, to the coarsest possible scale, are recursively used in such a manner, this algorithm has fast transition times. In one cycle it almost equilibrates at all scales, hence also almost decorrelates at all scales. Hence few cycles could be enough for calculating statistical averages, provided the slow balancing of deviations (cf. Sec. 2.3) is also treated at all levels. This is indeed possible, using suitable inter-grid transfers.

As a simple example, consider the calculation of $\langle uf \rangle$, the average of uf over the domain and over all configurations, weighted by their physical probabilities. If the fine-to-coarse transfer $I_f c$ is sum-preserving (or "full weighting", in the usual multigrid terminology – cf. [10, §4.4]), then $\langle I_f c uf \rangle = \langle uf \rangle$. To average out local deviations, $I_f c uf$ should be a local sum-preserving averaging not of uf itself, but of $\bar{u}f$, where $\bar{u}_i f$ is defined at each site i as the expected value of $u_i f$ given its neighborhood. (Using suitable pre-calculated tables, somewhat larger neighborhoods can quickly be taken into account – making the averaging out of local deviations even better.) Employing similar fine-to-coarse transfers at all levels, the configuration uM on the coarsest level will represent averaging of uf over all scales except for the coarsest. Hence $\langle uM \rangle$, calculated for example by averaging $\bar{u}M$ over a sequence of coarsest-grid Monte-Carlo steps over several multigrid cycles, will give us a good approximation to $\langle uf \rangle$.

An important feature of this algorithm is that its statistics are as accurate at large scales as at small ones. Hence, even rather crude approximations have reliable large-scale figures. Moreover, and for a similar reason, one need not calculate $\langle uf \rangle$ to a high precision in order, for example, to be able to calculate its derivative ∂ with respect to the temperature; $\langle \partial uf \rangle$ can itself be calculated, along with $\langle uf \rangle$, and to a comparable accuracy, by transfering it from finer levels to coarser ones in an analogous manner. The local computation of $\overline{\partial u}f$ is as easy as that of $\bar{u}_i f$, but, unlike the latter, it gives meaningful approximations to $\langle \partial uf \rangle$ only when repeated at all scales.

At scales where large local deviations are statistically important, if there are such scales, the algorithm should be modified (or not allowed to reach such scales), since the coarse-level moves are no longer independent of their statistical effects at the fine level. To see how such effects can be taken into account, consider the more extreme case of nonlinearity discussed in the next section.

7.2 Discrete-state problems

Returning to Ising spins as a model for discrete-state problems, it is clear from our zero temperature investigations (Sec. 3), that coarse-grid moves cannot correctly be decided without calculating the details of their fine-grid effects, in a certain recursive manner (see Sec. 5.1). The general outline for such an algorithm (not yet implemented) is as follows.

A multi-level cycle should start on the finest (single spin) level, and move to increasingly coarser levels. At each level Monte-Carlo sweeps over the entire domain should be carried out. Each step of a sweep at scale b level should consist of flipping a $b \times b$ block, then revising it by finer level Monte-Carlo sweeps in its vicinity, then deciding whether to accept or reject the revised block, employing for example the Metropolis rule. This rule should use the *physical* β, of course. When this β is large (substantially larger than critical), though, it should be useful to prefix such "physical sweeps" with some "artificial sweeps", using lower artificial β. Enough physical sweeps should in principle be made at each level (also within the revision processes) to reach local equilibrium, but a reasonable approximation to such an equilibrium should in practice be obtained in just a couple of sweeps.

Two important requirements that should (approximately) be satisfied by the revision processes seem uncertain: (a) Maintaining detailed statistical balance, i.e., ensuring the stationarity of the physical distribution (2.2). (b) Balancing statistical deviations: the revision may, especially at critical and smaller β, create deviations unrelated to (e.g., not opposing) those in the original configuration, hence weighting together the revised and the original configurations may not be fully efficient at averaging deviations out.

The two requirements can concurrently be met by the following three devices:

(A) The same revising process which is applied to the configuration with the flipped $b \times b$ square is also applied to the original (pre-flip) configuration. The two processes are correspondingly called the *flipped revision* and the *unflipped revision.*

(B) All revising processes are designed so as to preserve a certain property, such as sum total (average magnetization) or certain majorities (e.g., for $I_f c$ defined by a majority rule). Only moves which preserve that property should be considered by the revising Monte-Carlo. In addition, the revising processes are of course *local*, hence they preserve the configuration outside (a certain neighborhood of) the flipped square. Thus each revision is done within a certain set of permissible configurations. For the unflipped revision, that set is denoted by R; for the flipped revision – R'. Starting with any member of such a set, all other members of the same set should be accessible by the revising Monte-Carlo.

(C) The end products of the two revising processes should be weighted against each other (either for the purpose of choosing one of them to continue the process, or for calculating suitably weighted averages) *not* according to their relative probabilities (2.3), but according to the relative probabilities of the corresponding *sets*, R and R'. The relative probability of R (R') can be computed during the unflipped (flipped) revising process as the inverse of the average of $e\beta E(c)$, calculated along with other averages of interest. The averages are calculated using single-spin averaging of deviations (see Sec. 2.3), and also (for $b > 4$) similar *block* averaging – the whole process being recursive, like that described in Sec. 5.1.

These three devices need not always fully be implemented. Device (A) may not be necessary when the level scheduling rule (Sec. 5.1.1) is adopted, ensuring that the original configuration is statistically the same as the one produced by the unflipped revision. Device (B) may perhaps be skipped when a crude approximation is desired and only very few sweeps are therefore performed at each level, approximately preserving the desired property anyway. Device (C) may be avoided by (crudely) taking the relative probabilities of the end configurations as representing those of the corresponding sets. As noted before, owing to the multi-levelling, crude approximations may already yield reliable long-range statistics.

Appendix: Supplementary Minimization Techniques

The development of the multi-level energy minimization algorithm for Ising spin systems has led to the introduction of several new techniques, reported below. All these techniques could in principle be developed also for single-level minimization algorithms, but the multi-level framework has been important in their origination, for two related reasons: First, once multi-level processing has overcome the most fundamental sources of algorithmic slowness, less profound sources come into clear view and must be dealt with. (The history of this work is thus itself an illustration of multi-level evolution, as pictured in Sec. 4.) Secondly, in the multi-level context the task of relaxation at each level is more sharply focused (to the localization and flipping of only those reversed blocks whose width is comparable to that level's scale), hence easier to study and devise.

A.1 Lower starting β

The above value of β (5.3) has been chosen so as to make it probable to flip a block Q (correspondingly: a single spin) whenever it sits *at one end* of a reversed longer block. Sometimes, especially at the finest levels, reversed blocks can (usually in low probability) appear which are considerably long (compared with their width, which itself is comparable to the meshsize of the level). On these finest levels, several (even many) sweeps are usually performed (cf. Secs. 5.1.3 and A.4), and it is desired to make it highly probable that the first sweep will break any long reversed block, whose width is comparable to the meshsize, into shorter ones. It is desired, in other words, to make it probable to flip a block Q even when it sits *in the middle* of a reversed longer block. For this purpose, this first sweep should employ, instead of (5.3), a lower value of β, given by

$$\tilde{\beta} = \gamma / [B_\lambda(Q) + \tilde{B}_\lambda(Q)] \tag{A.1}$$

where, similarly to (5.2)

$$\tilde{B}_\lambda(Q) = \max\left[0, \max_{\substack{1 \le \nu_1 \le \mu \\ 0 \le \nu_2 - \nu_1 < \lambda}} \sum {\nu_2}_{\nu=\nu_1} \tilde{b}_\nu \right]. \tag{A.2}$$

Here $\tilde{b}_\nu = b_\nu$ except that $\tilde{b}_\nu = 0$ for $\bar{\nu}_1 \le \nu \le \bar{\nu}_2$, where $\bar{\nu}_1$ and $\bar{\nu}_2$ are the values of ν_1 and ν_2 for which the max in (5.2) has been attained.

Observe that on the finest (the single spin) level, where $\lambda = 1$, the above prescriptions reduce to choosing at site i

$$\beta = \gamma / \min(1, Bi), \qquad \tilde{\beta} = \gamma / \min(2, Bi) \tag{A.3}$$

where Bi is the number of bonds $< i,j >$ violated upon flipping S_i.

A.2 Deterministic trials

Another possible strategy is to replace the above stochastic relaxation with deterministic trials. That is, whenever $P_f(Q)$ is not small, *tentatively* flip Q and then flip and revise its neighboring block (neighboring on that part of the boundary at which the bond violation $B_\lambda(Q)$ has appeared). The two flipped blocks together are now forming one longer block $Q\prime$. In case $\delta E(Q\prime) \leq 0$, then accept this combined flipping. If $\delta E(Q\prime) > 0$ and $\delta E(Q\prime)/B_\lambda(Q\prime)$ is not small (still using the original $\lambda = |Q|1/d$), then reject the whole flipping, including the original tentative flipping of Q. If $\delta E(Q\prime)/B_\lambda(Q\prime)$ is positive but small, then continue the trial, now adding a third block. And so on. The criterion for smallness of $\delta E/B_\lambda$ can be made stricter (smaller) at each additional step. This strategy has not been tested yet.

A.3 Lowest Common Configuration (LCC)

The revised block Q is not always connected. Whenever it is disconnected, an important modification to the above algorithm is to make a separate flipping decision for each of its disconnected components. In other words, instead of choosing between the two configurations (the flipped Q vs. the original configuration), their "lowest common configuration" should be constructed.

Generally, given two spin configurations $\{S_i1\}$ and $\{S_i2\}$, their *lowest common configuration (LCC)* $\{S_i\}$ is defined as follows. Let Q be the set of sites i for which $S_i1 \neq S_i2$. And let $Q = \bigcup J_{j=1}Q_j$ be the decomposition of Q into its disconnected components; i.e., the sites of each Q_j are connected to each other by a chain of sites belonging to Q, but are not so connected to the sites of any other $Q_{j\prime}$. A *chain of sites* is defined as a sequence of sites where each pair of subsequent sites α and β are *strongly coupled*, i.e., $J_{\alpha\beta} \neq 0$ and $|J_{\alpha\beta}|$ is not much smaller than $\min[\max_\gamma |J_{\alpha\gamma}|,\ \max_\gamma |J_{\gamma\beta}|]$. (In current programs, α and β are considered strongly coupled if and only if they are nearest neighbors. The programs are thus inapplicable to cases of strong anisotropy or other large local variations in $|J_{\alpha\beta}|$. The sign of $J_{\alpha\beta}$ may however change arbitrarily.) Now, with this decomposition, for $i \notin Q$ define $S_i = S_i1 = S_i2$, while for each $1 \leq j \leq J$ define $S_i = S_i\nu(j)$ for all $i \in Q_j$, where $\nu(j)$ is either 1 or 2, whichever yields the lower energy.

Given Q, it is easy to calculate the LCC in $O(|Q|)$ operations, where $|Q|$ is the number of sites in Q. In our algorithms this work will be negligible.

A *stochastic lowest common configuration (SLCC)* of two configurations can similarly be calculated, the only difference being that the $\nu(j)$ are chosen stochastically, as prescribed in Sec. 5.2.2, using $\delta E(Q_j)$ and $B_\lambda(Q_j)$, with $\lambda =\mid Q_j \mid 1/d$.

The use of LCCs in other parts of the algorithm is sometimes also very important. In particular it is important when, typically on a large grid, the minimization processes at some particular regions of the grid are weakly coupled to each other. (For example, a block whose flipping changes the energy very little is weakly coupled to similar blocks far from it). In such a situation, each cycle is likely to attain the correct minimum at many of these regions, but the chances of getting all these minima simultaneously (hence the global minimum) may be slim, hence requiring many cycles. If, on the other hand, at the end of each cycle the attained configuration is replaced by its LCC with the configuration at the end of the previous cycle, each of those regions, once minimized, will remain so, and the global minimum will soon be reached.

The LCC can similarly further be used *within the revision processes*, especially for large blocks. Generally, *any* subprocess can be strengthened by accompanying it with a procedure of keeping in a special memory the configuration with lowest energy attained by it so far, replacing that configuration by its LCC with a newly encountered configuration whenever the latter shows promisingly low energy. At the end of the subprocess, its end product should be replaced by the LCC of this end product and the configuration in that special memory, before being used by an outer process.

Generalization. When a general number of configurations $\{S_i 1\}, \ldots, \{S_i m\}$ is given, their simultaneous LCC is defined as follows. Let Q be the set of all sites i such that $S_i k \neq S_i \ell$ for at least one pair (k, ℓ). Let as before $Q = \cup_{j=1} J Q_j$ be the decomposition of Q into its disconnected components. Then

$$\{S_i\} = \text{ LCC}(\{S_i 1\}, \ldots, \{S_i \ell\})$$

is defined by $S_i = S_i 1 = \cdots = S_i \ell$ for $i \notin Q$, while for each $1 \leq j \leq J$ define $S_i = \mu(j) S_i \nu(j)$ for all $i \in Q_j$, where the values of $\mu(j) = \pm 1$ and $1 \leq \nu(j) \leq \ell$ are chosen so as to yield the lowest possible energy. This generalization can be very useful (see end of Sec. A.4).

A.3.1 Minimized Common Configuration (MCC). A further generalization,

$$\{S_i\} = \text{ MCC}(\{S_i 1\}, \ldots, \{S_i \ell\})$$

is similarly defined by $S_i = S_i 1 = \cdots = S_i \ell$ for $i \notin Q$, whereas, unlike in LCC, the values of S_i in each Q_j are determined by a separate minimization process, trying to reach the minimal Q_j

configuration with all spins outside Q_j held fixed. This separate minimization can use very many sweeps, since Q_j would normally be small; it can use decimation procedures, in case Q_j is long but mostly one spin wide; Or it can itself use multi-level procedures, when Q_j is wider.

The generalized techniques can again be used within the revision processes. They should normally use only configurations obtained at a large β stage, lest they contain too much thermal noise.

A.4 Adaptive relaxation

The basic algorithm described above (Sec. 5.1.2) assumes pre-assigned relaxation domains $D_j\ell$ and fixed numbers $n_{\ell,k}$ of sweeps at level ℓ. Efficiency can be enhanced by adapting these parameter to the local situation. In other words, following the $n_{\ell,k}$ *fixed sweeps* (over the fixed domain $D_j\ell$), further *adaptive sweeps* can be made, each of which scans only the nearest neighbors of the sites (or blocks, if $\ell > 0$) which were flipped in the former sweep. In fact, for each block Q that was formerly flipped, the adaptive sweeps scan only its *violated neighbors*, i.e., those nearest neighbors which show large bonds violation (comparable to $B_\lambda(Q)$) with the flipped Q.

When the adaptive sweeps are stochastic, they should be followed up by deterministic adaptive sweeps (see Sec. 5.2.1). Each of the latter should then scan, in addition to the former-sweep's violated neighbors, also all those blocks whose last flipping was probabilistic (energy increasing).

Such adaptive schemes serve several purposes. First, they save scanning sub-domains of $D_j\ell$ where no further improvement occurs. More importantly, they allow the relaxation region to expand, when necessary, beyond the originally allotted $D_j\ell$. They thus enable using $D_j\ell$ of minimal size (see Sec. 5.1.3). Similarly, they permit the use of small $n_{\ell,k}$, with more passes performed only when and where needed.

A general rood procedure, in relaxing either the entire domain or within any revision process, is to first make one fixed sweep (employing usually the lower starting β; cf. Sec. A.1), then stochastic adaptive sweeps (with the usual β), and then deterministic adaptive sweeps, continuing each of these two adaptive successions until it becomes empty. On those levels where more sweeps are allowed (and desired – cf. Sec. 5.1.3), this whole sequence can be repeated several times, the end product of each such sequence being replaced by its LCC with the previous end product. Or, better still, several such end products at a time can be accumulated and then replaced by their *simultaneous* LCC or MCC.

A.5 Position shifts

Certain troublesome reversed blocks stand a better chance of being flipped when the unrevised blocks of a certain level are suitably positioned. Hence, whenever returning to relaxing the entire lattice on a given level, it is advisable to shift the positioning of that level's squares. In our power-two scheme ($b_k = 2k$), most important are half-size shifts (shifting level-k blocks by $b_k/2$ in either or both directions). Similarly, $D_j\ell$ can be shifted (e.g., replacing $b_{\ell,n}$ by $b_{\ell,n} - b_\ell$) in between relaxation passes.

In the present code only simpler shifts are implemented, produced by shifting the entire problem at the beginning of each new cycle, alternating between one-position-upward and one-position-leftward shifts.

A.6 Rectangular enlargements of flipped squares

Another device that was found useful is to start the revision process of a flipped square with rectangular enlargements. This means to add, one by one, rows and/or columns to the flipped region as long as this reduces the energy of the entire configuration. Having reached in this way the optimal rectangular flipping, the regular revision process then starts to work on it.

No detailed experiments were conducted to determine the effect of this device. It seems useful especially when skipping level $k-1$ in revising level k (see Sec. 5.1.3). Similar enlargements make also sense on the finest (single spin) level, and might in fact be very useful there (similar to the tentative trials; cf. Sec. A.2), but are not yet implemented.

Acknowledgement

We are deeply indebted to the Grenoble group of R. Rammal and J.C. Angles d'Auriac for providing us with invaluable detailed information on exact ground states of two dimensional spin-glasses and Ising models with random fields, before during and after publication. One of us DJA has also benefitted from discussions with G. Parisi, who has brought to his attention previous work on improving high temperature statistics.

References

[1] D. Andelman, H. Orland and L.C.R. Wijewardhana: Lower critical dimension of the random-field Ising model: a montecarlo study. *Phys. Rev. Lett.* **52** (1984), 145.

[2] J.C. Angles D'Auriac and R. Maynard: On the random antiphase state in the $\pm J$ spin glass model in two dimensions. *Solid State Commun.* **49** (1984), 785.

[3] J.C. Angles D'Auriac, M. Preissmann and R. Rammal: The random field Ising model: algorithmic complexity and phase transition. *J. Physique Lett.* **46** (1985), 173.

[4] J.C. Angles D'Auriac and R. Rammal: Phase Diagram of the Random Field Ising Model. CNR Grenoble, preprint, 1986.

[5] F. Barahona: On the computational complexity of Ising spin glass models. *J. Phys.* **A15** (1982), 3241.

[6] F. Barahona, R. Maynard, R. Rammal and J.P. Uhry: Morphology of ground states of the two-dimensional frustration model. *J. Phys.* **A15** (1982), 673.

[7] G. Bhanot, M. Creutz and H. Neuberger: Microcanonical simulation of Ising systems. *Nucl. Phys. FS* **B235** (1984), 417.

[8] I. Bieche, R. Maynard, R. Rammal and J.P. Uhry: On the ground states of the frustration model of a spin glass by a matching method in graph theory. *J. Phys.* **A13** (1980), 2553.

[9] E. Bonomi and J-L Lutton: The N-city travelling salesman problem: statistical mechanics and the metropolis algorithm. *SIAM Review* **26** (1984), 551. See also [16], [17] and [18].

[10] A. Brandt: Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics. Monograph, 191 pages. Available as GMD Studien 85 from GMD-AIW, Postfach 1240, D-5205, St. Augustin 1, W. Germany.

[11] M. Falcioni, E. Marinari, M.L. Paciello, G. Parisi, B. Taglienti and Y.S. Zhang: Large distance correlation functions for an $SU(2)$ lattice gauge theory. *Nucl. Phys. FS* **B215** (1983), 265.

[12] K.H. Fisher: Spin glasses I. *Phys. Stat. Solidi* **B116** (1983), 357.

[13] K.H. Fisher: Spin glasses II. *Phys. Stat. Solidi* **B130** (1985), 13.

[14] R. Graham and F. Haake: Quantum Statistics in Optics and Solid-State Physics. Springer Tracts in Modern Physics, Vol. 66, Springer-Verlag, Berlin, 1973.

[15] A. Hasenfratz and P. Hasenfratz: Lattice Gauge Theories. Florida State University, preprint

FSU-SCRI 85-2, 1985.

[16] S. Kirkpatrick: Models of disordered systems. Lecture Notes in Physics 149 (C. Castellani et al., eds.), Springer-Verlag, Berlin, 280.

[17] S. Kirkpatrick, C.D. Gelatt Jr. and M.P. Vecchi: Optimization by simulated annealing. *Science* **220** (1983), 671.

[18] S. Kirkpatrick and G. Toulouse: Configuration space analysis of travelling salesman problems. *J. Physique* **46** (1985), 1277.

[19] S.K. Ma: Renormalization group by Monte-Carlo methods. *Phys. Rev. Lett.* **37** (1976), 461.

[20] N.D. Mackenzie and A.P. Young: Lack of ergodicity in the infinite range Ising spin-glass. *Phys. Rev. Lett.* **49** (1982), 301.

[21] N.D. Mackenzie and A.P. Young: Statics and dynamics of the infinite range Ising spin glass model. *J. Phys.* **C16** (1983), 5321.

[22] E. Marinari, G. Parisi and C. Rebbi: Computer estimates of meson masses in $SU(2)$ lattice gauge theory. *Phys. Rev. Lett.* **47** (1981), 1795. For a recent review see [15].

[23] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller: Equation of state calculations by fast computing machines. *J. Chem. Phys.* **21** (1953), 1087.

[24] A.T. Ogielski and I. Morgenstern: Critical behavior of three-dimensional Ising model of spin glass. *J. Appl. Phys.* **57** (1985), 3382.

[25] G. Parisi, R. Petronzio and F. Rapuano: A measurement of the string tension near the continuum limit. *Phys. Lett.* **128B** (1983), 418.

[26] G. Parisi and Y.S. Wu: Perturbation theory without gauge fixing. *Scientia Sinica* **24** (1981), 483.

[27] H. Sompolinsky: Time dependent order parameters of spin-glasses. *Phys. Rev. Lett.* **47** (1981), 935.

[28] H. Sompolinsky and A. Zippelius: Relaxational dynamics of the Edwards-Anderson model and the mean-field theory of spin-glasses. *Phys. Rev.* **B25** (1982), 6860.

[29] R.L. Stratonovich: Topics in the Theory of Random Noise, Vol. 1, Gordon and Breach, New York, 1963.

[30] R.H. Swendsen: Monte-Carlo renormalization group. *Phys. Rev. Lett.* **42** (1979), 859.