Multilevel Computations: Review and Recent Developments[1]

Achi Brandt[2]

The Elaine and Bram Goldsmith Professor of Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel

Abstract

Multigrid and other multilevel algorithms, which combine local processing on different scales with inter-scale transfers of residuals and corrections, can solve most large-scale problems in CN operations, where N is the number of real unknowns and $10 \leq C < 10^3$ (depending on equation complexity). This includes general nonlinear discretized partial differential systems, discretized integro-differential equations (with asymptotically smooth kernels), and any geometrically-based optimization, constrained optimization, optimal control, mathematical programming, and system identification problems. In many cases the very value of C can *rigorously* be predicted by local mode analyses. It can rigorously be shown to be independent of boundary shapes and boundary conditions.

For highly nonlinear problems, and in particular for optimization problems with some discrete unknowns (such as Ising spins, yes-no decisions in econometric planning, appearance of edges in image reconstruction, combinatorial variables as in the traveling salesman problem, large-scale changes in protein folding, etc.), multilevel annealing is essential for rapidly escaping local minima with large attraction basins.

The parallel complexity of these solvers is polynomial in $\log N$. For essentially the same work they can incorporate continuation processes, local grid adaptation (including *local* coordinate transformation), etc. Re-solving a problem with small data changes would usually require much shorter processing still, allowing on-line

---

[1] The main body of this article is reproduced from [32].

[2] Research supported mainly by the Air-Force Office of Scientific Research, United States Air Force under Grants AFOSR-84-0070 and AFOSR-86-0127, and also by the United States Army Contract DAJA 45-84-C-0036.

design of complicated structures.

In many cases the multilevel interactions provide much better *discretization* schemes, because the most suitable scheme may be scale dependent. This includes nonelliptic steady state problems, highly indefinite problems with highly oscillatory solutions, problems with non-local boundary conditions (radiation conditions, flow exits, etc.), and various ill-posed problems.

Purely parabolic problems, for time $0 < t < T$, are solved in computational work equivalent to $O(\log T)$ explicit time steps.

For problems in statistical mechanics and field theory, multilevel Monte-Carlo techniques can simultaneously eliminate several kinds of slowness (critical slowing, domain vastness, slow balancing of deviations), and very inexpensively incorporate dynamical fermions. The obtained renormalization employs only simple interactions at all levels.

Content

1. Introduction

Most massive computational tasks facing us today have one feature in common: They are mainly governed by *local* relations in some low (e.g. 2 or 3) dimensional space or grid. Such are all differential problems, including flows, electromagnetism, magnetohydrodynamics, quantum mechanics, structural mechanics, tectonics, tribology, general relativity, etc., etc., as well as statistical or partly differential partly statistical problems (e.g. in statistical mechanics, field theory, turbulence), and many non-differential problems like those in geodesy, multivariate interpolation, image reconstruction, pattern recognition, many design, optimization and mathematical programming problems (e.g. traveling salesman, VLSI design, linear programming transportation), network problems, and so on. This common feature can be exploited very effectively by multi-level (multigrid) solvers, which combine local processing on different scales with various inter-scale interactions. Even when the governing relations are not strictly local (e.g., integral and integro-differential equations, x-ray crystallography, tomography, econometrics), any problem with a multitude of unknowns is likely to have some internal structure which can be used by multilevel solvers. In many cases, the computational cost of such solvers has been shown to be essentially as low as the cost can ever be; that is, the amount of processing is not much larger than the amount of real physical information.

This article is a brief survey of this field of study, emphasizing important recent developments and their implications. No attempt is made to scan the fast-growing multigrid literature. (A list of more than 600 papers will appear in [24]; see also the multigrid books [21], [25], [7], [28], [20], [26], [22].) A more detailed account will be given in a future version of [7].

Multigrid methods were first developed (see historical note, Sec. 16) as fast solvers for discretized linear elliptic PDEs (see Secs. 3, 4, 5), then extended to non-elliptic (Sec. 6), nonlinear (Sec. 7) and time-dependent (Sec. 10) problems, and to more general algebraic systems (Secs. 2, 11). The multigrid apparatus has also been used to obtain improved discretization schemes (Sec. 8), and is especially effective in treating compound problems and sequences of many similar problems (Sec. 9). Recently, mainly in response to current computational bottlenecks in theoretical physics, new types of multi-level methods have been developed for solving large lattice equations (e.g. Dirac equations in gauge fields – Sec. 11); for calculating determinants (Sec. 12) and accelerating Monte-Carlo iterations (Sec. 14); and for discrete-state and highly-non-quadratic minimization (Sec. 13), the latter being applicable to spin systems and also to image reconstruction, crystallography, protein folding and combinatorial minimization. Multilevel linear programming is reported in Sec. 15. The Appendices describe in more details some recent developments.

2. Slow Components in Matrix Iterations

Consider the real linear system of equations

$$Ax = b \tag{2.1}$$

where A is a general $n \times m$ real matrix. For any approximate solution vector $\tilde{x}$, denote the error vector by $e = x - \tilde{x}$, and the vector of residuals by $r = Ae = b - A\tilde{x}$. Given $\tilde{x}$, it is usually easy to calculate r – especially when A is a sparse matrix; e.g., when A is based on local relations. One can then easily use these residuals to *correct* $\tilde{x}$; for instance, by taking one residual r_i at a time, and replacing $\tilde{x}$ by $\tilde{x} + (r_i/a_i a_i^T)a_i^T$, where a_i is the i-th row of A (thus projecting $\tilde{x}$ onto the hyperplane of solutions to the i-th equation). Doing this for $i = 1, \ldots, n$ is called a Kaczmarz *relaxation sweep*. It can be shown (Theorem 3.4 in [9]) that the convergence to a solution x (if one exists), of a sequence of such (or other) relaxation sweeps, should slow down only when

$$|\bar{r}| \ll |e|, \tag{2.2}$$

where $\bar{r}$ is the *normalized* residual vector ($\bar{r}_i = a_i e/|a_i|$) and $|\cdot|$ is the Euclidean (ℓ_2) norm. From the normalization of $\bar{r}$ it is clear that, for most error vectors, $|\bar{r}|$ is comparable to $|e|$; (2.2) can clearly hold only for special error vectors, dominated by special components (eigenvectors with small eigenvalues), whose number is small. Thus, when relaxation slows down, the error can be approximated by vectors in some much-lower dimensional space, called the *space of slow components*.

The concrete characterization of slowness depends on the nature of the problem, and is sometimes far from trivial (see e.g. the "multiple representations" in Sec. 8). In many cases of interest, however, we will now see that slowness simply means smoothness (see Sec. 11 for a generalization).

3. Discretized Differential Equations

In case the system (2.1) represents a discretization of a stationary partial-differential equation $Lu = F$ on some grid with meshsize h, we customarily rewrite it in the form

$$L^h u^h = F^h, \tag{3.1}$$

where u^h is a grid function. Barring cases of alignment (see Sec. 6), such a system is numerically stable if and only if L^h has a good *measure of ellipticity on scale* h, inherited either from a similar h-ellipticity measure of L, or (e.g. in case L is non-elliptic) from artificial ellipticity introduced either by "upstream" differencing

or through explicit "artificial viscosity" terms. (Ellipticity measures on uniform grids, and their scale dependence, are discussed in [7, §2.1].)

For any h-elliptic operator L^h, relation (2.2) holds if and only if the error is smooth on the scale of the grid; i.e., iff its differences over neighboring grid points are small compared with itself. (This in fact is exactly the meaning of h-ellipticity.) The space of slow components can therefore be defined as the space of grid-h functions of the form $I_H^h v^H$, where v^H are functions on a *coarser grid*, with meshsize $H > h$, and I_H^h is an interpolation operator from grid H to grid h.

The coarse grid should not be too coarse; $H = 2h$ or so is about optimal: It keeps on one hand H close enough to h, so that all errors which cannot be approximated on grid H are so highly oscillatory that their convergence by relaxation on grid h must be very fast (convergence factor .25 per sweep, typically). On the other hand $H = 2h$ already yields a small enough number of coarse grid points, so that the work associated with the coarse grid (in the algorithms described below) is already just a fraction of the relaxation work on the fine grid.

Let $\tilde{u}^h$ be an approximation to the solution u^h, obtained for example after several relaxation sweeps. To define a coarse-grid approximation v^H to the smooth error $v^h = u^h - \tilde{u}^h$, one approximates the "*residual equation*"

$$L^h v^h = r^h \underset{\text{def}}{=} F^h - L^h \tilde{u}^h \tag{3.2}$$

by the coarse-grid equation

$$L^H v^H = I_h^H r^h \tag{3.3}$$

where I_h^H is a fine-to-coarse interpolation (local averaging in fact, sometimes called "weighting" or "restriction") and L^H is a coarse grid approximation to L^h. One can either use the Galerkin-type approximation $L^H = I_h^H L^h I_H^h$, or derive L^H directly from L by differencing (replacing derivatives by finite differences) on grid H, which is less automatic but often also far less expensive in computer time and storage. A generally sensible approach is to use *compatible coarsening*, i.e., the Galerkin approach when L^h itself has been constructed by Galerkin (or variational) discretization, and direct differencing in case L^h itself is so derived, using the same discretization order and "double discretization" (see Sec. 10) as used by L^h, etc. (see discussion in [7, §11]).

A *coarse grid correction* is the replacement of $\tilde{u}^h$ by $\tilde{u}^h + I_H^h v^H$. Using alternately a couple of relaxation sweeps and a coarse grid correction is called a *two-grid cycle*.

There is no need of course to solve (3.3) exactly. Its approximate solution is most efficiently obtained by again alternately using relaxation sweeps (now on grid H) and corrections from a still coarser grid ($2H$). We thus construct a sequence of grids, each typically being twice as coarse as the former, with the coarsest grid containing so few equations that they can be solved (e.g., by Gaussian elimination) in negligible time.

A multigrid cycle for improving an approximate solution to (3.1) is recursively defined as follows: If h is the coarsest grid, solve (3.1) by whatever method. If not, denoting by H the next coarser grid, perform the following three steps: (A) ν_1 relaxation sweeps on grid h; (B) a coarse grid correction, in which (3.3) is approximately solved by starting with $\tilde{v}^H = 0$ and improving it by γ multigrid cycles; (C) ν_2 additional relaxation sweeps on grid h.

The *full multigrid algorithm* N-FMG for solving (3.1), when h is not the coarsest grid and H is the next coarser, is recursively defined as follows: (A) Solve $L^H u^H = F^H$ by a similar N-FMG algorithm, where $F^H = I_h^H F^h$. (F^H may also be derived directly from F.) (B) Start with the first approximation $\tilde{u}^h = \bar{I}_H^h u^H$, and improve it by N multigrid cycles. The solution interpolation $\bar{I}_H^h$ has usually a higher order than the correction interpolation I_H^h mentioned above.

For almost any discretized stationary PDE problem, a 1-FMG algorithm, employing cycles with $\nu_1 + \nu_2 = 2$ or 3 and $\gamma = 1$ or 2, is enough for solving (3.1) *to the level of truncation errors* (i.e., to the point where the approximate solution $\tilde{u}^h$ satisfies $\| \tilde{u}^h - u^h \| \leq \| u^h - u \|$, in any desired norm) – provided proper relaxation and interpolation procedures are used (see Sec. 5). Only when L^h has a high approximation order p, larger-N-FMG may be required, with N growing linearly in p.

This means that the solution is obtained in just few L^h-work-units, where an L^h-work-unit is the amount of computer operations involved in just *expressing* L^h at all grid-points. The only solvers with an almost comparable (but on large grids still inferior) speed are the direct solvers based on the Fast Fourier Transform (FFT), but they are essentially limited to equations with constant coefficients on rectangular domains and constant boundary operators. The FMG solver, by contrast, attains the same efficiency for general nonlinear, not necessarily elliptic, problems (see Secs. 6, 7), for any boundary shape and boundary conditions, for compound problems (Sec. 9), for eigenproblems, and for problems including free surfaces, shocks, reentrant corners, discontinuous coefficients and other singularities.

Moreover, the multigrid solvers can fully exploit very high degrees of parallel and/or vector processing. In case L^h is the standard 5-point approximation to

the Laplacian, for example, (3.1) has been solved on the CDC CYBER 205 at the rate of 5 million equations per second [3]. Also, for little extra computer work these solvers can incorporate local grid adaptation (Sec. 7) or provide a sequence of extra solutions to a sequence of similar problems (Sec. 9).

5. Performance Prediction, Optimization, and Rigorous Analysis

The multigrid algorithms have many parameters, including their relaxation schemes, orders of interpolations, their treatment of boundaries and of the interior equations near boundaries, etc. To obtain their best performance, and to debug the programs, an analytical tool is needed which can predict, for example, the precise convergence factor per cycle. Such a tool is the following *local mode analysis*.

For equations with constant coefficients on infinite uniform grids, only few (ℓ, say) Fourier components of the error function $\tilde{u}^h - u^h$ are coupled at a time by the processes of the two-grid cycle, and it is thus easy to calculate (usually by a small computer program) the two-grid convergence factor (the largest among the spectral radii of the corresponding $\ell \times \ell$ transfer matrices). For general equations in a general domain, the *local two-grid convergence factor* is defined as the worst (largest) two-grid convergence factor for any "freezing" of the equation at any given point (extending that equation to the infinite domain).

For a general elliptic system of equations $Lu = F$ with continuous coefficients, discretized on a uniform (or continuously changing) grid in a general domain, it has been proved ([10]; see App. C) that for small meshsizes ($h \to 0$) the local two-grid convergence factor is actually obtained globally, provided the algorithm is allowed to be modified near boundaries, by adding there local relaxation sweeps that cost negligible extra work. Numerical tests clearly show that this local relaxation is indeed sometimes necessary, e.g., near re-entrant corners and other singularities [2, §4]. The performance of *multi*grid cycles can also be precisely predicted, either by perturbations to the two-grid analysis or by more complex (e.g., three-level) Fourier analyses (coupling more components at a time).

Moreover, it can also be proved that the two-grid convergence factor, λ, can itself be anticipated by the "smoothing factor" of the relaxation process, $\bar{\mu}$, which can be calculated by a much simpler local mode analysis. Namely, $\lambda = \bar{\mu}^s$ can always be obtained, provided s, the number of fine-grid relaxation sweeps per cycle, is not large, and provided suitable inter-grid transfers (high enough interpolation orders) are used. Furthermore, in case of a complicated system of q differential equations, i.e., when L is a $q \times q$ matrix of differential operators, a relaxation scheme can always be constructed for which $\bar{\mu} = \max(\bar{\mu}_{L_1}, \ldots, \bar{\mu}_{L_k})$, where $L_1 \cdots L_k$ is a factorization, into first and second order scalar operators, of the h-principal part (the principal part on scale h) of the determinant of L, and $\bar{\mu}_{L_i}$ is the smoothing

factor obtainable for a relaxation of L_i^h (see [7, §3.7]). Thus, the entire multigrid efficiency can be anticipated from the smoothing factors obtainable for simple scalar operators, and the practical task then is to construct the intergrid transfers so that λ indeed approaches $\bar{\mu}^s$, and then to adjust the boundary processes until the convergence factor per multigrid cycle indeed approaches λ.

In case of uniformly elliptic problems, for example, the factors of $\det L$ are usually Laplacians, for which the smoothing factor $\bar{\mu} = .25$ is obtainable, using the (fully-parallelizable and extremely cheap) Gauss-Seidel relaxation in red-black ordering. Hence a multigrid cycle can be constructed with convergence factors .25 per fine-grid relaxation, or about .4 per work unit (taking coarse-grid overhead into account).

For highly discontinuous equations or discretizations, the theoretical treatment is far less precise, but practical approaches were developed [1], successful enough to yield fairly general black-box solvers [15].

Many situations are analyzed by *non-local* theories, developed over a vast literature; see e.g. [20] and references therein. The trouble with the non-local approach is that its estimates are not realistically quantitative: the convergence factor per cycle *is* indeed shown to be bounded away from 1 independently of h, but its actual size is either not specified or is so close to 1 that it is useless for practical purposes (such as prediction and selecting, omtimizing and debugging the various processes), and no one believing it would use the algorithm. In fact, it led to several practical misconceptions [7, §14].

The theory in [9] gives rigorous realistic two-grid convergence estimates for very irregular cases, in fact for general symmetric algebraic systems without any grids or any other geometrical basis. This This theory is nearly optimal for the crude (geometry-less) interpolations it considers. To extend it to the prediction of the *multi*-grid rates obtainable with better (geometrically-based) interpolations, it should be combined with some local analysis, not yet developed.

6. Non-Ellipticity and Slight Ellipticity

For non-elliptic differential equations (or equations with small ellipticity measures on scale h, which for numerical purposes is the same), it is a mistake to try to obtain uniformly fast convergence per cycle. Much simpler and more efficient algorithms are obtained by allowing components with larger truncation errors (such as the "characteristic components") to converge slower, insisting only that the 1-FMG algorithm still solves the problem well below truncation errors. That this can be obtained is shown by modified types of local mode analysis (infinite-space FMG analysis supplemented by half-space FMG analysis. See [6]).

Indeed, the usual FMG algorithm need only be modified in case of *consistent alignment*, i.e., in case the grid is *consistently* aligned with the characteristic directions. Such alignment is necessary when accuracy is desired in the "characteristic components", i.e., components which are smoother along than across characteristic lines. For obtaining that accuracy, L^h should be non-h-elliptic, and the usual point-by-point relaxation will then smooth the error only in the characteristic directions (in which semi-h-ellipticity is necessarily still maintained). One should therefore either modify relaxation, by *simultaneously* relaxing points along characteristic lines ("line relaxation"), or use "semi coarsening", i.e., a coarser grid whose meshsize is larger only in the characteristic directions. Semi coarsening, sometimes combined with line relaxation, is especially recommended in higher-dimensional situations where the alignment is not in lines but in planes.

Expensive procedures of *alternating-direction* line or plane relaxation are not needed in natural coordinates, since only *consistent* alignment matters in solving to the level of truncation errors. Such expensive procedures *will* however very *often* be needed if anisotropic coordinate transformations, and nonuniform gridline spacings in particular, are employed, thereby artificially creating excessively strong, grid-aligned discrete couplings. It is therefore generally not recommended to use global grid (or coordinate) transformations, but instead to create local refinements and local grid curvings in the multigrid manner (see Sec. 7).

For non-elliptic or slightly elliptic problems it is also recommended to use double discretization schemes (see Sec. 8), since some natural (e.g. central) discretizations are good for smooth components but bad for non-smooth ones.

7. FAS: Nonlinear Equations, Local Grid Adaptation, τ Extrapolation, Small Storage

In the *Full Approximation Scheme (FAS)* the coarse-grid unknown v^H is replaced by the unknown $u^H \underset{\text{def}}{=} \hat{I}_h^H \tilde{u}^h + v^H$, where $\hat{I}_h^H$ is another fine-to-coarse interpolation (or averaging). In terms of u^H, the coarse grid equation (3.3) becomes

$$L^H u^H = F^H + \tau_h^H, \tag{7.1}$$

where $F^H = I_h^H F^h$ and $\tau_h^H = L^H \hat{I}_h^H \tilde{u}^h - I_h^H L^h \tilde{u}^h$. This equation evidently has the form of a "defect correction" (correcting L^H by L^h, their difference being measured by $\tilde{u}^h$), hence it makes full sense even in the case that L is nonlinear.

Indeed, using FAS, nonlinear equations are solved as easily and fast as linear ones. No linearization is required (except for some local linearization, in relaxation, into h-principal terms, which in almost all cases means no linearization

at all). The 1-FMG algorithm has solved, well below truncation errors, various flow problems, including compressible and incompressible Navier-Stokes and Euler equations, problems with shocks, constrained minimization problems (complementarity problems, with free surfaces) and many others. "Continuation" techniques, sometimes needed for reaching the solution "attraction basin", can be incorporated for little extra calculations (see Sec. 9).

In FAS, averages of the full solution are represented on all coarser grids (hence the name of the scheme). This allows for various advanced techniques which use finer grids very sparingly. For example, the fine grid may cover only part of the domain: outside that part (7.1) will simply be used without the τ_h^H term. One can use progressively finer grids at increasingly more specialized subdomains, effectively achieving a non-uniform discretization (needed near singularities) which still uses simple uniform grids, still has the very fast multigrid solver, and yet is very flexible. Grid adaptation can in fact in this way be incorporated into the FMG algorithm: On proceeding to finer levels the algorithm also defines their extent (see [5],[2]). Moreover, each of the local refinement grids may use its own local coordinate system, thus curving itself to fit boundaries, fronts, characteristic directions or discontinuities (all whose locations are already approximately known from the coarser levels), with the additional possibility of using anisotropic mesh-sizes (e.g. much finer across than along the front). Since this curving is only local, it can be accomplished by a trivial transformation, which does not add substantial complexity to the basic equations (in contrast to global transformations).

The fine-to-coarse correction τ_h^H gives a rough estimate of the local discretization error. This can be used in *grid adaptation criteria*. It can also be used to *h-extrapolate* the equations, in order to obtain a higher order discretization for little extra work. This extrapolation is more useful than the Richardson type, since it is local (extrapolating the equation, not the solution): it can for example be used together with any procedure of local refinements.

In view of (7.1), the role of grid h is really only to supply the defect correction τ_h^H to grid H. For that, only a local piece of the fine grid is needed at a time. Similarly, only a piece of grid $H = 2h$ is needed at a time, to supply τ_{2h}^{4h}, etc. This gives rise to algorithms that can do with *very small computer storage* (even without using external storage).

The above *local refinements*, *local coordinates*, *refinement criteria*, *local h extrapolations* and *small-storage techniques* were examples of using the multilevel apparatus to obtain better *discretizations*, not just fast solvers. Other examples are:

Double discretization schemes. The discrete operator L^h used in calculating the residuals (3.2), for the global process of coarse grid corrections, does not need to coincide with the one used in the local process of relaxation. The latter should have good *local* properties, such as stability (possibly obtained by adding artificial viscosity) and admittance of sharp discontinuities (through suitable "limiters"), while the first should excel in *global* attributes, such as high accuracy (obtained by omitting artificial viscosities and possibly using higher-order differencing) and conservation (through conservative differencing). Such schemes do not converge to zero residuals, of course, but can approximate the differential equations much better than either of their constituent discretizations alone, especially in cases of conflicting requirements (cf. Sec. 6).

Multiple representation schemes. The coarse-grid solution representation does not need to coincide with that on the fine grid. For example, some nearly singular smooth components (typical in slightly indefinite problems) should on some coarser grids be singled out and represented by one parameter each (see [14]). Or, more importantly, highly oscillatory components showing small normalized residuals (typical in standing wave problems, as in acoustics, electromagnetism, Schrödinger equations, etc.) should be represented on coarser grids by their slowly varying amplitudes. The coarser the grid the more such "rays" should be separately represented. Grids fine enough to resolve the natural wavelength can be used only locally, near boundary singularities, where ray representations break down. This hybrid of wave equations and geometric optics can treat problems which neither of them can alone, in addition to supplying a fast solver for highly indefinite equations.

Global conditions and non-local boundary conditions (radiation conditions, flow exit boundaries, etc.) are easily incorporated, by transferring their residuals from fine grids and imposing them only at suitably coarser levels.

Treating large domains by placing increasingly coarser grids to cover increasingly wider regions.

Fast integrals. In case of integral equations with suitably smooth kernels, most of the work involved in just performing the integrations can be spared, by performing them mainly on coarser grids using suitable FAS versions (see App. A).

Finally, multigrid convergence factors always *detect bad discretizations*, es-

pecially when "compatible coarsening" is used (see Sec. 3). Several previously unnoticed flaws in widely accepted discretization schemes were so discovered. Furthermore, brief 1-FMG algorithms tend to *correct bad discretizations*, by being very slow in admitting ill-posed components (components showing small residuals compared with other components of comparable smoothness). For example, quasi-elliptic discretizations (resulting e.g. from central differencing on non-staggered grids of elliptic systems with first-order principal derivatives) are so solved with their highly-oscillating bad components left out [13]. More generally, the FMG algorithm and the multi-level structure provide effective tools to deal with *ill-posed problems*, whether the ill-posedness is in the differential problem or only in its discretization: finer grids can be introduced (in the manner of Sec. 7) only where their scale does not admit ill-posed components; nonlinear controlling constraints, either global, local or at any intermediate scale, are easily incorporated; etc.

9. Compound Problems and Problem Sequences

A compound problem is one whose solution would normally involve solving several, or even many, systems of equations similar to each other. With multilevel techniques, the work of solving a compound problem can often be reduced to that of solving just one single system, or just a fraction more.

Take for example *continuation* (embedding) processes, in which a problem parameter is gradually changed in order to drive the approximate solutions into the attraction basin of the desired solution to some target nonlinear problem. Flow problems, for instance, are easily solved for the case of large viscosity, which can then gradually be lowered to the desired level, with the equations being solved at each step taking the previous-step solution to serve as a first approximation. This process is almost automatically performed by the FMG algorithm (Sec. 4) itself, since it starts on coarse levels, where a large artificial viscosity is introduced by the discretization, and then gradually works its way to finer grids with proportionately smaller viscosity. The process, by the way, can then be continued to still lower viscosity by using still finer levels only locally (see Sec. 7), at regions where the size of viscosity matters (i.e., where the flow is driven by viscosity), and eliminating viscosity elsewhere (e.g. by double discretization – see Sec. 8).

One 1-FMG algorithm, with no extra iterations, can even be directed to *locate limit points* (turning and bifurcation points) on solution diagrams; or to *optimize* some problem parameters, including optimization of boundary shapes, diffusion coefficients, control parameters, etc.; or to *trace* free boundaries, strong shocks, and other discontinuities; or to solve related *inverse problems* (e.g. system identification); and so on – all with accuracy below truncation errors.

In many cases, however, *repeated* applications of the FMG solver are still

needed: cases of complicated bifurcation diagrams, interactive design situations, etc. Even then, the multigrid machinery generally provides for extremely cheap *re*-solving: one should only be careful to apply FMG to the *incremental* problem (calculating only the *change* from the old solution; using FAS this is easily done even in nonlinear problems) and to skip finer grids (or parts thereof) wherever they describe negligible high-frequency *changes*.

In designing a structure, for example, one often wants to *re*-solve the elasticity equations after modifying some part of the structure. The *changes* in the solution are then very smooth, except near the modified part. In *incremental* re-solving one therefore needs the fine grid h only near that part, while at other regions the coarser grid H can suffice – provided the τ_h^H correction (see (7.1)) is kept in those regions frozen at its previous (pre-modification) values (otherwise one ignores the high frequency components themselves, not just their changes). Similarly, at some larger distance from the modified part, grid $H = 2h$ itself can also be omitted, then grid $4h$, etc. In this way re-solving can be so inexpensive in computer time and storage as to allow on-line interactive design of complicated structures. Similar *frozen-τ techniques* can be used in continuation processes and in evolution problems.

10. Evolution Problems

Some time-dependent problems may need no multileveling. These are hyperbolic schemes where all the characteristic velocities are comparable to each other, and their explicit discretization on one grid is therefore fully effective: the amount of processing is essentially equal to the amount of physical information. However, as soon as any stiffness enters, implicit discretization and multigrid techniques similar to those in Sec. 9 become desired.

Solving the sequence of implicit systems, the 1-FMG algorithm is all one needs per time step – provided it is consistently applied to the time *incremental* problem, since one needs to solve to the level of the incremental (not the cumulative) truncation errors. Moreover, in most cases, notably in parabolic problems, this work can vastly be reduced, because most of the time at most places the increment is very smooth, hence seldom requires fine-grid processing.

For example, it has been demonstrated for the heat equation $\partial u/\partial t = \Delta u + F$ with steady boundary conditions and steady sources F that, given any initial conditions at $t = 0$, the solution at *any* target time T can be calculated, to the level of spatial truncation errors, in less than 10 work units, where the work unit here is the work invested in one *explicit* time step. To obtain the solution with that accuracy *throughout* the interval $0 \le t \le T$, the number of required work

units is $O\left(\log \frac{T}{h^2}\right)$.

By combining methods developed for such purely parabolic problems with the method of characteristics, it may be possible to obtain similar results for problems with *convection*, because the time increment can be described as a smooth change superposed on pure convection.

All *multigrid discretization techniques* (see Sec. 8) can be useful for time dependent problems, too. One example: the popular Crank-Nicholson discretization, which offers superior accuracy for smooth components, has the disadvantage of badly treating high-frequency components at large time steps. This conflict is easily resolved by a double discretization scheme, which at some initial time steps, and only at the fine grids' relaxation process, replaces Crank-Nicholson by the Fully-Implicit scheme. Other examples that were already used include local refinements, the τ refinement criteria, τ extrapolations, and a treatment of an ill-posed (the inverse heat) problem.

Time-periodic solutions, or more generally, solutions with the same solution growth w per time period, can inexpensively be computed, for any spatial grid h, by integrating basically on grid $2h$: once a steady growth ω^{2h} has been established on grid $2h$, a defect correction to w^{2h} can be found by integrating one period on grid h; then the calculations on grid $2h$ resume, with that defect added at each period, until a new steady growth is established. The calculations on grid $2h$ can similarly be done by integrating basically on grid $4h$, and so on. Each grid integration may of course also use the above frozen τ techniques.

11. Geometrically Based Problems. AMG

Most large systems, even those not derived from discretized continuous problems, still have a geometric basis; that is, each unknown has a location in some low (usually at most 4) dimensional underlying space – indeed, the unknowns are often still arranged in lattices – and the equations reflect this geometry, e.g. by more strongly coupling closer unknowns. Examples abound (see Sec. 1). Excluding for the moment probabilistic aspects (see Sec. 14), these systems can usually be cast as minimization problems: the solution vector u should minimize some functional $E(u)$, called "energy". This naturally leads to various Gauss-Seidel-type relaxation schemes, in which E is decreased as far as possible by changing one unknown (or one block of unknowns) at a time. (Kaczmarz relaxation in Sec. 2 can be viewed as Gauss-Seidel for $\bar{u}$, where $u = A^T\bar{u}$ and $E(u) = \frac{1}{2}u^Tu - \bar{u}^Tb$).

Excluding now the case of discrete or partly discrete unknowns (see Sec. 13), in all such geometrically-based systems the slow components (see Sec. 2) are either "smoothly representable" or ill-posed. A general smooth representation of compo-

nents is for example by short sums of terms such as $a(x)\varphi(x)$, where $a(x)$ is smooth (at least in some directions) while $\varphi(x)$ may be highly non-smooth but is fixed and known (or easily computable). A multilevel solver can then be constructed in which $a(x)$ is interpolated from coarser levels. The coarser level equations may be derived either variationally (i.e., from the requirement that $E(u)$ is lowered as far as possible by the interpolated $a(x)$), or by simulating direct differencing approximations.

A multigrid solver of the latter kind has been constructed for a simple case of lattice Dirac equations in a gauge field. In QED and QCD (quantum electrodynamics and chromodynamics) simulations, this type of equations should be solved at each Monte-Carlo iteration, consuming enormous computer resources (see e.g. [18]). This solver, which employs itself also for updating $\varphi(x)$, exhibits the usual multigrid speed, and requires only a short cycle, costing far less than the rest of the calculations, per Monte-Carlo iteration. (See also Sec. 12.)

In many problems, including first-kind integral equations in fields like image reconstruction, tomography and crystallography, there exist slow components which are not smoothly representable. Since they give large errors for small residuals without being smooth in any sense, they are by definition ill posed. Such error components are introduced only very slowly by the multigrid solvers. Hence they are harmful only in as far as their absence causes the solution to "look bad". Specifying what "looking bad" is, can be done by augmenting $E(x)$ and/or by imposing nonlinear constraints. Such constraints, on any scale, can be incorporated in the multilevel solver (see [11]), even when they are discrete (see Sec. 13).

Multilevel solvers can be constructed even when the geometric basis is not explicit. In such "*algebraic multigrid*" *(AMG)* solvers the coarse-level variables are typically selected by the requirement that each fine-level variable is "strongly connected", by the fine-level equations, to at least some coarse-level variables. The coarse-to-fine and fine-to-coarse transfers may also be purely based on the algebraic equations, although geometrical information may be used too (see [9], [29]). AMG solvers are good as black boxes, even for discretized PDEs, since they require no special attention to boundaries, anisotropies and strong discontinuities, and no well-organized grids (allowing, e.g., general-partition finite elements).

12. Calculating Determinants

At each Monte-Carlo iteration in QED and QCD simulations, what is really required is not to *solve* the lattice Dirac equations (see Sec. 11), but actually (if possible) to calculate $\delta \log\det Q$, where Q is the matrix of that system and δ denotes change per iteration. Since the steps are small, $\delta \log\det Q \approx$ trace of $Q^{-1}\delta Q$, for which calculations one needs to know $(Q^{-1})_{ij}$ for all pairs of neighboring (on

the lattice) i and j. Now, it can be shown that by storing and updating similar information for coarse-grid approximations to Q (for which purpose one also needs to store and update the function $\varphi(x)$ mentioned in Sec. 11), all updates can immediately be done. The implied coarse-level work, including the coarsening of Q, is just a small overhead.

This approach leads to a general fast method for calculating determinants of lattice equations.

13. Discrete-State Minimization: Multilevel Annealing

In statistical physics, combinatorial optimization (e.g., traveling salesman, or integrated circuits design), pattern recognition, econometrics, and many other fields the unknowns u_i, or part of them, may only assume discrete states. A typical example is Ising spins, where $u_i = \pm 1$. To minimize $E(u)$ in such problems is far more intricate than in continuous-state problems, since the relaxation process is not only slow, but is very likely to get trapped in a "local minimum"; i.e., in a configuration u which is not the true minimum but for which no allowable change of any one u_i, or even a small block of them, can lower E.

"*Simulated annealing*" is a general technique for trying to escape such local minima by assigning at each step a certain probability for the energy to grow. This is done by simulating thermal systems: to each configuration u the "Boltzmann probability"

$$P(u) = e^{-\beta E(u)}/Z(\beta) \tag{13.1}$$

is assigned (physically $\frac{1}{\beta}$ is proportional to the absolute temperature and $Z(\beta)$ is a normalization factor), and the above strict-minimization relaxation sweeps are replaced by "Monte-Carlo iterations", in which each u_i change is governed by (13.1). Gradually and carefully β is increased (the system is "cooled") so that the Monte-Carlo process tends back to strict minimization. (See [23].)

In many cases, unfortunately, the global minimum is likely to be reached only if β is increased impractically slowly, requiring exponentially growing computer times, or else the process will be trapped in some local minimum with a large "attraction basin" (usually containing smaller-scale sub-basins from which the process does escape). This difficulty is removed by *multilevel annealing*, based on the following principles:

(i) A hierarchy of changes is selected. In two-dimensional Ising spin lattices, for example, a change on level ℓ is defined as the simultaneous flipping (sign reversal) of all the spins in a $2^\ell \times 2^\ell$ block. (ii) Each coarse-level change is decided only *after* recursively calculating its effects (i.e., minimizing around it) at all finer

levels, starting from the finest. (iii) At each level a specific β, just large enough to escape local minima on that scale, is first employed, then, still at that level, strict minimization follows. (iv) A procedure (LCC) for keeping track of the so-far minimal configuration is added at each level. (See [12].)

These principles were applied to difficult two-dimensional lattice problems with N Ising spins. The global minimum has always been reached in $O(N^{3/2})$ to $O(N^2)$ computer operations. The parallel-processing complexity is polynomial in $\log N$. Similar algorithms are being developed for the traveling salesman problem. (The "statistical" TSP with N cities is solved in $O(N)$ operations).

The above principles should also apply in many problems where the discrete-state nature is less obvious. Take for example XY spins or Heisenberg spins, where each u_i is a 2 or 3 dimensional vector *of length 1.* Although each u_i can change continuously, some large-scale topological features of the field of spins (such as the existence of closed curves along which the spins gradually rotate a full circle) can only change discretely. Similar situations arise in x-ray crystallography and protein folding problems. Another example: in image reconstruction, each unknown u_i, representing the grey level in the i-th pixel, can be considered continuous, but nonlinear constraints that should be added to the problem (cf. Sec. 11) may well include discrete elements, such as the appearance of an "edge". In each of these cases a certain combination of the multilevel annealing with classical multigrid should be used. More generally, coarse-level annealing should apply in any minimization problem with large-scale local minima, and *multilevel annealing is required whenever a hierarchy of attraction basins is involved.*

14. Statistical Problems. Multilevel Monte-Carlo

The aim in statistical physics is to calculate various average properties of configurations governed by the probability distribution (13.1). This is usually done by measuring those averages over a sequence of "Monte-Carlo iterations", in which each u_i in its turn is randomly changed in a way that obeys (13.1) (using e.g. Metropolis rule [27]). Unfortunately, in such processes statistical equilibrium is usually reached very slowly, and, more severely, even when it has been reached, some averages are still very slow to converge, especially those long-range correlations the physicist needs most.

These two troubles may be cured by multilevel Monte-Carlo techniques, in which coarse-level changes (changing the solution in preassigned blocks in preassigned patterns) are added and averaged over. In problems and at levels where the physical states may be considered continuous, this can be done quite straightforwardly and very efficiently: once per several coarse-level sweeps, the probabilities associated with coarse changes are defect-corrected by fine-level Monte-Carlo it-

erations (see [12, §7.1]). In case of discrete states, principles similar to those in Sec. 13 should apply. Namely, the exact *pattern* of each coarse-level change, as well as the probabilities associated with it, are recursively decided by finer-level Monte-Carlo. But see App. B.

15. Linear Programming (LP)

A multilevel approach, called iterative aggregation, has been developed LP problems (see [16], [31]), especially for situations in which the planned system is naturally divided into a hierarchy of sectors and sub-sectors. This considerably speeds up the calculations, and also provides the manager with a very useful hierarchical view of the system.

For very large systems, to obtain the typical speed of *multigrid* solvers, more refined aggregations are needed. This can easily be done, for example, in problems with a geometrical basis (cf. Sec. 11), such as the LP transportation problem (see e.g. [19]). Recent tests were made with a method that lumps together two (or so) neighboring destinations into a "block destination", two neighboring blocks into a super-block, etc. Shipping costs to a block are determined from the current intra-block marginal costs. It turns out that a 1-FMG-like algorithm gets very close (practically obtains) the solution. The required work is even smaller since many of the blocks that are supplied by one origin need no fine-level processing. Several orders of magnitude savings, compared to simplex solutions, were indicated.

16. Historical Note

Various multi-level solution processes have independently occurred to many investigators (see partial list in [5]). The earliest we know is Southwell's acceleration of relaxation by "group relaxation" [30], a two-level algorithm. The first to describe a recursive procedure with more than two levels is Fedorenko [17]. Similar approaches were early introduced to economic planning (see Sec. 15). All these early works lacked full understanding of the real efficiency that can be obtained by multileveling, and how to obtain it, since they did not regard the fine-grid processes as strictly local, hence thought in terms of too-crude aggregations. Fedorenko's estimates of the work involved in solving simple Poisson equations are off by a factor 10^4, for example. Fully efficient multigrid algorithms, based on local analysis, were first developed at the Weizmann Institute in 1970-1972 (see [4]), leading then to most of the developments reported in the present article.

Appendices

A. Integral Equations

When an integral equation of the general type

$$\int_\Omega K(x,y)u(y)dy + f\big(x,u(x)\big) = 0, \qquad x \in \Omega \subseteq \mathbb{E}^d \tag{A.1}$$

is discretized in a usual way on a grid with $n = O(h^{-d})$ points, the unknowns are all connected to each other; the matrix of the (linearized) discrete system is full. A solution by elimination would require $O(n^3)$ operations. An FMG solution would require $O(n^2)$ operations, since each relaxation sweep costs $O(n^2)$ operations. Even when (A.1) is ill posed (Fredholm equation of the first kind), the FMG solver can still be as effective (cf. Sec. 11). In case (A.1) is nonlinear in u, FAS-FMG should be used, still retaining the same efficiency (see Sec. 7). But potentially the most important contribution of the multilevel approach to the solution of integral equations is in reducing the work far below $O(n^2)$, sometimes to $O(n)$ and most often to $O(n \log n)$, by exploiting smoothness properties of K. (In fact, $O(n^2)$ solution time for second kind Fredholm equations is already nearly obtained by simple relaxation.) This is done by using the FAS structure in the following special way (first presented in [8, §8.6]).

The discretization of (A.1) on grid h has the form

$$\sum_{j=1}^{n} K_{ij}^{hh} u_j^h + f_i(x_i, u_i^h) = 0, \qquad (i = 1, \ldots, n) \tag{A.2}$$

where i and j are multi-indices, $x_i = ih$, and $u_i^h = u^h(x_i)$ approximates $u(x_i)$. Since $K(x,y)$ is fully known, (A.2) is essentially obtained by performing the integration in (A.1) with $u(y)$ being replaced by values polynomially interpolated from the grid values u_j^h. Hence the chosen discretization (e.g., the order of the polynomial interpolations), and its truncation errors, depend solely on the smoothness of u, not of K. If $K(x,y)$ as a function of y is much smoother than $u(y)$, an error much smaller than the truncation error would be introduced when K_{ij}^{hh} is replaced by $\tilde{K}_{ij}^{hh} = (\hat{I}_H^h K_{i\cdot}^{hH})_j$, where $K_{i\cdot}^{hH}$ is the representation of $K_{i\cdot}^{hh}$ (K_{ij}^{hh} as a function of j, for a fixed i) on the coarse grid H, and $\hat{I}_H^h$ is an interpolation from H to h of a sufficiently high order. This will replace each summation in (A.2) by $\sum_J K_{iJ}^{hH} u_J^H$, where J runs over the coarse grid and $u^H = (\hat{I}_H^h)^T u^h$, superscript T denoting adjoint (i.e., matrix transposition). Hence, choosing $\hat{I}_h^H = (\hat{I}_H^h)^T$ for the FAS fine-to-coarse solution averaging (see Sec. 7), the summation is done on the coarse grid. Moreover, if $K(x,y)$ is also thus smooth as a function of x, each K_{iJ}^{hH}

can similarly be replaced by interpolation from $K_{\cdot J}^{HH}$, so that those coarse-grid summations will actually be calculated only for coarse-grid values of i. If K is sufficiently smooth, one can similarly replace grid-H summations by summations on still coarser grids, etc. (with suitably growing interpolation orders). All this can very easily be incorporated into the FAS-FMG algorithm; it simply requires that, at each level h, $w_i^h = \sum_j \tilde{K}_{ij}^{hh} u_j^h$ is stored along with u_i^h, its values (on finer levels) being interpolated from level H along with the interpolation of u_i^h or its corrections.

It is easy to see that if the order of smoothness of K is twice that of u, this algorithm (with $w_i^{h_1}$ being used all the way to grid $h_1 = O(h^{1/2})$) will solve the problem to the level of truncation errors in $O(n)$ operations. In most physical problems the smoothness of $K(x,y)$ increases indefinitely with increasing distance $|x-y|$. In such cases the algorithm can be used (all the way to the coarsest h_1), but the values of w_i^h should be corrected (after being interpolated from w^H) by summation over some m points on grid h in the vicinity of x_i. For example, for potential-type equations (i.e., $K(x,y) = \log|x-y|$ or $K(x,y) = |x-y|^{-1}$) the algorithm should be used with $m = O(\log n)$, and the order of $\hat{I}_H^h$ should also be $O(\log n)$, resulting in $O(n \log n)$ solution time.

This algorithm can in the same way be used even when the given grid is non-uniform; e.g., when the given grid points represent an actual self gravitating set of point masses. To facilitate high order interpolations, the best way in this case may be to organize the next coarser grid in a semi-uniform structure, based on a collection of progressively finer uniform grids defined over increasingly more specialized subdomains (cf. Sec. 7).

Recently, an algorithm which solves potential-type equations in $O\big(n(\log n)^3\big)$ operations has been presented [33]. It seems to be substantially slower than the above algorithm, less general and more complicated.

B. Multilevel Monte-Carlo

Suppose a grid function u has the boltzmann distribution (13.1), and the task is to calculate averages $\langle M(u)\rangle = \sum_u P(u)M(u)$, for various functionals $M(u)$. Assume first that each u_j, the value of u at gridpoint $x_j = jh = (j_1,\ldots,j_d)h$, is a real number.

When $\beta \to \infty$ (zero temperature), the task boils down to finding the "ground state(s)", i.e., the configuration(s) for which $\min E(u)$ is attained. This problem is solved very effectively by the usual multigrid algorithm, that for simplicity can be described as the alternating use of the following two steps. (i) *Gauss-Seidel relaxation*: The gridpoints are scanned in some prescribed order; at each point

x_j in its turn, the value of u_j is changed so as to minimize E as far as possible. This process by itself can in many cases converge to the desired minimum, but the convergence will normally be slow, because smooth errors will be reduced very slowly. (ii) *Coarse-grid correction* is a correction of a given approximation $\tilde{u}$ by a function of the form $I_H^h v^H$, where v^H is a function defined on a coarser grid (e.g., with meshsize $H = 2h$), and I_H^h denotes coarse-to-fine (H to h) interpolation, whose weights in any direction should generally reflect the strength of interactions in that direction. v^H itself is selected so as to minimize $E(\tilde{u} + I_H^h v^H)$. To (approximately) calculate v^H, the resulting coarse-grid minimization problem is itself (approximately) solved by using again steps (i) and (ii). (Thus, a sequence of increasingly coarser grids is in fact recursively used.) Since v^H very well approximates smooth errors, the overall process converges very fast. (See Secs. 3, 4, 5. In case local minima are obtained instead of global ones, elements of multilevel annealing should be incorporated; cf. Sec. 13.)

For finite β, relaxation is replaced by a Monte-Carlo process: at each x_j in its turn, a new value of u_j is randomly chosen according to the probability distribution (13.1) (given that all other values of u are fixed at their current value). When this is done many times over, a sequence of configurations is generated with "*detailed balance*", i.e., with the property that, if at a certain stage in that sequence an *equilibrium* has been reached (meaning that the probability to have obtained any configuration u is the physical probability $P(u)$), then that will also be true in all subsequent stages, making the subsequent sequence representative enough for calculating the desired averages. In practice, *equilibration is slow*: equilibrium is (approximately) obtained only after many steps, because large-scale (i.e., smooth) deviations are slow to disappear. More seriously, even when equilibrium has been reached, the calculated averages are often very slow to converge and very expensive, because of the following four difficulties. (A) "*Critical slowing-down*" (typically occuring near critical temperatures, which are physically most important): the space of configurations is slowly sampled, because large-scale solution features are slow to change. (B) *Slow balancing*: Deviations at all scales are slowly averaged out. If a standard deviation σ is contributed by the features of some scale, these features have to completely change $O\big((\sigma/\varepsilon)^2\big)$ times in order to obtain accuracy ε. (C) *Domain vastness*. Large scale features, physically very important (especially close to critical temperatures), obviously require very large grids to be simulated. (D) *Fermionic interaction*. In QCD problems, at each Monte-Carlo iteration, to account for the dynamics of fermions, a system of lattice Dirac equations should be solved. Moreover, the change in the logarithm of the *determinant* of that system should in fact be calculated, possibly requiring enormous amount of calculations.

These four difficulties actually *multiply* each other, and therefore usually result in intractable calculations. Fortunately, they can all simultaneously be overcome by multilevelling.

We first describe how to eliminate the slow equilibration and the critical slowing. The first simple approach, in a straightforward analogy with the zero-temperature case, is to alternate the usual Monte-Carlo process with a *coarse-grid Monte-Carlo*, in which only changes of the form $I_H^h v^H$ are considered to a given fine-grid configuration $\tilde{u}$. Detailed balance is preserved if the Hamiltonian (energy) governing this coarse Monte-Carlo is $E^H(v^H) = E(\tilde{u} + I_H^h v^H)$. By some pre-calculation this Hamiltonian can be rewritten in a simple form, quite similar to the given (i.e., the fine-grid) Hamiltonian. (Using the FAS formulation, such a coarse-grid Hamiltonian can quite generally be devised, even when E is not quadratic; see [12, §7.1]. It is interesting to note that with this formulation, external fields can generally be viewed as defect-corrections of some finer structures to a coarser physics.) In some cases (e.g., when large local deviations are improbable), this coarse Monte-Carlo well represents all moves which are slow to equilibrate in the usual (fine-grid) Monte-Carlo. In such cases, a two-level cycle, composed of a couple of fine-grid sweeps followed by coarse-grid equilibration followed by an additional couple of fine-grid sweeps, will nearly equilibrate the fine-grid configuration. The coarse-grid equilibration itself can rapidly be (nearly) obtained by similarly alternating between sweeps on that grid and (near) equilibration on a still coarser grid. This recursively yields a *multigrid cycle*. Since coarser sweeps are computationally much cheaper, the total work in such a cycle is only a fraction more than the work invested in the fine-grid sweeps. Thus, equilibrium (and hence also decorrelation) is nearly obtained in a work equivalent to just few Monte-Carlo sweeps. (This has been demonstrated by Goodman and Sokal [34].)

In most cases of interest, as Murphy would predict, this straightforward approach will not quite work, mainly because the probable slow-to-equilibrate moves cannot generally be characterized as having the form $I_H^h v^H$. For example, this is obviously the case for Ising spins. To be sure, coarse-level moves of Ising spins *are* feasible: they would typically consist of the simultaneous flipping of $b \times b$ squares (or $b \times b \times b$ cubes); and $b^\ell \times b^\ell$ squares at the ℓ-th level of coarsening. But such plain square flips will most often increase the energy very much (the more so the coarser the level) and will therefore most probably be rejected by the Monte-Carlo process. To employ *probable* coarse moves, the blocks being flipped should tend to be broken along "weak links", i.e., at interactions which currently carry high energy (at violated bonds, in case of Ising spins). To obtain such blocks and still maintain detailed balance, we propose to employ the following *stochastic coarsening* process.

Take first, for example, the two-dimensional Ising spin model, with variables $u_{\alpha\beta} = \pm 1$ and Hamiltonian

$$E(u) = -\sum_{\alpha,\beta} u_{\alpha,\beta}(J_{\alpha,\beta} u_{\alpha+1,\beta} + K_{\alpha,\beta} u_{\alpha,\beta+1} + h_{\alpha,\beta}). \qquad (B.1)$$

The blocking described above (with $b = 2$, say) can be done in more steps, alter-

nating a horizontal coarsening with a vertical one. Consider for example the first horizontal coarsening. In the straightforward process, the flip of each coarse-grid spin $u^H_{\alpha,\beta}$ would represent the simultaneous flip of $u_{2\alpha,\beta}$ and $u_{2\alpha+1,\beta}$. This block *freezes* the interaction $V_{2\alpha,\beta}(u) = J_{2\alpha,\beta}u_{2\alpha,\beta}u_{2\alpha+1,\beta}$. In the *stochastic* coarsening process, we only assign a *probability* $1 - P_{2\alpha,\beta}$ for this freezing, while a probability $P_{2\alpha,\beta}$ is assigned to simply *deleting* this interaction from the Hamiltonian governing the next MC moves. It is easy to see that detailed balance is maintained provided (i) whatever is obtained, a freeze or a deletion, it is maintained for the same MC moves (e.g., for the entire coarse MC); (ii) the probability used is $P_{2\alpha,\beta} = q_{2\alpha,\beta}\exp\bigl(-\beta V_{2\alpha,\beta}(\tilde{u})\bigr)$, where $\tilde{u}$ is the current configuration and $q_{2\alpha,\beta}$ is any non-negative constant (independent of $\tilde{u}$, but depending on the interaction), sufficiently small to assure that $P_{2\alpha,\beta} \leq 1$. The blocking can now be changed: in case $V_{2\alpha,\beta}$ has been deleted, the spin $u_{2\alpha+1,\beta}$ is blocked with $u_{2\alpha+2,\beta}$, not with $u_{2\alpha,\beta}$. The flip of each coarse grid spin will now represent the flipping of between one and three fine-grid spins. (This is a simplified example. The actual process is somewhat more sophisticated.)

It is easy to see that the Hamiltonian of the coarse grid will no longer have the general form (B.1); interactions with diagonal neighbors may enter. But this is the most complicated the stencil can get: even if (B.1) included all eight (nearest plus diagonal) neighbor interactions, the above horizontal coarsening would still produce just eight coarse neighbor interactions. (More flexible coarsening schemes would allow the neighborhood to grow just a little more.) So the process can be repeated. At each level a couple of Monte-Carlo sweeps are first made, to settle to a local equilibrium, followed by stochastic coarsening first horizontally and then vertically. Then the coarse-level Monte-Carlo is performed (similarly using still coarser levels). On returning to the fine grid, each coarse-grid spin whose final value is different from its initial value is translated into flipping of the corresponding block of fine-grid spins. This whole cycle can be repeated γ times before returning to the *next* finer grid (when the current fine-grid is not the finest). The whole trip thus defined from the finest level through all coarser ones and back to the finest is called a *multi-grid cycle* (V cycle if $\gamma = 1$, W cycle if $\gamma = 2$).

If the maximal possible value is always assigned to $q_{2\alpha,\beta}$, deletion is sure to occur at each violated bond (i.e. when $V_{2\alpha,\beta}(\tilde{u}) < 0$). Hence, islands of reversed signs will be blocked together and easily disappear in the coarse Monte-Carlo. Also, new islands will easily appear. Most of the configuration will change in one cycle. The work in a cycle is dominated by the couple of finest-grid Monte-Carlo passes. Thus, decorrelation is obtained in a work equivalent to just a few MC sweeps.

A similar process can easily be devised for any discrete-state or continuous-state system. The Hamiltonian is first written in the form $E(u) = -\Sigma_j V_j(u)$, where the straightforward coarse-to-fine interpolation I^h_H would be equivalent to

freezing some of the interactions V_j. But instead of straightforward freezing, each such interaction is frozen in probability $1 - P_j$ and deleted in probability $P_j = q_j \exp\bigl(-\beta V_j(\tilde{u})\bigr)$, where the best value for q_j is perhaps the largest allowed, i.e., $q_j = \min_u \exp\bigl(\beta V_j(u)\bigr)$. Then the interpolation I_H^h is modified accordingly: its weights are larger in directions of stronger remaining interactions, keeping of coarse frozen all those interactions that escaped deletion. (The original I_H^h itself should best be based on the strength of the original interactions.)

Eliminating the critical slowing still leaves very significant *slow balancing*. Even at high temperatures, Ising spins should be flipped 10^{10} times in order to get five-digit accuracy in the average magnetization. At sufficiently high (or sufficiently low) temperatures, this slowness can be eliminated by changing the way statistics is extracted from the sequence of configurations; by replacing, for example, each Ising spin with its expected value given its neighborhood (thus regarding the Monte-Carlo sequence as a process for creating detail-balanced *neighborhoods*). Such balancing of deviations is still very effective even at quite moderate temperatures, if those neighborhoods are suitably enlarged. But close to the critical temperature, similar balancing needs to be done at all scales. This can naturally be done with the multilevel Monte-Carlo outlined above: the neighborhood of a spin is itself balanced in terms of the coarser level, and so on recursively.

The above multilevelling can also be used to deal with *vast domains*: the latter should be simulated only on coarse levels. The coarser the level, the larger its domain. This can be achieved in various ways, depending on the nature of the problem and the desired statistics. One simple way is still to use the finest grid over the *entire* domain, but with only few passes being made on that grid, since every such pass produces *many* fine-grid local samples, as against perhaps only *one* coarse-level sample produced by a pass on the coarsest level. A multigrid cycle, as described above, with $\gamma = 2^d$, for example, would produce comparable number of samples at all scales. If still larger domains are needed without more samples at the finest level, one can extend that level to the larger domain (using the periodicity – in case periodic boundary conditions have been used), then create from it the next coarser level by the stochastic coarsening process and delete the finest level from subsequent processing. (In some problems the finest level will still subsequently be processed at some special zones, e.g., near boundaries.) The domain may latter similarly be extended again and again, deleting each time the currently finest level.

The solution of the lattice *Dirac equations*, with associated matrix Q, and the calculation of $\delta \log\det Q$, is rapidly obtained by multilevel solvers (see Secs. 11 and 12). Moreover, these solvers can very nicely collaborate with the multilevel Monte-Carlo: they yield fine-to-coarse defect corrections to $(Q^{-1})_{ij}$ for neighboring i and j, so that $\delta \log\det Q$ can be followed also during coarse-level changes, without calculations on finer levels.

It is also expected that multilevelling in the above style, using FAS formulations, will lead to simple and natural renormalizations, i.e., descriptions of the system behavior in the limit of (infinitely) large scales. In case of Ising spins, for example, FAS formulation means that the initial value of a coarse spin is decided by the current configuration in the block it represents, e.g., by a majority rule. On increasingly coarser levels, the spin dynamics thus created, monitored by any statistics of interest, will tend to lose dependence on the level, and its "flow" between levels, as a function of various physical and computational parameters, can readily be studied. The advantage is that only simple interactions are maintained at all levels; e.g., only eight neighbor interactions are entailed by the simple example above. Still, sophisticated dynamics of superstructures can be performed at coarse levels, for negligible work, maintaining detailed statistical balance. Such techniques are therefore obvious candidates for treating turbulent flows, too.

Needless to say, the multilevel Monte-Carlo is highly parallelizable. With enough processors, solution time will be small-order polynomial in the number of *levels* employed.

References

[1] R.E. Alcouffe, A. Brandt, J.E. Dendy, Jr. and J.W. Painter: The multi-grid methods for the diffusion equation with strongly discontinuous coefficients. *SIAM J. Sci. Stat. Comp.* **2** (1981), 430–454.

[2] D. Bai and A. Brandt: Local mesh refinement multilevel techniques. *SIAM J. Sci. Stat. Comp.*, to appear.

[3] D. Barkai and A. Brandt: Vectorized multigrid Poisson solver. *Appl. Math. Comp.* **13** (1983), 215–227.

[4] A. Brandt: Multi-level adaptive technique (MLAT) for fast numerical solutions to boundary value problems. Proc. 3rd Int. Conf. Numerical Methods in Fluid Mechanics (Paris, 1972); Lecture Notes in Physics 18, Springer-Verlag, Berlin, pp. 82–89.

[5] A. Brandt: Multi-level adaptive solutions to boundary value problems. *Math. Comp.* **31** (1977), 333–390.

[6] A. Brandt: Multi-grid solvers for non-elliptic and singular-perturbation steady-state problems. Weizmann Institute of Science, December 1981.

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

[8] A. Brandt: Guide to Multigrid Development. In [21]. (Superseded by [7].)

[9] A. Brandt: Algebraic multigrid theory: The symmetric case. In [26].

[10] A. Brandt: Rigorous local mode analysis. Lecture at the 2nd European Conference on Multigrid Methods, Cologne, October 1985.

[11] A. Brandt and C.W. Cryer: Multi-grid algorithms for the solution of linear complementarity problems arising from free boundary problems. *SIAM J. Sci. Stat. Comp.* **4** (1983), 655-684.

[12] A. Brandt, D. Ron and D.J. Amit: Multi-level approaches to discrete-state and stochastic problems. In [22].

[13] A. Brandt and S. Ta'asan: Multigrid solutions to quasi-elliptic schemes. In: Progress and Supercomputing in Computational Fluid Dynamics (E.M. Murman and S.S. Abarbanel, eds.), Birkhäuser, Boston 1985, pp. 235–255.

[14] A. Brandt and S. Ta'asan: Multigrid method for nearly singular and slightly indefinite problems. In [22].

[15] J.E. Dendy: Black box multigrid. *J. Comp. Phys.* **48** (1981), 366–386.

[16] L.M. Dudkin and E.B. Yershov: Interindustries input-output models and the

material balances of separate products. *Planned Economy* **5** (1965), 54–63.

[17] R.P. Fedorenko: On the speed of convergence of an iteration process. *Ž. Vyčisl. Mat. i Mat. Fiz.* **4** (1964), 559–564.

[18] F. Fucito and S. Solomon: Concurrent pseudo-fermions algorithm. *Comp. Phys. Comm.* **36** (1985), 141.

[19] S.T. Gass: Linear Programming, 3rd ed., McGraw-Hill, New York, 1969.

[20] W. Hackbusch: Multigrid Methods and Applications. Springer-Verlag, 1985.

[21] W. Hackbusch and U. Trottenberg (eds.): Multigrid Methods (Proceedings, Köln-Porz 1981). *Lecture Notes in Math.* **960**, Springer-Verlag.

[22] W. Hackbusch and U. Trottenberg (eds.): Multigrid Methods II (Proceedings, Cologne 1985). *Lecture Notes in Math.* **1228**, Springer-Verlag.

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

[24] S. McCormick (ed.): *SIAM Frontiers in Applied Math*, Vol. 3 (a book on multigrid methods), to appear.

[25] S. McCormick and U. Trottenberg (eds.): Multigrid Methods. *Appl. Math. Comp.* **13** (1983), 213–474 (special issue).

[26] S. McCormick (ed.): Proceeding of the 2nd Copper Mountain Multigrid Conference. *Appl. Math. Comp.* **19** (1986), 1–372 (special issue).

[27] 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.

[28] D.J. Paddon and H. Holstein (eds.): Multigrid Methods for Integral and Differential Equations, Clarendon Press, Oxford, 1985.

[29] J. Ruge and K. Stueben: Algebraic multigrid (AMG). In [24].

[30] R.V. Southwell: Stress calculation in frameworks by the method of systematic relaxation of constraints. I, II, *Proc. Roy. Soc. London, Ser A* **151** (1935), 56–95.

[31] I.Y. Vakhutinsky, L.M. Dudkin and A.A. Ryvkin: Iterative aggregation – a new approach to the solution of large-scale problems. *Econometrica* **47** (1979), 821–841.

[32] A. Brandt: Multi-level approaches to large scale problems. Proceedings of International Congress of Mathematicians (Berkeley, California, August, 1986).

[33] V. Rokhlin: Rapid solution of integral equations of classical potential theory. *J. Comp. Phys.* **60** (1985), 187–207.

[34] J. Goodman and A.D. Sokal: Multigrid Monte-Carlo methods for lattice field theories. *Phys. Rev. Lett.* **56** (1986), 1015–1018.