Chapter 1

MULTIGRID SOLVERS AND MULTILEVEL OPTIMIZATION STRATEGIES

Achi Brandt
Department of Computer Science and Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel
Tel: +972-8-9342345, Fax: +972-8-9342945
achi@wisdom.weizmann.ac.il

Dorit Ron
Department of Computer Science and Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel
Tel: +972-8-9342141, Fax: +972-8-9342945
dron@wisdom.weizmann.ac.il

Keywords: multigrid, algebraic multigrid (AMG), bootstrap AMG, homogenization, constrained optimization, eigenproblems, Monte Carlo, renormalization multigrid, global optimization, multilevel annealing, graphs, nested revision, multilevel formulation

Introduction

An optimization problem is the task of minimizing (or maximizing — for definiteness we discuss minimization) a certain real-valued "objective functional" (or "cost", or "energy", or "performance index", etc.) $E(x)$, possibly under a set of equality and/or inequality constraints, where $x = (x_1, \dots, x_n)$ is a *vector* (often the discretization of one or several *functions*) of unknown variables (real or complex numbers, and/or integers, and/or Ising spins, etc.). A general process for solving such problems is the *point-by-point minimization*, in which one changes only one variable x_j (or few of them) at a time, lowering E as much as possible in each such step. More generally, the process accepts any candidate change of one or few variables if it causes a drop in energy ($\delta E < 0$). Even without constraints, this process would usually suffer from the following two types of difficulties:

(i) **Slow convergence**: due to the localness of the process, large-scale features (e.g., smooth components) in x are slow to converge. Acceleration by multiscale (e.g., multigrid) methods is the general cure to this trouble, since it supplements the local processing with increasingly larger scale processing, based on information suitably gathered from the fine scale. The slow-convergence primarily occurs in some neighborhood of the desired solution, where the optimization problem can often be approximated by quadratic optimization, which is equivalent to solving a linear system of equations. Therefore the multiscale methods for treating this slowness often takes the form of fast solvers for large linear systems of equations, with various extensions to nonlinear system. Such multiscale solvers have been developed for several decades now, first as solvers for discretized partial differential equations (PDEs), then expanded to general algebraic systems, mainly of *local* equations.

The first part of this article (Secs. 1–7) surveys these solvers, the emphasis being on an algebraic point of view, with only little discussion of PDE motivations and applications.

(ii) **False convergence**: In many highly nonlinear or discrete-state problems, instead of converging to the true *global* minimum of E, the point-by-point minimization process converges to the minimum of E in a certain restricted "*attraction basin*", in which the process is trapped. The basin is a set of configurations from which the *employed* process cannot proceed to configurations with lower E, although such configurations do exist. The emphasis in *global* optimization methods is the treatment of this type of trouble.

In this survey we do not attempt to fully cover the very extensive topic of global optimization. We only outline (especially in Secs. 9–11) some basic multilevel strategies to deal with highly non-quadratic optimization problems characterized by a multitude of nested attraction basins. The methods are of course partly based on approaches described in the first part, since tools for accelerating large-scale convergence are also useful for escaping large-scale attraction basins.

Content outline. The sections in this survey are loosely related to each other and can be approached independently. The following outline can help the reader locate particular topics.

1. *Unconstrained quadratic optimization and basic multiscale concepts*: Relaxation can always efficiently reduce the information content of a problem, enabling its coarsening. Interscale transfers, coarse equations (FAS, CS and EIS schemes) and multilevel cycles.

2. *Linear geometric multigrid*: PDE origin. Relaxation smoothing rates. Multigrid cycle and its efficiency: theoretical and rigorous analysis. Full multigrid (FMG). Method development, scope and needed expertise. (Local grid refinements and unbounded domains are discussed in Sec. 7, inverse PDE and eigenproblems – in Sec. 8.)

3. *Algebraic multigrid (AMG)*: Choice of coarse variables based on compatible relaxation convergence speed, implying a near-locality property. Galerkin coarsening. Interpolation rules in classical AMG and in Bootstrap AMG (BAMG). Adaptive relaxation. In-cycle iterant recombinations.

4. *Numerical homogenization: High-accuracy coarsening*: Macroscopic equations for repetitive systems. Solution methods for many similar problems, many eigenvectors or many near-zero eigenvalues.

5. *Non-symmetric and highly indefinite matrices*: Modified relaxation and coarsening. Multi-Galerkin coarsening. From wave equations to rays.

6. *Non-local equations: Dense matrices*: Exploiting underlying smoothness or asymptotic smoothness for fast matrix multiplication. Multigrid solvers based on distributive relaxation.

7. *Non-quadratic optimization: Nonlinear systems*: Advantages and disadvantages of Newton linearization for coarsening. Full Approximation Scheme (FAS) for direct nonlinear multigrid solution; Yavneh's modification. Algebraic quasilinearity. Local refinements of PDE discretizations, once-visited refinements and unbounded domains. Continuation and bifurcation. Systematic upscaling of autonomous systems.

8. *Constrained optimization and eigenproblems*: Local equality constraints: optimal control and inverse PDE. Feedback optimal control. Global and intermediate-scale constraints. Multigrid and AMG eigenproblem solvers. The Exact Interpolation Scheme (EIS) for general constraints and for eigenproblems. Many and high eigenvectors. Inequality constraints.

9. *Non-deterministic systems*: Statistical mechanics. Monte Carlo methods and their slowness. Cluster methods. Interpolation-based multiscale methods. Renormalization multigrid (RMG) – general principles. Low temperature algorithms.

10. *Global optimization: Multilevel annealing*: Inefficiency of classical simulated annealing. Coarse-level variables for highly repetitive systems. Identification of large-scale moves and their multi-scale re-shaping.

Nested revisions. Taming local excitations and multiscale population algorithms.

11. *Graph and hypergraph problems*: Applications of various optimization strategies mentioned above to representative graph/hypergraph problems: eigensolvers, weighted aggregations, post-relaxation, pre-relaxation, linearization, multilevel annealing, excitation taming, re-shaping large-scale movements and multiscale population algorithms.

12. *Multilevel formulation*: Multiscale processing can serve not only for efficient solution of problems, but also for their improved formulation when fuzzy or ill-posed. Example: image segmentation.

1. Unconstrained quadratic optimization and basic multiscale concepts

Notation. The general unconstrained quadratic optimization problem can be stated as the problem of calculating a real n-vector x for which

$$E(x) = \tfrac{1}{2}x^T Ax - b^T x \qquad \text{is minimal}\ , \tag{1.1}$$

where A is a given real symmetric positive-definite $n \times n$ matrix, and b is a given real n-vector. The minimizing vector x satisfies the linear system of equations

$$Ax = b\ . \tag{1.2}$$

Thus, our discussion next (Secs. 1–4) will focus on fast multiscale solvers for this general system (adding in Sec. 5 comments related to the cases that A is not symmetric and/or not positive-definite, which may arise in *constrained* optimization problems).

For any approximate solution $\widetilde{x}$, we denote by $e = x - \widetilde{x}$ the error vector, by $r = Ae = b - A\widetilde{x}$ the vector of residuals, and by a_{ij} the entries of A, $(i,j = 1,\dots,n)$. Thus, $r_i = b_i - \sum_{j=1}^n a_{ij}\widetilde{x}_j$ and the error satisfies the *residual equation*

$$Ae = r\ , \qquad \text{i.e.,} \quad \sum_{j=1}^{n} a_{ij}e_j = r_i\ . \tag{1.3}$$

We use $\|\cdot\|$ for the ℓ_2 norm and $\|\cdot\|_{/A}$ for the A-normalized ℓ_2 norm, i.e.,

$$\|e\|^2 = \sum_{i=1}^{n} |e_i|^2\ , \qquad \|r\|_{/A}^2 = \sum_{i=1}^{n} \frac{|r_i|^2}{\sum_{\lambda=1}^n |a_{i\lambda}|^2}\ , \tag{1.4}$$

so that for "most" (e.g., random) errors $\|Ae\|_{/A}$ is comparable to $\|e\|$, but always $\|Ae\|_{/A} \le \|e\|$.

Relaxation. A point-by-point minimization of (1.1) is equivalent to the *Gauss-Seidel* (*GS*) *relaxation* of (1.2), defined as follows. In each GS *sweep*, the n equations in (1.2) are scanned in their natural order, and each equation i in its turn is satisfied by replacing the current approximation to the associated unknown x_i with the new value $\widetilde{x}_i = a_{ii}^{-1}(b_i - \sum_{j \neq i} a_{ij}\widetilde{x}_j)$.

A common feature of this and any other type of relaxation is that at each step some corrections to $\widetilde{x}$ are calculated based on a small number of residuals. As a result, convergence must be slow when the individual residuals do not show the true magnitude of the error, i.e., when $\| r \|_{/A} \ll \| e \|$. The converse is also true (and proved in [Brandt, 1986] even for non-symmetric matrices): If the convergence of a *suitable* (e.g., GS for symmetric A) relaxation scheme is slow, then $\| r \|_{/A} = \| Ae \|_{/A} \ll \| e \|$ must hold. Since the deeper the condition $\| Ae \|_{/A} \ll \| e \|$ is satisfied the more special must be the type of the error $\| e \|$, *a suitable relaxation can always efficiently reduce the information content of the error, and quickly make it approximable by far fewer variables*. This observation is very general and holds even for quite general *nonlinear* systems.

Coarser level. Thus, following a small number of relaxation sweeps, the remaining error e, and hence also the solution x itself can be approximated by a "coarser" (or "diluted") system, i.e., a system with variables $(x_1^c, \dots, x_m^c)$, where m is a fraction of n, typically $n/8 \leq m \leq n/2$. A first issue in any coarsening scheme, whether for a linear or nonlinear system, is how to define the set x^c. In the classical case of "geometric multigrid", where the fine-level set x is defined on a well-structured grid, the coarse set x^c is naturally defined in terms of coarsening that grid, for example by omitting from it every other row and/or column (see Sec. 2). In "algebraic multigrid" x^c is typically a subset of x (see Sec. 3), and in various types of other problems (e.g., wave equations, atomistic and molecular particles models, graph problems, etc.) all kinds of other coarse sets make sense (see some examples in Secs. 5–11). The general principle is that x^c and x should be easily derived from each other: x^c is usually either a subset of x or averages of x or otherwise explicitly calculated from x, while x should be derivable from x^c by a short sequence of relaxation sweeps. This means that it is enough to find x^c in order to obtain either x itself or at least an approximation to e and hence a substantially improved approximation to x. For that purpose, however, equations for calculating x^c — the coarse-level equations — should be constructed.

Coarse equations. There are several ways to construct approximate coarse equations. An inexpensive *direct* (i.e., not interpolation-based) derivation is available in geometric multigrid, based on the PDE origin of the equations (see Sec. 2). In other cases (non PDE origin or unstructured grids) direct derivations are still possible, either by combining fine-level equations so as to nearly cancel dependence of a coarse unknown on non-coarse variables (which is suitable for linear systems; see [Brandt, 2000]), or by tabulating relations between the coarse variables x^c that correspond to the many fine-level configurations x that appear during extensive fine-level simulations (see the systematic upscaling of nonlinear systems in Sec. 7). These algebraic (non PDE) direct derivations are however too expensive, except in the case of autonomous systems (see Sec. 7).

Relatively inexpensive algebraic derivations of coarse equations are obtained via the derivation of an explicit coarse-to-fine *interpolation operator* $\widetilde{\uparrow}_c^f$ such that $x \approx \widetilde{\uparrow}_c^f x^c$. The coarse equations are then simply the equations for minimizing $E(\widetilde{\uparrow}_c^f x^c)$. In the quadratic minimization problem (1.1), if $\widetilde{\uparrow}_c^f$ is linear or affine, then this coarse minimization problem is again quadratic, yielding again a linear system of equations.

Interpolation-based schemes. A basic rule that the interpolation operator must satisfy is *stationarity*, meaning that the current approximation $\widetilde{x}$ (the approximation to the minimizing x *just before switching to the coarse level*) must satisfy $\widetilde{x} = \widetilde{\uparrow}_c^f \widetilde{x}^c$. This ensures that if $\widetilde{x}$ is already at the desired minimum of $E(x)$, the algorithm will never move away from it: $x^c = \widetilde{x}^c$ is the solution of the coarse equations, since it minimizes $E(\widetilde{\uparrow}_c^f x^c)$.

The classical multigrid (including algebraic multigrid) way to ensure stationarity is to define

$$\widetilde{\uparrow}_c^f x^c = \uparrow_c^f (x^c - \widetilde{x}^c) + \widetilde{x} \tag{1.5}$$

which yields the so-called *Full Approximation Scheme* (*FAS*; see Sec. 7). Here $\uparrow_c^f$ is a linear interpolation operator (i.e., an $n \times m$ matrix), independent of the current approximation $\widetilde{x}$ or its coarse representation $\widetilde{x}^c$, designed once for all so that *any* error e^c left after several relaxation sweeps satisfies $e \approx \uparrow_c^f e^c$. In fact, instead of x^c, the variable $e^c = x^c - \widetilde{x}^c$, which approximates the sought correction (i.e., the error) e, can be employed at the coarse level, thus yielding the so-called *Correction Scheme (CS)*, which is widely used for linear problems (as in Secs. 2–6 below). In the recent *Exact Interpolation Scheme* (*EIS*, appearing in Sec. 8) $\widetilde{\uparrow}_c^f$ is itself *linear* ($n \times m$ matrix), and it is updated each time the algorithm switches to the coarse grid, so as to satisfy $\widetilde{x} = \widetilde{\uparrow}_c^f \widetilde{x}^c$ with the current approximation $\widetilde{x}$.

A more complex, *multi-interpolation scheme* is needed in some cases, such as highly indefinite problems: see Sec. 5.

Multilevel cycles. Having constructed the coarse-level equations, they are then (approximately) solved by a similar procedure: a small number of relaxation sweeps followed by approximating the remaining error with a still coarser system. This recursively defines the multilevel cycle, which, for a work comparable to that of just few relaxation sweeps over the finest level (the given system), would usually reduce the error to a small fraction (far less than .5, typically) of its pre-cycle size.

2. Linear geometric multigrid

Multilevel solvers were first developed in the form of multigrid algorithms for solving linear partial differential equations discretized on regular grids. Their main features are surveyed below. For an extended elementary acquaintance, see for example the early chapters in [Briggs *et al.*, 2000], [Trottenberg *et al.*, 2000] or in [Venner and Lubrecht, 2000].

When the matrix equation (1.2) is a discretization of a differential system $Lu = f$ on some grid, the condition $\| Ae \|_{/A} \ll \| e \|$, which is necessarily obtained when relaxation slows down, can be interpreted as saying that the error e approximates a continuous function v satisfying $\| Lv \| \ll \| L \| \cdot \| v \|$ in some corresponding norms. In case L is a uniformly elliptic operator, this implies that v, and similarly e, is a smooth function. In case L is not elliptic, a certain smoothness along special lines, called *characteristics*, is at least implied. Hence, *relaxation efficiently reduces non-smooth components, thus making the remaining error smooth and hence approximable on a coarse grid.*

Smoothing factor. A precise measure for the efficiency of smoothing by a given relaxation scheme can be calculated in the common case that the discretization grid is uniform and the relaxation sweep scans its points in some consistent order. Ignoring boundaries and variations in the coefficients of equations, a simple calculation yields in this case the convergence factor per sweep of each *Fourier component* of the error (see, e.g., [Brandt, 1977] or [Trottenberg *et al.*, 2000]). The *smoothing factor* $\overline{\mu}$ is then defined as the largest (i.e., the worst) convergence factor per sweep among all those components which are too oscillatory to be visible on the coarse grid (hence must be reduced by the relaxation process). For example, for the Poisson equation in two dimensions $\partial^2 u/\partial x^2 + \partial^2 u/\partial y^2 = f$, discretized by the standard five-point second-order discretization, and for the standard coarsening (1:2 meshsize ratio), it turns out that $\overline{\mu} = .5$ if the GS relaxation scans the gridpoints in "lexicographic" (row by row) ordering, and $\overline{\mu} = .25$ if they are scanned in red/black (checkerboard) order. Hence, two red/black GS sweeps reduce all error components invisible to the coarse grid by a factor $.25^2 = .0625$.

Correction Scheme (CS). To approximate the smooth error $e = x - \widetilde{x}$ left after several relaxation sweeps, its equation $Ae = r$ is approximated on the coarse level by an equation of the form

$$A^c e^c = r^c \ . \tag{1.6}$$

If (1.2) is a discretization on a grid with meshsize h of a certain differential problem, one can define A^c as the matrix obtained in a similar discretization of the same problem on a grid with a larger meshsize H. The standard, and usually the most efficient, meshsize ratio is $h : H = 1 : 2$. The right-hand side of (1.6) is defined by $r^c = \uparrow_f^c r$, where $\uparrow_f^c$ is a fine-to-coarse transfer operator, called *residual weighting* or *restriction*. That is, $\uparrow_f^c r$ is a coarse-grid function whose value at each point is (typically) a certain weighted average of values of r at neighboring fine-grid points. (A^c and r^c can also be defined by the Galerkin procedure described in Sec. 3 below).

Having obtained (in a way discussed later) an approximate solution $\widetilde{e}^c$ to Eq. (1.6), we then use it as a correction to the fine-grid solution. Namely, we replace

$$\widetilde{x} \longleftarrow \widetilde{x} + \uparrow_c^f \widetilde{e}^c \ , \tag{1.7}$$

where $\uparrow_c^f$ is a coarse-to-fine *interpolation* matrix (also called *prolongation*). That is, at each fine-grid point the value of $\uparrow_c^f \widetilde{e}^c$, designed to approximate e, is interpolated from values of $\widetilde{e}^c$ at neighboring coarse-grid points. Linear interpolation can be used in most cases. (General rules concerning the orders of $\uparrow_c^f$ and $\uparrow_f^c$ are given in [Brandt, 1982, §4.3] and [Brandt, 1994].)

The whole process of calculating $r^c = \uparrow_f^c r$, solving (1.6) and interpolating the correction (1.7) is called a *coarse-grid correction*.

Multigrid cycle. To efficiently get an approximate solution to the coarse-grid equation (1.6), we employ the above solution process recursively; i.e. (1.6) is itself solved by relaxation sweeps combined with still-coarser-grid corrections. Thus *one multigrid cycle for improving a given approximate solution $\overline{x}$ to the system $Ax = b$* is defined recursively as the following 6 steps.

1 If the given grid includes a small number of points — solve the equations directly, e.g., by Gaussian elimination.

2 *Pre-relaxation*: Perform ν_1 relaxation sweeps on (1.2) starting with the *initial* $\overline{x}$, resulting in a *new* approximate solution $\widetilde{x}$.

3 *Restriction*: Calculate $r^c = \uparrow_f^c (b - A\widetilde{x})$.

4 *Recursion*: Starting with $\widetilde{e}^c = 0$, make γ successive multigrid cycles for improving $\widetilde{e}^c$ as an approximate solution to the system (1.6).

5 *Correction*: $\widehat{x} = \widetilde{x} + \uparrow_c^f \widetilde{e}^c$.

6 *Post-relaxation*: Perform ν_2 additional relaxation sweeps on (1.2), starting with $\widehat{x}$ and yielding the *final* $\overline{x}$ of the cycle.

The sweep counts ν_1 and ν_2 are typically 0, 1 or 2, with $\nu_1 + \nu_2$ being 2, 3 or 4. The *cycle index* γ is usually 1 or 2 (called V or W cycle, respectively, due to the shape of its flowchart; see for example Fig. 1.1).

Cycle efficiency. The $\nu = \nu_1 + \nu_2$ sweeps performed in a V cycle on any of its grids are expected to reduce the corresponding error components (those visible on that grid but not on the coarser ones) by the factor $\overline{\mu}^\nu$, where $\overline{\mu}$ is the smoothing factor. Since all grids are so traversed, *the cycle should heuristically reduce all error components at least by the factor* $\overline{\mu}^\nu$. The work invested per such cycle is only about $(1-2^{-d})^{-1}(\nu+1)$ work units, where a work unit is the work of one relaxation sweep on the finest level, $\nu + 1$ work units is therefore roughly the total work at the finest level (relaxation plus residual restriction), and d is the dimension, so the number of gridpoints (and hence the work) is reduced by 2^{-d} per coarsening level. Experience and theory show that for regular scalar elliptic problems and small enough ν this efficiency is indeed attained, provided the boundary conditions are properly relaxed and correct inter-grid transfers $\uparrow_c^f$ and $\uparrow_f^c$ are used. Thus $\overline{\mu}$ can serve as an excellent practical predictor of the multigrid performance one *should be able* to obtain.

For example, for the 2D Poisson problem mentioned above, each V cycle with 2 red/black Gauss-Seidel sweeps on each level, costing only about 20 computer additions per unknown, reduces the error by more than an order of magnitude.

Theoretical and rigorous analysis. There is a vast literature of rigorous studies of the multigrid cycle; see for example the classical book ([Hackbusch, 1985]) and many papers appearing in the proceedings of the ten (so far) Copper-Mountain and six (so far) European conferences on multigrid methods. These studies rigorously prove that the convergence factor per V or W cycle, under various assumptions concerning the differential equation and its discretization, is *meshsize independent*: no matter how fine is the discretization, the factor is smaller than some constant C smaller than 1. Most often C is not numerically specified, or the proved value of the convergence rate $\log\frac{1}{C}$ is very far from reflecting the orders-of-magnitude larger rate observed in practice. Also, the rigorous proofs apply only to relatively simple problems and synthetic algorithms (often different from the best real algorithms).

The only quantitatively realistic (in fact quite precise) theoretical predictors are those based on localized Fourier analysis, often called *local mode analysis* (LMA). The easiest and most practical LMA predictor is the smoothing factor described above. A more elaborate predictor is obtained by a similar Fourier analysis of a several-level (most often two-level) multigrid cycle, thus analyzing both the relaxation and the inter-grid transfers. (See detailed results and software for calculating such convergence factors in [Weinands, 2001].) Although the employed Fourier analysis is rigorously valid only for equations with constant coefficients in an infinite or rectangular domains, in practice the predictions hold in a much wider class of problems, so they are routinely used in algorithm development and program debugging, even for complicated nonlinear systems.

Moreover, for general linear elliptic PDE systems with piecewise smooth coefficients in general domains discretized by uniform grids, it has been rigorously proved in [Brandt, 1991a] and [Brandt, 1994] that, in the limit of small meshsizes, the quantitatively sharp convergence factors predicted by LMA are indeed obtained, provided the multigrid cycle is supplemented with a proper processing at and near the boundaries. That processing, it is proved, costs negligible extra computer work. Apart from mode analysis, a Coarse Grid Approximation condition has been introduced in [Brandt, 1991a] and [Brandt, 1994] which is both necessary and sufficient for the multigrid algorithm to work properly. Various error norms and their relations to the orders of the inter-grid transfer operators are analyzed. Global mode analysis, required to supplement the local analysis in various border cases, is developed, and practical implications of the analysis, including practical ways for constructing and debugging multigrid solvers, are generally reviewed. A major emphasis is on the importance and practicality of adding partial (local) relaxation passes to the multigrid algorithm (cf. [Brandt, 1977, App. A.9]): Theory and practice show that multigrid efficiency is greatly enhanced by adding special relaxation steps at any local neighborhood exhibiting unusually large residuals (cf. the *adaptive relaxation rule* in Sec. 3).

Full Multigrid (FMG) algorithm. The multigrid cycles described above can be applied to any first approximation given on the finest grid. In a full multigrid (FMG) algorithm, the first approximation is obtained by interpolation from a solution on the next coarser grid, which has previously been calculated by a similar FMG algorithm. With such a first approximation, and provided that the interpolation correctly corresponds to the error norm used, one multigrid cycle should suffice to solve the fine-grid equations to the level of discretization errors. A typical FMG algorithm, with one V cycle per refinement, is shown in Fig. 1.1. High-order discretization accuracy may require another cycle or two per level.

The FMG algorithms are less sensitive than the multigrid cycles. That is, in many irregular cases the asymptotic convergence factor of the cycles is not good, but the FMG algorithm with one cycle per refinement still guarantees solution to the level of discretization errors. This is because unusually slow-to-converge components tend also to have unusually large discretization errors. In any case, the apriori guarantee itself is not really needed: From differences between the final solutions at different meshsizes (e.g., differences between the solutions at the doubly-circled stages in Fig. 1.1), one can directly calculate the rate of convergence to the *differential* solution, which is all that really matters.

"Textbook" multigrid efficiency (TME). A fully successful FMG algorithm, employing one V or W cycle per refinement level and a couple of strongly smoothing relaxation sweeps at each level, costs less than ten "minimal work units" and can solve the discretized PDE to at least second-order accuracy (error $O(h^2)$ on a given grid with meshsize h), the "minimal work unit" being defined as the amount of operations involved in expressing the simplest discretization (or performing the simplest relaxation sweep) on the given grid. This ideal efficiency has been termed "*textbook multigrid efficiency*" (TME).

Note in addition that all the FMG processes, except at the coarsest levels (whose computational work is negligible), can be executed by very many processors *in parallel*.

Method development. The first *two*-grid method has probably been [Southwell, 1935], followed by several others, and the first *multi*-grid solver was described in [Fedorenko, 1964], but only for rectangular domains and with extremely poor performance estimates (see the historical notes in [Brandt, 1977]). The first multigrid algorithms of modern (near TME) efficiency and generality appeared in [Brandt, 1973], including attempts to extend them to fluid dynamics problems. The textbook efficiency was first obtained only for scalar, linear uniformly elliptic

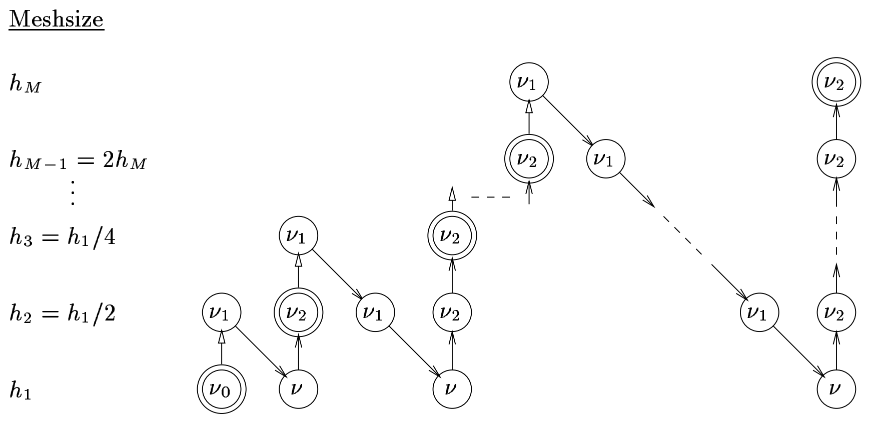

Figure 1.1. FMG algorithm with one V cycle per level.

⇧	is the solution interpolation to a new grid.
↑	is the interpolation of corrections.
↘	is the fine-to-coarse transfer of residuals.
ν_i	stands for ν_i relaxation sweeps. On the coarsest grid $\nu = \nu_1 + \nu_2$ and somewhat larger ν_0 are usually used, or the equations are solved directly.
◎	shows the stage in the algorithm where the final solution is obtained for the corresponding meshsize.

h_M is the finest level and h_1 is the coarsest level.

problems, but has since then been extended to nonlinear, anisotropic, non-symmetric, non-elliptic, highly indefinite (waves – see Sec. 5) and non-scalar PDE systems, in complex geometries and with local grid refinements (described in Sec. 7 below), with various discontinuities, singularities, boundary layers, shocks, etc. Each of these features/difficulties has required further conceptual development of the method. A table summarizing many of these concepts, together with bibliographic pointers, is given in [Brandt, 1998].

The multigrid methods surveyed above are sometimes called "geometric multigrid", since they are defined in terms of well structured grids. A serious disadvantage of these methods is the demanding amount of expertise and development effort needed to achieve ideal performance for real-world problems. A practical, more general (working also for unstructured grids), somewhat less efficient but still high-performance alternative is described in the next section.

3. Algebraic multigrid (AMG)

Algebraic multigrid (AMG) algorithms are solvers of linear systems of equations which are based on multigrid principles but do not explicitly use the geometry of grids. Introduced in [Brandt *et al.*, 1982], the emphasis in AMG is on automatic procedures for *coarsening* the set of equations, relying exclusively on its algebraic relations. AMG is widely employed for solving discretized partial differential equations (PDEs) on unstructured grids, or even on structured grids when the coarse grid can no longer be structured, or when the PDE has highly disordered coefficients. AMG can also be used (as in [Brandt *et al.*, 1982]) for many types of discrete systems *not* arising from differential equations.

As described above (Sec. 1), following a small number of relaxation sweeps, the error e in solving (1.2) will satisfy $\| Ae \|_{/A} \ll \| e \|$, and can therefore somehow be approximated by interpolation from a "coarser" vector e^c:

$$e \approx \uparrow_c^f e^c \ . \tag{1.8}$$

In the general algebraic setting, the choice of coarse variables that describe e^c, the derivations of coarse equation (1.6) that approximates the fine equation (1.3) and the interpolation operator $\uparrow_c^f$ are all determined automatically, using the following AMG procedures.

The set C of coarse variables is chosen as a *subset* of the set of fine (original) variables; or, more generally, each coarse variable is chosen to be a linear combination of a small number of fine variables. (Even more generally, each coarse variable can be defined as a linear combination of several fine *ghost* variables, where the fine variables themselves are also linear combinations of those ghost variables; see [Brandt, 2000, App. A]). Thus, in any case, for each fine level vector x (or e) there corresponds a unique coarse-level vector x^c (respectively e^c). In classical AMG (see [Brandt *et al.*, 1982], [Brandt *et al.*, 1984], [Brandt, 1986], [Ruge and Stüben, 1987], [Stüben, 2000]), the set C is chosen so that each fine variable is "strongly coupled" to C through one or several equations (cf. the definition of coupling $a_{ij}^{(\nu)}$ in (1.13) below). More generally, a criterion for gauging, and a practical method to control, the quality of this set can be based on sweeps of *compatible relaxation*. This is a modified fine-level relaxation scheme that keeps the coarse-level variables invariant (i.e., it keeps the fine-level configuration always compatible with the same coarse-level configuration: x changes but so that x^c does not change). For example, if x^c is defined to be a subset of x, compatible GS relaxation simply avoids relaxing those variables of x that belong to x^c. *The set C is guaranteed to be good when* (*and only to the extent* *that) the compatible relaxation exhibits uniformly fast convergence rates.* Where these rates are too slow, a diluted subset of the slow-to-converge variables should be added to C (or, alternatively, subsets of the slow to converge variables should be relaxed *simultaneously*; see [Brandt, 2000]). Following this modification of C (and/or the relaxation) the convergence factor of compatible relaxation should be checked again. Generally, that factor is a good predictor to the potential AMG efficiency, similar to the *smoothing factor* in the geometric multigrid (see Sec. 2 above).

The derivation of the coarse-level equations is described below for systems of *local* equations, i.e., systems $Ax = b$ whose variables $(x_1, x_2, \ldots)$ each has a location in a low-dimensional space, and whose equations each involves only few, neighboring variables in that space. Generalizations exist to "sparsely positive definite" matrices, including positive-type matrices (see [Brandt, 1986]), to "asymptotically smooth" and "asymptotically smoothly oscillatory" matrices, including electrostatic or gravimetric interactions (see Sec. 6 below), and to some other types of systems. Also, the same procedures most often work reasonably well even for cases not belonging to any of these types.

The fast convergence of the compatible relaxation implies that the values of the coarse set of variables indeed determine, up to fast local processing, the values of the fine set. Moreover, it implies that the chosen coarse set satisfies a property called "*near locality*": the fine level solution at each point can be calculated from the coarse level *locally*, given just its coarse *neighborhood*, with very weak remnant dependence on coarse values outside that neighborhood: the remnant dependence decays *exponentially* (or even faster) as a function of the neighborhood radius. For 2D discrete Poisson equations, for example, the remnant dependence tends (after enough coarsening levels) to $\exp(-\pi r^2/2)$, where r is the neighborhood radius measured in meshsizes of the coarse level ([Schröder *et al.*, 1976], [Zimare, 1980]). Since each coarse variable is defined locally by few fine variables, it too depends only *nearly-locally* on all other coarse variables. Hence, an *equation* for each coarse variable in terms of other coarse variables can be derived locally, using only a *local set* of fine-level equations. The error in that coarse equation will decrease exponentially as a function of the size of that local set.

Galerkin coarsening. Suppose the interpolation $\uparrow_c^f$ has been chosen to be a certain matrix P; i.e.,

$$(\uparrow_c^f e^c)_i = \sum_{j=1}^{n_i} P_{ij} e^c_{I_{i,j}} \ , \qquad (1.9)$$

where the sequence $\{I_{i,j}\}_{j=1}^{n_i}$ is an ordered set of the indices of the n_i coarse-level variables from which interpolation to the i-th fine-level variable is made. (They are chosen in the "neighborhood" of x_i, defined either geometrically or in terms of algebraic couplings.) Then, for linear systems (1.2) resulting from (1.1), a natural way to define the coarse approximation e^c to the error $e = x - \tilde{x}$ is to require that e^c minimizes (1.1) over all possible substitutions (1.8), i.e., e^c minimizes the quadratic form

$$\tfrac{1}{2}(\tilde{x} + Pe^c)^T A(\tilde{x} + Pe^c) - b^T(\tilde{x} + Pe^c) \ . \tag{1.10}$$

It is easy to see that e^c minimizes (1.10) if and only if it satisfies (1.6) with

$$A^c = P^T AP \tag{1.11}$$

and

$$r^c = P^T(b - A\tilde{x}) = P^T r \ . \tag{1.12}$$

The coarsening of (1.3) by (1.6), (1.11) and (1.12) (as well as the generalization (1.18)-(1.19) to the non-symmetric case) is called *Galerkin coarsening*. Using this prescription, the only part that remains to be automated is the choice of the interpolation matrix P.

Two basic approaches for choosing the interpolation operator, the "classical AMG" and the "Bootstrap AMG", will be described below. Other approaches include the smoothed-aggregation variant of classical AMG introduced in [Vanek *et al.*, 1994], [Vanek *et al.*, 1996] and [Vanek *et al.*, 1998]; local eigenvector methods like AMGe [Brezina *et al.*] and approaches based on local energy minimization [Mandel *et al.*, 1998] and [Wan *et al.*, 1998] and on the Euclidean norm instead of the energy norm [Brandt and Galun, 1996]. For completeness, see [Stüben, 2000] and references therein.

Classical AMG. In classical AMG, the interpolation weights are based on the "strength of couplings". The following are general formulae for such couplings. (To understand, please follow first the case with zero thresholds: $\alpha_i^{(\nu)} = 0$ in which case $a_i^{(\nu)} = 1$; also start with the case $\nu = 1$. Note that for a zero-sum matrix, i.e., $\sum_j a_{ij} = 0$ for every i, as in many applications, the normalized couplings satisfy $\sum_j a_{ij}^{(\nu)} = 1$ and therefore also $\sum_{j \in N_i^{(\nu)}} P_{ij} = 1$ for every i.) The *ν-generation normalized* *couplings* $a_{ij}^{(\nu)}$ are recursively defined by

$$a_{ij}^{(\nu)} = \begin{cases} 1 & \text{for} \quad i \in C\ ,\ j = i \\ 0 & \text{for} \quad i \in C\ ,\ j \neq i \\ 0 & \text{for} \quad i \notin C\ ,\ j = i \\ \widetilde{a}_{ij}^{(\nu)}/a_i^{(\nu)} & \text{for} \quad i \notin C\ ,\ j \neq i\ ,\ \widetilde{a}_{ij}^{(\nu)} \geq \alpha_i^{(\nu)} \\ 0 & \text{for} \quad i \notin C\ ,\ j \neq i\ ,\ \widetilde{a}_{ij}^{(\nu)} < \alpha_i^{(\nu)} \end{cases} \tag{1.13}$$

where

$$\widetilde{a}_{ij}^{(\nu)} = \begin{cases} -a_{ij}/a_{ii} & : \nu = 1 \\ \sum_l a_{il}^{(\nu-1)} a_{lj}^{(\nu-1)} / (1 - \sum_l a_{il}^{(\nu-1)} a_{li}^{(\nu-1)}) & : \nu > 1 \end{cases} \tag{1.14}$$

$$a_i^{(\nu)} = \sum_{k \neq i} \widetilde{a}_{ik}^{(\nu)} / \sum_{k \neq i \text{ s.t. } \widetilde{a}_{ik}^{(\nu)} \geq \alpha_i^{(\nu)}} \widetilde{a}_{ik}^{(\nu)} \tag{1.15}$$

and $\alpha_i^{(\nu)} \geq 0$ are thresholds chosen to control complexity. Defining for each $i \notin C$ a *coarse neighborhood* $N_i^\nu = \{j \in C\ ,\ a_{ij}^{(\nu)} > 0\}$, the classical AMG interpolation matrix P can be defined by

$$P_{ij} = a_{ij}^{(\nu)} \sum_{k \neq i} a_{ik}^{(\nu)} / \sum_{k \in N_i^\nu} a_{ik}^{(\nu)} \quad \text{for} \quad j \in N_i^\nu \tag{1.16}$$

and $P_{ij} = 0$ if $j \notin N_i^\nu$; most often $\nu = 1$ or $\nu = 2$ is used. Such interpolations are reasonable for a matrix A which is close to an M *matrix* (a matrix whose off-diagonal terms are non-positive, while its row sums are non-negative). Even for such matrices the interpolation (1.16) is sensitive to scaling and sometimes not accurate enough (especially for the high-accuracy purposes described in Sec. 4 below), unless the generation number ν is raised and the thresholds $\alpha_i^{(\nu)}$ are lowered so much that the neighborhood N_i^ν become inefficiently large.

Moreover, the classical AMG interpolation is restricted to *subset coarsening*, where the coarse set of variables is simply a subset of the fine set, whereas in many cases the desired coarsening is in term of *averages* (see item (i) in Sec. 4).

Bootstrap AMG (BAMG) Aiming at a greater generality and solver efficiency, the BAMG approach is based on *iterative* derivation of the interpolation operator P. The basic requirement from that operator is that it should interpolate well low-residual errors. For example, Theorem 4.1 in [Brandt, 1986] implies that for symmetric positive definite systems relaxed by Gauss-Seidel relaxation, $\| e - Pe^c \|^2$ should not be large compared with the normalized residual norm $\| Ae \|_{/A}$. The BAMG iterations produce such low-residual vectors and adjust P to fit them, applying the evolving solver itself for obtaining vectors with progressively lower residuals.

A set of some K low-residual vectors $\{x^{(k)}\}_{k=1}^{K}$ can first be obtained by relaxation. Namely, each $x^{(k)}$ is a result of several fine-level relaxation sweeps on the homogeneous equation $Ax = 0$, starting either from a *random* approximation or, in case the geometric location is known of each variable x_i, from a *smooth* function. (Those smooth functions can often easily be chosen to fit the homogeneous boundary conditions; for different k they should be designed to be sufficiently different from each other). A first approximation to the set of interpolation coefficients $\{P_{ij}\}_j$ for each $i \notin C$ can then be determined so that it satisfies best, in a *weighted least-square* sense, the over-determined set of equations

$$x_i^{(k)} = \sum_{j \in N_i} P_{ij} x_j^{(k)c} \ , \qquad (k = 1, \ldots, K) \tag{1.17}$$

where $x^{(k)c}$ is the coarse vector corresponding to $x^{(k)}$ and N_i is the coarse neighborhood of i (such as N_i^1 or N_i^2 defined above). The weight of the k equation is chosen, e.g., proportional to $\| Ax^{(k)} \|_{/A}$. This least-square procedure can also detect when the neighborhood N_i can be reduced or should be enlarged or modified; it is thus possible to keep its size $|N_i|$ at a minimum. The number K of relaxed vectors should be larger than, or at least equal to any $|N_i|$.

Having constructed in this way the first approximation to P, one can then build the first approximation to the coarse-level matrix $A^c = P^T AP$, which can then be used in a similar way to obtain a first approximation for the next, still-coarser-level set of unknowns, interpolation and matrix. And so on.

Once several coarse levels have been so defined, they can be used to obtain a much better approximation to P. This is defined similarly to the first approximation described above, but instead of the *relaxed* vectors $x^{(k)}$, one obtains each of these vectors by a short *multilevel cycle* applied to the homogeneous equation $Ax = 0$, followed by normalizing $x^{(k)}$ to obtain $\| x^{(k)} \| = 1$. If after the normalization the residuals $\{Ax^{(k)}\}$ are not reduced much from their pre-cycle values, then instead of using the homogeneous equation, the cycles should be applied to the eigenproblem $Ax^{(k)} = \lambda_k x^{(k)}$, with orthonormalization and eigenvalue calculation at the coarsest level, using the Exact Interpolation Scheme (EIS; see Sec. 8). That scheme is particularly appropriate here, since it includes re-fitting of the interpolation operators, which is anyway the main target here. Also, with EIS the test vectors $\{x^{(k)}\}$ for all levels correspond to each other: for example, $\{x^{(k)c}\}$ are the appropriate test vectors for the first coarse level.

With the improved approximation to P and to $A^c = P^TAP$, and likewise at the coarser levels, one can use them to similarly obtain such matrices for additional, still coarser, levels. Then one obtains still better approximations by repeating these procedures once more, now with more levels and much better accuracy.

At each coarse level one should check whether any of its variables represent an *almost disjoint subsystem* (ADS). An ADS is a subset of fine-level variables whose couplings with variables external to the subset are much weaker than its internal couplings. Such a subset can be detected at that coarser level where it will be represented by a single variable. For each ADS, a *separate* set of test vectors needs to be constructed according to the above procedures.

An important expected advantage of the BAMG algorithm is that it keeps all $|N_i|$ as small as possible while still yielding highly accurate interpolations, hence producing A^c almost as sparse as possible, saving much work in its calculation, and also in the actual operation of the multigrid solver. The latter is often the most important consideration, as the solver is re-used many times. In many applications, the re-usage of the solver would be needed upon some small changes in the problem. The BAMG re-derivation of P and A^c would then usually be very inexpensive, e.g., requiring at each level at most one additional iteration, and only in regions around the introduced changes.

Unlike classical AMG, BAMG can obtain *arbitrarily high interpolation accuracy*, which is important for solving high-order differential systems and for many other tasks (see Sec. 4). Moreover, BAMG-type methods can produce accurate interpolation even for *aggressive coarsening* (fast decrease in the number of degrees of freedom upon each coarsening step), potentially important to problems whose complexity would otherwise increase with coarsening.

Additional AMG tips. Several general AMG rules are worth at least brief mentioning here (for more details see [Brandt, 2000, §12 and App. A]).

Relaxation schemes. Classical AMG uses always the Gauss-Seidel (GS) relaxation scheme. For general systems, including those arising in discretizing non-scalar PDE systems, GS is not always adequate. Relaxation schemes that can always be used are Kacmarz and least-squares (see Sec. 5), although they are usually much less effective than GS for systems where the latter is applicable (systems nearly symmetric, nearly definite). Other schemes often used are *distributive schemes* (relaxing an equation by distributing changes to several unknowns, as in Sec. 6 below, and in many discretized PDE, where the distribution pattern can be designed at the differential level) and *block relaxation* (relaxing simultaneously several equations, which can sometimes serve as an alternative for increasing the number of coarse variables).

Adaptive Relaxation Rule: Keep adding relaxation passes at any local neighborhood where the *normalized* residuals $|r_i|/\sum_j |a_{ij}|$ are much larger in magnitude than their larger-scale average magnitude. Such relaxation adaptation is useful in inexpensively and automatically eliminating mutigrid slowness caused by various local singularities, such as boundaries (especially with re-entrant corners), strong shocks, source singularities, etc. (See [Brandt, 1977, App. A.9], [Bai and Brandt, 1987] and theoretical background in [Brandt, 1991a], [Brandt, 1994].)

Recombination of iterants. A general way to eliminate m error components that are particularly slow to converge (such as AZMs; see (iv) in Sec. 4) is to recombine $m + 1$ iterants (each being, for example, the approximate solution obtained after another multilevel cycle) so as to minimize the l_2 residual norm; see [Brandt and Mikulinsky, 1995]. An efficient (and popular) way to organize such iterant recombinations is to regard the multigrid cycle as a preconditioner in a conjugate-gradient or GMRES procedure. However, note that in addition such recombinations may be very useful at *coarse-level sub-cycles*. Such coarse-level recombinations should converge those components that are well approximated on the first coarse level, but not on still coarser levels. Also, using the FAS scheme, coarse-level recombinations can replace fine-level recombinations, saving dramatically on the amount of storage needed to store previous iterants. See examples in [Washio and Oosterlee, 1997].

4. Numerical homogenization: High-accuracy coarsening

Due to the *near locality* property mentioned above, highly accurate coarse equations exist, and a relatively inexpensive method to derive them is offered by BAMG. Other methods to achieve highly accurate coarse equations not through the Galerkin formulation are described and demonstrated in [Brandt, 2000] (including a method due to Irad Yavneh). These other methods are much more expensive, but can be afforded for highly-repetitive systems and they belong to a class of coarsening methods applicable in non-linear and non-deterministic problems, which are often indeed repetitive (see Secs. 7 and 9 below).

For the purpose of multi-level (multigrid) *cycles*, a rather low (but uniform) coarsening accuracy would usually suffice. For example, a coarse grid equation with at most 10% error for all "smooth" components (i.e., all those slow to converge in relaxation) can yield a multilevel cycle with a convergence factor close to 0.1. By performing successively any number of such cycles, any desired solution accuracy can rapidly be obtained. This will usually be far more cost effective than deriving higher accuracy coarsening.

In many other cases, however, higher degrees of coarsening accuracy are really needed. Examples:

(i) **Once-for-all coarsening**, for the purpose of deriving coarse-level equations not just for the error e remaining after relaxation, but for the full solution x, i.e., a coarse approximation to the original equation (1.2). In this case the most appropriate coarse-level variables are such that they are indeed largely insensitive to relaxation, each one usually being a certain weighted *average* of fine-level variables. Note that in this case classical AMG cannot be used for deriving the interpolation and BAMG-type derivations must enter; this generalizes the role of relaxation, from mere error smoothing to interpolation determination.

(ii) **Homogenization**. In particular if the original differential equations are *repetitive* (the same equations with the same coefficients repeat themselves in many different subdomains), fine grids (resolving various features of these equations) need be used only around *one* subdomain, and only (very) coarse equations (derived by, e.g., BAMG at that subdomain) need be used elsewhere. This will allow derivation of *macroscopic equations* for microscopically complicated materials or processes.

(iii) **Repeat solve**. Many industrial and scientific problems are solved many times over and over again, with only small changes between two successive solves. In such a situation, the finer levels need not be used everywhere at each solve: each level is needed only around the region where the equations have changed. This will allow, for example, very fast calculations and updates of *matrix inverses and determinants* (see [Brandt, 2001, §12]).

(iv) **Problems with almost-zero modes (AZMs)**, i.e., eigenvectors with unusually close to zero eigenvalues. Such modes often reflect some ill defined global moves, such as rigid-body motions of the entire system in problems of elasticity, or a gliding motion of two rigid bodies along their contact surface. Such AZMs also plague various disordered problems, such as Dirac equations on critical gauge fields (cf. [Brandt, 2001, §11]). For problems with *many* AZMs, a general cure is to increase the coarsening accuracy. A small number of AZMs (such as those associated with global rigid body motions) may still exhibit slow convergence even at higher accuracies, but they can be eliminated by recombining iterants (see end of Sec. 3).

(v) **Collective calculation of many eigenvectors of a matrix** requires high-accuracy interpolations (see Sec. 8).

(vi) **Cases of massively parallel processing** with poor inter-processor communications may benefit from replacing multigrid cycles by once-for-all solution coarsening.

5. Non-symmetric and highly indefinite matrices

The multigrid and algebraic multigrid solvers described above for symmetric and definite matrices (unconstrained quadratic minimization) can be extended in many ways. Some important types of generalization are briefly reviewed in this section and in Secs. 6–7 below.

Non-symmetric matrices. If A in (1.2) is not symmetric, the Galerkin coarsening (1.11)-(1.12) should be generalized to

$$A^c = QAP \tag{1.18}$$

and

$$r^c = Q(b - A\widetilde{x}) = Qr \ , \tag{1.19}$$

where the matrix Q is a *fine-to-coarse transfer operator*, also called *restriction*. The automatic methods (either classical AMG or BAMG) for deriving the interpolation matrix P from the system matrix A can still be used here. In a fully analogous way, the transposed restriction Q^T (or the transposed conjugate restriction $Q^\dagger$, in the complex case) can be derived (e.g., at each BAMG stage) from A^T (respectively $A^\dagger$). This choice of Q can be motivated by the requirement that high-residual vectors, which are efficiently reduced by the relaxation process, should not be amplified by the coarse-grid correction.

GS relaxation can still be used in non-symmetric cases with dominant diagonal. Other cases can be relaxed by the schemes mentioned next for indefinite systems.

Highly indefinite problems. For highly indefinite problems, the multigrid (including algebraic multigrid) methods presented above will not work. One simple reason is that the GS relaxation is fast-diverging when (1.2) is highly indefinite, so it cannot be used even as a smoother. Instead, however, one can use a variety of other schemes, such as the least-square relaxation (equivalent to GS for the definite system $A^TAx = A^Tb$) or the Kacmarz relaxation (equivalent to GS for the system $AA^Ty = b$, with $x = A^Ty$ describing how to change x upon each relaxation step [Tanabe, 1971]).

A more serious difficulty with highly indefinite systems is that the low-residual components (slow to converge upon relaxation) cannot all be described by a single relation like (1.8) (unless the interpolation-point numbers $|N_i|$ increase unboundedly at increasingly coarser levels, causing unbounded complexity). Some *definite* systems exhibit similar traits; e.g., definite systems with indefinite factors, such as $A = B^T B$, where B is highly indefinite.

In this situation (1.8) has to be generalized to the form

$$e_i = \sum_{j=1}^{J} \varphi_i^{(j)} w_i^{(j)} \ , \qquad \varphi^{(j)} \approx \uparrow_f^c \varphi^{(j)c} \ , \tag{1.20}$$

where $w^{(j)}$ are already-known low-residual vectors, obtained for example by relaxing the homogeneous system $Ax = 0$, and J is a small number (but possibly exponentially increasing with the underlying (physical) space dimension). The coarse-level Galerkin equation ((1.6), with (1.18)-(1.19)) in this case becomes a set of J equations

$$\sum_{j=1}^{J} A_{l,j}^c \varphi^{(j)c} = r_l^c \ , \qquad (l = 1, \ldots, J) \tag{1.21}$$

where

$$A_{l,j}^c = Q\bar{w}^{(l)} * Aw^{(j)} * P \ , \qquad r_l^c = Q\bar{w}^{(l)} * r \tag{1.22}$$

and $*$ stands for point-to-point multiplication, $\bar{w}$ being the complex conjugate of w.

If *local* (Gram-Schmidt-type) orthonormalization of $\{w^{(j)}\}$ with respect to each other has accompanied the fine-level relaxation, the coarse system (1.21) is dominated by its diagonal blocks $A_{l,l}^c$, and can therefore easily be relaxed. With corrections from several such relaxation sweeps at the coarse level, better $w^{(j)}$ (with lower residuals and wider local orthonormalization) can be obtained at the fine level, followed by re-coarsening in which the diagonal blocks $A_{l,l}^c$ are even more dominating. After more sweeps at the coarser level, as the recursive algorithm now switches to a still-coarser level, the error in each $\varphi^{(j)c}$ should itself be represented in a form similar to (1.20). And so on to ever coarser levels. Due to properly-widening scale of orthonormalization at finer levels, off-diagonal couplings (A_{lj}^c for $l \neq j$) can always be ignored at the *next* coarsening step, keeping the systems at all levels sparse (each unknown function being coupled to only few others).

Geometric model case: waves and rays. A model case for these techniques (first discussed in [Brandt, 1980, §3.2]) is the discretized Helmholtz equation

$$\Delta^h u(\xi) + k(\xi)^2 u(\xi) = f(\xi) \ , \qquad \xi \in \Omega^h \subseteq R^d \ , \tag{1.23}$$

especially on domains with large diameter (compared to the wavelength $2\pi/k$) and with radiation boundary conditions. Ω^h here is the discretized domain in R^d (i.e., it is a d-dimensional grid), Δ^h is the discrete Laplacian, and $u(\xi)$ here is the vector of unknowns (x). This is the model equation for standing (or time-harmonic) waves appearing in electromagnetics, seismology, acoustics, radar, condensed-matter electronics, etc. Traditional multigrid solvers are not effective for this problem, because some "characteristic" oscillatory components (those with wavelength close to $2\pi/k$) are non-local (their size is determined by conditions many meshsizes away) exactly on all those grids which are fine enough to approximate such components.

The geometric multigrid solver developed for this problem [Brandt and Livshits, 1997] represents the solution as

$$u(\xi) = \sum_{j=1}^{J} \varphi_j(\xi) \exp\left(i\psi_j(\xi)\right) . \tag{1.24}$$

At the finest level this sum includes just one term and $\psi_j(\xi) \equiv 0$, so the representation includes just one function — the desired solution — and the equation for it is (1.23). Increasingly coarser levels of the solver (on increasingly coarser grids of ξ to discretize each amplitude $\varphi_j(\xi)$ and each eikonal $\psi_j(\xi)$) employ progressively *finer* sets of "momenta" (i.e., larger J). The interaction between these levels has been shown to yield a solver for (1.23) which is as efficient as the best traditional multigrid solvers for definite elliptic systems. The radiation boundary conditions are naturally enforced at the coarsest level, where the representation essentially coincides with geometrical optics (ray representation, appropriate for scales much larger than the wavelength). This indeed can be developed into a new setting where only geometrical optics is used in most of the domain, while the wave equations, as well as intermediate levels with representations of the type (1.24), are just introduced at special restricted subdomains where geometrical optics breaks down, yielding a general numerical tool for computing diffraction (the rays produced by small-scale disturbances; cf. [Keller, 1962]).

6. Non-local equations: Dense matrices

Fast evaluation (matrix multiply). When the matrix A is dense (not sparse), the first concern, before designing a solver for (1.2), is how to represent A efficiently and how to execute fast the "evaluation", i.e., the multiplication of A by any given vector x. An *arbitrary* $m \times n$ matrix A would of course require mn storage and its evaluation $O(mn)$ computer operations. Very often, however, these can be reduced to just

$O(m+n)$ storage and $O(m+n)$ operations, using the following general approach (first preliminarily appearing in [Brandt, 1982, §8.6] , then in [Brandt and Lubrecht, 1990], [Brandt, 1991b]). Other approaches exist for particular important cases; see, e.g. [Ewald, 1921] and [Greengard, 1994]. In this discussion $m \neq n$ is allowed, although $m = n$ is usually assumed when solving (1.2).

Nearly all non-local operators in physical problems have certain smoothness properties in terms of the coordinates of the underlying d-dimensional physical space (or space and time; thus usually $1 \leq d \leq 4$). Carefully discretized integro-differential operators (see, e.g., [Brandt and Venner, 1998] and [Brandt and Venner, 1999]), as well as interaction of discrete particles, will share these properties. Thus, if the unknown x_j has a location $\eta_j = (\eta_j^1, \ldots, \eta_j^d)$ in the underlying space, and if the i-th equation of (1.2) is associated with the location $\xi_i = (\xi_i^1, \ldots, \xi_i^d)$, then the matrix element a_{ij} can be described by a function $G(\xi_i, \eta_j)$, where the so-called "kernel" $G(\xi, \eta)$, as a function of the continuous variables ξ and η, is either "smooth" or "asymptotically smooth" or suitably "asymptotically smoothly oscillatory" (for the exact meaning of these concepts, see [Brandt, 1991b]).

If $G(\xi, \eta)$ is a smooth kernel (arising, e.g., from first-kind Fredholm integral equations), then each $G(\xi_i, \eta_j)$ can be interpolated from the values $\{G(X_I, Y_J)\}$, where the points $\{X_1, \ldots, X_M\}$ is a coarse grid, or a diluted set of points, in the ξ space (possibly a *subset* of $\{\xi_1, \ldots, \xi_m\}$, with M much smaller than m), and $\{Y_1, \ldots, Y_N\}$ is likewise a coarser grid in the η space (possibly a subset of $\{\eta_1, \ldots, \eta_n\}$); often $\xi_i = \eta_i$ $(i = 1, \ldots, n;\ m = n)$ and then $X_I = Y_I$, $(I = 1, \ldots, N;\ M = N)$. Namely,

$$a_{ij} = G(\xi_i, \eta_j) \approx \sum_{I=1}^{M} \sum_{J=1}^{N} \widetilde{P}_{iI} \widetilde{Q}_{jJ} G(X_I, Y_J) \ , \tag{1.25}$$

where $\widetilde{P}$ and $\widetilde{Q}$ are local (hence sparse) interpolation matrices in the ξ and η spaces respectively ($\widetilde{P} = \widetilde{Q}$ if the spaces are identical, as is often the case). Usually the approximation $\approx$ in (1.25) can be as close to equality as desired, by increasing the interpolation orders. The evaluation of the product Ax can then be executed fast as the product

$$Ax \approx \widetilde{P} A^c \widetilde{Q}^T x \ , \tag{1.26}$$

where $A^c_{I,J} = G(X_I, Y_J)$.

More often $G(\xi, \eta)$ is only *asymptotically* smooth, meaning that it can be decomposed as $G(\xi, \eta) = L(\xi, \eta) + \widetilde{G}(\xi, \eta)$, where $L(\xi, \eta)$ is local, so that the matrix R defined by $R_{ij} = L(\xi_i, \eta_j)$ is sparse, and $\widetilde{G}(\xi, \eta)$ is smooth, so that (1.26) can be generalized to

$$Ax \approx (R + \widetilde{P}\widetilde{A}^c\widetilde{Q}^T)x \ , \tag{1.27}$$

where $\widetilde{A}^c_{I,J} = \widetilde{G}(X_I, Y_J)$. For R to be sufficiently sparse, the coarse grids $\{X_1, \ldots, X_M\}$ and $\{Y_1, \ldots, Y_N\}$ cannot be too coarse, typically MN still being some fraction of mn, hence the multiplication by $\widetilde{A}^c$ may still be expensive. In this case the same algorithm is employed recursively: $\widetilde{G}(\xi, \eta)$ itself is decomposed as $\widetilde{G} = \widetilde{L} + \widetilde{\widetilde{G}}$, where $\widetilde{L}$ is local *in the coarse scale*, allowing $\widetilde{\widetilde{G}}$ to be interpolated from a *still coarser* grid. And so on: using a sequence of increasingly coarser grids, $O(m+n)$ evaluation of Ax is obtained in the form (1.27), where similarly $\widetilde{A}^c = R^c + \widetilde{P}^c\widetilde{A}^{cc}(\widetilde{Q}^c)^T)$, etc.

In most cases of interest, this method can yield one evaluation in $O((m+n)(log\frac{1}{\varepsilon})^q)$ operations, where ε is the desired accuracy and q depends on the dimension d of the underlying space and on the uniformity of the grids $\{\xi_i\}_i$ and $\{\eta_j\}_j$. Typically $q = d$ or $q = d + 1$ for asymptotically smooth kernels.

Fast solvers. Once fast evaluation is available in this form, a rather fast solver of the system (1.2) is in many cases obtained by simple iterations, such as

$$x^{(\nu)} = R^{-1}(b - \widetilde{P}A^c\widetilde{Q}^T x^{(\nu-1)}) \ , \qquad x^{(0)} = 0 \ , \tag{1.28}$$

where multiplications by A^c are calculated by the above fast evaluation scheme, and multiplication by R^{-1} is an approximate solver employing, for example, one sparse-matrix multigrid (or algebraic multigrid) cycle. Actually, quite often R is so strongly diagonally dominant that just one relaxation sweep would suffice (instead of a multigrid cycle) at each iteration (1.28).

A faster solver can usually be obtained by applying multigrid cycles to the full equations (1.2), based on the following principles.

Distributive relaxation. Relaxation of local (sparse) matrices efficiently reduces locally oscillatory errors, leaving more global (e.g., smoother) error components for the coarse grid corrections. For most *dense* matrices (e.g., those representing electrostatic interactions or arising upon descretizing integral or integro-differential equations), however, simple relaxation schemes would efficiently converge smooth components, while being slower for oscillating ones. The latter cannot be approximated on a coarse level, not just because of their character, but more basically because they are too numerous. To reverse this, special schemes, called *distributive relaxation* schemes, should be used [Brandt and Lubrecht, 1990]. Each step of such a scheme distributes changes to *several* unknowns simultaneously, in a pre-designed pattern that ensures little disturbance to non-local (e.g., smooth) components.

Namely, the change to a current solution $\widetilde{x}$, at the i-th step of a distributive relaxation sweep, can be describes as $\widetilde{x} \leftarrow \widetilde{x} + \delta D_i$, where each D_i is a pre-designed sparse vector and the real number δ is calculated so that the new $\widetilde{x}$ satisfies, e.g., the i-th equation of (1.2). For example, each D_i can be designed so that $\widetilde{Q}^T D_i$ vanishes, at least approximately or on the average, where $\widetilde{Q}^T$ is as in (1.27) above. Or D_i can be designed so that its average, and possibly also some higher moments of it as a grid function in the η space, vanish. This relaxation scheme can be viewed as a Gauss-Seidel relaxation for a ghost vector w, where $x = Dw$ and D_i is the i-th column of the "distribution matrix" D.

Such a relaxation scheme has two advantages: it efficiently reduces all oscillatory components (or more generally, the majority of components), and it requires only one global evaluation with the (dense) matrix A per cycle, since the distributive steps do not significantly affect the global part (the term $\widetilde{P}A^c\widetilde{Q}^T$ in (1.27)). One can of course apply such a scheme *only* as part of a multiscale algorithm that supplement it with a coarse-level process for treating that global part — which is discussed next.

Coarsening. It is quite natural to choose $x^c = \widetilde{Q}^T x$ as the coarse-level variables. In fact, if $G(\xi, \eta)$ is smooth then it is only $\widetilde{Q}^T x$ which is well-determined by the equations (1.2), not x itself; more precisely, $\widetilde{Q}^T x$ is well-determined if the grid $\{Y_1, \dots, Y_N\}$ is coarse enough. The coarse system of equations in this case is $A^c x^c = b^c$, where A^c is defined in (1.26) and b^c is a vector such that $\widetilde{P}b^c$ best approximates b, the RHS in (1.2).

If $G(\xi, \eta)$ is *asymptotically* smooth, with A approximated as in (1.27), then the above coarsening can be combined with a Galerkin-type coarsening for the local part R. Namely, the coarsening of (1.3) can be written in the form (1.6), with

$$\begin{aligned} e^c &= \widetilde{Q}^T e \\ A^c &= Q^T R P + \widetilde{A}^c \\ r^c &= Q^T r \ , \qquad e \approx P e^c \end{aligned} \tag{1.29}$$

where the interpolation operators P and Q are derived geometrically or by the methods of Sec. 3 (BAMG in particular, applied to the full matrix A, not just to the local part R).

Ideally, a multigrid solver to a dense-matrix system of equations should cost just a fraction more than just *one* fast evaluation with the desired accuracy (ε), since most of the computations are carried out on coarser levels and/or with lower evaluation accuracies.

7. Non-quadratic optimization: Nonlinear systems

The multiscale methods for fast quadratic minimizations (solving linear systems of equations) can be extended to many (unconstrained) problems having a *non* quadratic energy $E(x)$ with real variables $x = (x_1, \dots, x_n)$. A necessary condition for x to minimize E is that it satisfies the nonlinear system

$$\frac{\partial E}{\partial x_i}(x) = 0 \ , \qquad (i = 1, \dots, n) \ . \tag{1.30}$$

In this section we discuss fast multiscale solvers of such systems, leaving for later (Secs. 9–10) the issue of *global* convergence, namely, how to escape false attraction basins and ensure that the obtained solution of (1.30) is the true *global* minimum.

Relaxation. The relaxation process is essentially the same point-by-point minimization for all problems, quadratic or not, except that in the case that $\partial E/\partial x_i$ is a *nonlinear* function of x_i, the relaxation of each x_i in its turn should not waste work on solving the equation $\partial E/\partial x_i = 0$ *exactly*. Rather, only an inexpensive substantial step toward such a solution should be made, meaning for example just *one approximate local Newton step*, i.e., changing x_i approximately by the quantity

$$\delta x_i = -\frac{\partial E}{\partial x_i} / \frac{\partial^2 E}{(\partial x_i)^2} \ , \tag{1.31}$$

where the derivatives are calculated just before this particular step. Actually, the second derivative need be calculated only approximately, and need not be updated at each relaxation sweep. Various terms contributing little to $\partial^2 E/(\partial x_i)^2$ can be neglected without changing the overall efficiency. Such terms, called *non-principal terms*, cannot however be neglected in calculating $\partial E/\partial x_i$.

Coarsening and Global Newton linearization. The way to extend coarsening and coarse-level corrections to nonlinear systems is less straightforward. One obvious general approach is *global* Newton linearization. Namely, given a current approximation $\widetilde{x}$ (e.g., after relaxation), the system (1.30) is approximated by the *correction equation*

$$A(\widetilde{x})e = b(\widetilde{x}) \ , \quad a_{ij}(\widetilde{x}) = \frac{\partial^2 E}{\partial x_i \partial x_j}(\widetilde{x}) \ , \quad b_i(\widetilde{x}) = -\frac{\partial E}{\partial x_i}(\widetilde{x}) \ , \tag{1.32}$$

where $A(\widetilde{x}) = \{a_{ij}(\widetilde{x})\}_{i,j=1}^{n}$ is the current Jacobian, $b(\widetilde{x}) = \{b_i(\widetilde{x})\}_{i=1}^{n}$ is the current *residual* (or "residual force") vector, and $e \approx x - \widetilde{x}$ is the correction sought. A coarse-level approximation to e, denoted e^c, can be obtained by any of the coarsening methods described above for linear systems: multigrid in the case of PDEs descretized on structured grids (Sec. 2), or algebraic multigrid (classical AMG or BAMG — see Sec. 3) in more general cases.

The advantage of the Newtonian approach is indeed exactly this: the possibility to use all sophisticated machinery developed for linear systems to obtain robust coarse-level approximations, even for highly discontinuous and disordered equations. The usual *disadvantages* are the need to perform iterations (usually not many, though), the need to start with a good enough first approximation to ensure convergence, and the need to store and perhaps repeatedly update the matrix $A(\widetilde{x})$. In addition, from the point of view of multiscale solvers, global linearization has a most serious disadvantage in the very common case of *autonomous*, and hence *highly repetitive*, nonlinear systems. By "autonomous" system we mean a system whose local equations are independent of external information, such as a current approximate solution. In such systems, the nonlinear equations are the same repeating equation everywhere, or nearly everywhere, or the same set of few equations keep repeating itself. This in principle would allow accurate *nonlinear* coarse equations to be derived in some representative small regions and then be used all over large domains (see "systematic upscaling" below). This autonomy of the nonlinear system is however lost upon linearization.

Exact Interpolation Scheme (EIS), reviewed in Sec. 8 below, is a general promising approach that avoids global linearization iterations in optimization problems. Here however we focus on the following, currently more popular and tried out scheme.

Full Approximation Scheme (FAS). Writing now the nonlinear fine level system (1.30) in the form

$$N(x) = 0 \; , \tag{1.33}$$

a corresponding coarse-level nonlinear system

$$N^c(\hat{x}^c) = 0 \tag{1.34}$$

is often directly accessible. In particular, if the system of equations (1.33) represents a discretization of some continuum equations (PDEs, for example), then a similar discretization can be obtained for the coarse grid (often coded by the same computer program, with different discretization parameters). If $\widetilde{x}$ is the *current* (e.g., after relaxation) fine-level approximation, and hence $r = -N(\widetilde{x})$ is the current residual, then a general coarsening scheme is to approximate the fine-level residual equations

$$N(x) - N(\widetilde{x}) = r \tag{1.35}$$

by the nonlinear coarse system

$$N^c(x^c) - N^c(\widetilde{x}^c) = \uparrow_f^c r \ , \tag{1.36}$$

where $\uparrow_f^c$ is a fine-to-coarse transfer as defined, e.g., in Sec. 3, and (the unknown) x^c and (the known) $\widetilde{x}^c$ are the coarse versions of x and $\widetilde{x}$, respectively. (Note that generally x^c is *not* the solution $\hat{x}^c$ of (1.34); unlike the latter, it would have the accuracy of the *fine* level in approximating the PDE solution.) It is important to remember that it is the *error* $e = x - x^c$ which is smoothed by relaxation and thus approximated by the residual equations (1.35), so after obtaining a solution x^c to (1.36), it is the correction $x^c - \widetilde{x}^c$ which should be interpolated, i.e, the coarse-grid correction to the fine level is

$$\widetilde{x}_{\text{NEW}} = \widetilde{x} + \uparrow_c^f (x^c - \widetilde{x}^c) \ . \tag{1.37}$$

It is also important to use exactly the same $\widetilde{x}^c$ in (1.36), in (1.37) and as the first approximation in the process of solving (1.36), so that the solver is *stationary*, meaning that if the exact solution has already been obtained at the fine level ($r = 0$, hence $\uparrow_f^c r = 0$, hence $x^c = \widetilde{x}^c$) then (1.36) is immediately at its solution, and then (1.37) does not change $\widetilde{x}$.

This scheme, introduced in [Brandt, 1977], is called the Full Approximation Scheme (FAS), because it gives directly an equation in terms of the full solution x^c, not in terms of the correction e^c. Defining

$$\tau^c = N^c(\widetilde{x}^c) - \uparrow_f^c N(\widetilde{x}) \ , \tag{1.38}$$

Eq. (1.36) can be written as

$$N^c(x^c) = \tau^c \ . \tag{1.39}$$

Except for a different forcing function, this equation has the same form as the original coarse equation (1.34), and can use the same computer program. Neither storage nor updating of linearization coefficients (such as $A(\widetilde{x})$ above) are needed. No global linearization iterations are performed, and the solver is usually as inexpensive as solving just *one* corresponding linear equation: For discretized PDEs, just one cycle per level in an FMG algorithm will typically yield errors well below the discretization errors (cf. Sec. 2). Such FMG-FAS algorithms are widely used in many fields of PDE applications.

Yavneh's Multilevel Nonlinear Method (MNM). Sometimes, or in some subdomains, the given coarse-level operator N^c is not good or robust enough approximation, while the Newton linearization $A(\widetilde{x})$ does have a good and robust coarsening $A^c(\widetilde{x})$, obtained for example by an advanced AMG algorithm. Then, instead of $N^c(x^c)$, an improved coarse level operator that can be used in the FAS algorithm is

$$\widetilde{N}^c(x^c) := N^c(x^c) + A^c(\widetilde{x}) - \widetilde{A}^c(\widetilde{x}^c) \ , \tag{1.40}$$

where $\widetilde{A}^c(\widetilde{x}^c)$ is the Jacobian of N^c at $\widetilde{x}^c$. This MNM method [Yavneh and Dardyk] may enjoy wider convergence basin than either Newton or the simple FAS method. More generally one can use

$$\widetilde{N}^c(x^c) := \alpha N^c(x^c) + \beta A^c(\widetilde{x}) - (1 - \alpha - \beta)\widetilde{A}^c(\widetilde{x}^c) \ , \tag{1.41}$$

giving MNM for $\alpha = \beta = 1$, Newton's method for $\alpha = 0$, $\beta = 1$ and simple FAS for $\alpha = 1$, $\beta = 0$. The coefficients α and β can be vector parameters chosen locally, for example switching to Newton's method near a large discontinuity or to the simple FAS near strong nonlinear effects.

Quasilinearity. Nonlinear problems can often usefully be written in the *algebraic quasilinear* form $A(x){\cdot}x = b$, where the dependence of $A(x)$ on x is *non-principal*, by which we mean that $\| A(x+\delta)\cdot(x+\delta) - A(x)\cdot(x+\delta) \| \ll \| A(x)\cdot\delta \|$ for any small δ. For example, most nonlinear PDE systems in mathematical physics are *differentially quasilinear*, meaning that each term in the system is linear in the highest derivative included in it; then in the discretization, only the dependence on the highest derivative (in each such term) is principal, so the algebraic quasilinearity comes there naturally. Unlike Newton linearizations, this quasilinear discretization can be *autonomous* wherever the PDE is autonomous.

In a quasilinear system, to a very good approximation the interpolation $\uparrow_c^f$ depends only on $A(x)$ and can therefore be derived as in Secs. 2–3. Moreover, $\uparrow_c^f$ needs seldom be changed when x changes. Also the form of $A(x)$ is often simple and explicit; e.g., in fluid dynamics and other areas, each term in A depends on x *linearly*. It is then possible to transfer this form of dependence also to the coarse level, enabling the employment of an FAS-like algorithm, thus solving the nonlinear problem *directly*, without linearizations.

Fine-to-coarse defect correction and local grid refinements. The function τ^c defined in (1.38) is called the *fine-to-coarse defect correction*. It has many applications (see e.g., [Brandt, 1982]). *In solving PDE systems*, τ^c can serve as a good approximation to the local discretization error of the coarse level, and by a proper scaling this yields an estimate also for the *fine*-level local discretization error, which gives the criterion to decide whether the discretization is fine enough. ("Local" errors means errors in satisfying the *equations*. Large such errors (indicated by large values of τ^c), even when appearing only in one small subdomain, would pollute (i.e., cause errors in) the entire *global* solution, hence must be corrected by grid refinements.)

Often, the discretization grid needs to be refined only in small regions, e.g., near singularities. The FAS scheme is very convenient, efficient and flexible for this purpose: the refinement is done by adding a finer level to the multigrid system, consisting of patches of finer grid that cover only the required regions. Using (1.34) everywhere and (1.39) in the refined regions, where local processing (interpolation to the finer patches, followed by relaxation there) yields the needed correction τ^c. Part of the refined regions may require further refinement (indicated by large τ^c), and so on, leading to *multilevel nested self-adaptive local grid refinements*. An automatic grid self-adaptation process can naturally be integrated into the FAS-FMG algorithm: as the FMG (see Sec. 2) proceeds to increasingly finer levels, it can also decide (using the τ^c criterion) *where* those finer levels should be employed. This yields a *one-shot solver-adaptor*, with no need for grid-adaptation iterations. Furthermore, each of the local refinement patches can use a grid with its own coordinate system, aligned with solution characteristics or fitted to the local geometry (such as spherical coordinates around an isolated singularity, or local boundary-fitted grids in flow boundary layers), allowing local anisotropic grids with very high aspect ratios, while the global grids remain simple Cartesian. (For details consider [Brandt, 1982], [Brandt, 1984] and [Bai and Brandt, 1987].)

An important feature of this adaptation is that often the calculation within the local-refinement patch can be done *once for all*: Although the solution in the patch changes when the parent-grid solution changes, the fine-to-coarse defect corrections usually change very little. At most one more short "visit" to the patch (e.g., one more relaxation sweep at the finer level) toward the end of the calculation will normally be needed to update the defect corrections. Alternatively, one can calculate apriori the approximately linear dependence of the defect corrections on the local parent-grid values.

A similar FAS structure can be used to solve problems on *unbounded domains*, using increasingly coarser levels to cover increasingly larger subdomains, controlled by criteria based on τ^c (see [Brandt, 1997, §4] or [Brandt, 2001, §6.2]).

First Approximation: Continuation and Bifurcation. For nonlinear problems, a common way to obtain a good first approximation, or generally to trace branches of solutions, is by *continuation* (also called "embedding" or "Davidenko method"): The problem, including its discretization and its approximate solution is written as depending on some continuous parameter, like γ in the range

$$\gamma_0 \le \gamma \le \gamma_* \ , \tag{1.42}$$

where for γ_0 the problem is easily solvable (e.g., it is linear), while for γ_* it is the problem one really needs to solve. One advances γ from γ_0 to γ_* in steps $\delta\gamma$ small enough to ensure that the solution to the γ problem can serve as a good first approximation in solving the $\gamma + \delta\gamma$ problem. Sometimes γ is a physical parameter; sometimes the solutions are better defined in terms of a non-physical parameter, such as the generally-applicable *arclength* of the solutions path [Keller, 1977].

In multigrid solvers, one will usually perform *one* FAS or Newtonian cycle per continuation step. Better still, in solving PDEs by an FMG-FAS algorithm, one can often integrate the continuation process into the FMG sequence of increasingly finer levels, by attaching to them progressively more advanced values of γ, thus avoiding the need to iterate. Note that here, as also elsewhere, the notion of "coarser grid" is generalized to the notion of "simpler-to-solve approximation", in the sense of having both fewer variables and easier parameter values (e.g., smaller γ). Note also that a proper continuation parameter may sometimes enter the FMG process even without explicitly intending it, such as the parameter of viscosity in flow problems, which is larger at coarser levels due to numerical viscosity.

For calculating *bifurcation diagrams* (curves that trace solution dependence on one or several parameters, revealing multiple-solution branches, bifurcation points, etc.), various multigrid situations arise. For some problems the whole continuation process can be done on very coarse levels, proceeding only with the target γ_* to finer levels to obtain improved accuracy. In other problems, the continuation process does need fine-level resolution of important features, but it can still be done mostly at coarse levels by providing those levels with the fine-to-coarse defect corrections τ^c, which may need only little updating from visits to finer levels; the finer the level, the rarer the visits.

See more about multigrid continuation strategies in [Brandt, 1982, §8.3.2], [Mittelmann, 1982], [Bank and Mittelmann, 1986], [Bolstad and Keller, 1986], [Schwichtenberg, 1985], [Trottenberg *et al.*, 2000, §10.2]. See also a related comment in the discussion of global constraints in Sec. 8 below.

Systematic upscaling of autonomous systems. In autonomous systems, the same small set of (nonlinear, local) equations repeats itself all over a large domain, so it is desired to construct for that domain a similar small set of *coarse autonomous equations*.

The coarse *variables* should be chosen according to the same criterion as in linear systems, based on the compatible-relaxation convergence speed (see Sec. 3).

Coarse *equations* can take the general form of a detailed numerical *dependence table*, giving detailed account of the dependence of any coarse variable (the "pivot") on its neighboring coarse variables (the "neighborhood"). For each "neighborhood bin" (some narrow range of neighborhood values), the average value of the pivot is calculated during local *fine-level simulations* (relaxation sweeps). Due to the near-locality property (see Sec. 3), accurate dependence tables can be constructed efficiently using a *branching structure*. In such a structure, any current bin with enough cases in it is partitioned into finer bins, either representing a finer resolution of the same neighborhood or an *expansion* of the neighborhood (i.e., sub-binning according to values of *additional*, e.g., more remote, neighbors; see an example in [Brandt and Ron, 2001], where such a branching method was first introduced).

Having prepared the dependence table, for any actual neighborhood, the corresponding value of the pivot can be calculated by a suitable *interpolation* from that table. This is all one needs in order to run *coarse-level simulations* (point-by-point relaxation sweeps). Such simulations can then be used to construct a branching dependence table for the *next*, still coarser level, and so on. The coarser the level the larger the physical domain on which its simulations are performed. Whenever at some coarse level a neighborhood arises for which there is no adequate statistics, a return to the next finer level can be done in some local *window* (a local refinement patch over the coarse level region suffering lack of statistics), to gather more statistics. This is done by first running several passes of *compatible* relaxation (see Sec. 3) over the window, followed by *usual* (not compatible) relaxation sweeps in the *interior* of the window, while keeping a zone around its boundary compatible.

Such a derivation of coarse nonlinear equations is admittedly expensive, but easily affordable for autonomous or highly repetitive systems, with the important benefit of avoiding fine resolutions of very large domains. Note that the equations often become much less autonomous in special parts, e.g., near boundaries, so separate dependence tables may need to be constructed there, that will involve external data (such as distance to boundary and boundary conditions) in their "neighborhoods".

Much less expensive coarse equations can be obtained by settling for lower coarsening accuracy, then using that approximate coarse system as N^c in FAS cycles. Conversely, if an approximate coarsening N^c is known (e.g., a coarser discretization of the same PDE), the systematic upscaling method can use it to derive less expensively a more accurate coarsening: The dependence table described above is constructed not for the full value of the pivot, but only for the value of the fine-to-coarse defect correction τ^c.

The systematic upscaling method has been inspired by an analogous approach first developed for non-deterministic problems (the "renormalization multigrid" (RMG) approach: see Sec. 9), and like the latter, it may be extended to time-dependent problems. In fact, a systematic upscaling of a deterministic system may in some cases give rise to non-deterministic coarser systems.

8. Constrained optimization and eigenproblems

Frequently the problem of minimizing $E(x) \equiv E(x_1, \ldots, x_n)$ is subject to side conditions, possibly including L equations

$$f_l(x) \equiv f_l(x_1, \ldots, x_n) = 0 \ , \qquad (l = 1, \ldots, L) \ , \tag{1.43}$$

and M inequalities

$$g_m(x) \equiv g_m(x_1, \ldots, x_n) \geq 0 \ , \qquad (m = 1, \ldots, M) \ . \tag{1.44}$$

Assuming existence of continuous derivatives of E, $\{f_l\}$ and $\{g_m\}$ at the minimum, the solution x will satisfy, instead of the system (1.30), the augmented system of $L + M + n$ relations (1.43), (1.44) and (1.45),

$$\lambda_0 \frac{\partial E}{\partial x_i}(x) = \sum_{l=1}^{L} \lambda_l \frac{\partial f_l}{\partial x_i} - \sum_{m=1}^{M} \mu_m \frac{\partial g_m}{\partial x_i} \ , \qquad (i = 1, \ldots, n) \ , \tag{1.45}$$

having the "Lagrange multipliers" $\lambda_1, \ldots, \lambda_L$ and $\mu_1, \ldots, \mu_M$ as additional unknowns ($\lambda_0 = 1$, except in some "abnormal cases" in which $\lambda_0 = 0$), together with the "complementarity conditions"

$$\begin{array}{lll} \mu_m \geq 0 & \text{if} & g_m(x) = 0 \ , \\ \mu_m = 0 & \text{if} & g_m(x) > 0 \ , \qquad (m = 1, \ldots, M) \ . \end{array} \tag{1.46}$$

The following are some general comments concerning the multiscale solution of such augmented systems.

Local equality constraints: optimal control and inverse PDE. In the case of equality constraints only ($M = 0$), the fast multiscale solution methods described above, with their various extensions, would usually apply in principle to the augmented system (1.43)+(1.45) as well, particularly when the constraints (1.43) and $\{\partial E/\partial x_i\}$ are all local. This is the typical situation in *differential optimal control* problems, where the equality constraints (1.43) correspond to the discretization of the underlying constitutive differential equations, and $E(x)$ is some performance index. This is also the case in many *inverse PDE problems*, in which incomplete, noisy or ill-posed data are given, supplemented by a certain solution fitness measure $(-E)$ which has to be optimized.

The Lagrange multipliers λ in such problems correspond, similar to x, to one or several discretized functions, so the equations to be solved correspond to a discretized augmented PDE system. To achieve top multigrid efficiency in such problems, however, is usually more complicated than in standard PDE problems, since their mathematical character is less straightforward. For example, for constitutive PDE which is a hyperbolic time-dependent system, the optimal control (or fitness index) introduces backward-in-time couplings, such that the problem cannot be solved by simple time marching. It need be treated by *full-dimension multigrid* (coarsening both the space *and* time dimensions), which is, however, quite tricky, since the underlying hyperbolic nature implies that some high-frequency (non smooth) components are not local, prohibiting simple smoothing. The relaxation process should depend on the level: On sufficiently fine levels, locally dominated by the underlying PDE (or its discretization (1.43)) and its adjoint (resulting from the right-hand side of (1.45)), the relaxation would typically involve forward time marching of changing x according to the underlying PDE, and backward marching of changing λ according to the adjoint equations. On coarse levels (on scales comparable, e.g., to the control horizon), the coupling terms between the PDE and its adjoint (resulting from E) become dominant and should be relaxed according to their character (see example in [Gandlin, 2002]).

This fundamentally different treatment necessarily needed at dissimilar scales explains why such problems are often impossible to solve without multiscale techniques, thus traditionally requiring various compromises in their formulation and discretization. Another important benefit of multiscaling is that the separate processing at different scales makes it possible to attach different objectives (performance indices or fitness measures) at different scales (see Sec. 12 below).

Often the assimilation of non-standard, ill-posed data into a direct PDE solver should cost relatively little, i.e., the extra computer time needed for the data assimilation should ideally be smaller than the time required by the direct solver itself. This is because: (1) Large scale data *averages* can inexpensively be assimilated on correspondingly coarse levels of the solver (coarser both in space and time). (2) *Deviations* from such averages should be assimilated at finer scales, but *their* correlations on those scales are local. (3) The underlying equations must usually be resolved at scales finer than those on which assimilation should be done. The overall solver of an ill-posed inverse problem can sometimes cost even far less than the solver of a corresponding well-posed problem, since ill-defined high frequencies (e.g., at regions far from measurements) need not be calculated at all. See examples in [Gandlin, 2002], and discussion in [Brandt, 2001, §4] listing several additional potential benefits of multiscaling for data assimilation problems.

Feedback optimal control. In *feedback* optimal control problems new initial values of the underlying PDE are continuously fed from the controlled device, requiring very fast real-time updating of the control. The fast multigrid solver for the *open-loop* (i.e., not feedback) optimal control problem allows super-fast updates upon feedbacks, based on the observation that, upon changing the initial conditions, the change in the solution is increasingly smoother at times increasingly far from the initial. (In various actual problems, the sense of this smoothness has to be carefully understood.) This makes it possible for the multigrid *re*-solving algorithm to re-process its *fine* grids only at the very early times, while at increasingly later times only progressively coarser levels are re-processed, with FAS fine-to-coarse defect corrections (τ^c) being frozen there (cf. Sec. 7). As a result, the computational cost of re-resolving is equivalent to only *local* re-processing (essentially just few steps near the initial time) of the full solver.

Global and intermediate-scale constraints. Problems often come with one or several global conditions, i.e., a single condition with a large effect on a substantial fraction of the unknowns. For example, if all equations in a system give only differences between unknowns, except for one equation that gives, say, the average of all unknowns, then a change in that average would introduce an identical change to *every* unknown. In fact, a global condition need not itself explicitly involve all or many of the unknowns. The exceptional equation in the example, instead of giving the average of all unknowns, may specify the value of just *one* of them — it will still have the same global effect. A boundary or initial condition for an *ordinary* (i.e., one-dimensional) differential equation, or its discretized version, is a global condition. There may also be *implicit* global conditions. For example, the average along the boundary of the values of a boundary condition of a *partial* (i.e., higher dimensional) differential equation is a global condition, even when it is not specifically given as a single condition.

There may also be conditions which are neither local nor global but have some intermediate scale. For example, instead of boundary or initial conditions, a PDE may be supplemented with a data set representing measurements of solution values. The *scale* of each such value is, roughly, its typical distance to neighboring values.

A global (or intermediate-scale) condition most often comes with a corresponding global (or intermediate-scale) *unknown*, such as the Lagrange multiplier corresponding to a global constraint in a minimization problem. In continuation processes (see Sec. 7) a global condition (fixing a parameter, such as the arclength of the continuation step [Keller, 1977]) is often added to a problem while "freeing" (i.e., turning into global *unknown*) one of the original problem parameters, thereby obtaining a better-posed problem (having for example a unique solution in terms of the new parameter instead of multiple solutions in terms of the original parameter).

In multiscale solvers, for best efficiency, the simple rule is to avoid relaxing a global condition or changing a global unknown at any fine level of the multigrid cycle. All one has to do on such levels is to transfer the residual of the condition to the next coarser level to serve as a forcing term in a similar condition on that level. *Each condition and unknown should best be treated (relaxed, changed) at the level corresponding to its scale.* Sometimes a treatment at the wrong level may even be damaging, not just a waste of effort. (See some supplementary rules in [Brandt, 1982, §5.6].)

This rule is often useful also for an *implicit* global condition. For example, in relaxing boundary conditions of a PDE, it is often best to relax at each step the *difference* between the conditions at two neighboring boundary points, leaving the *average* of all such conditions unchanged, since it constitutes an implicit global condition. This is in fact a special case of the following.

Distributive relaxation. Since the scale of side conditions is often not local, the relaxation should generally be *distributive*, as in the case of non-local equations (see Sec. 6). A suitably chosen distributive change D_i of an unknown function (either a control function, or an unknown PDE coefficient, or a Lagrange-multiplier function) has only localized effect, hence its size (δ in Sec. 6) can inexpensively be calculated.

Eigenproblems. A frequent global condition is a normalization condition, such as $\frac{1}{2}\sum x_i^2 = 1$ or more generally $\frac{1}{2}x^T Bx = 1$, where B is a positive definite matrix. For the case that $E(x) = \frac{1}{2}x^T Ax$, Eq. (1.45) then takes the form of the generalized eigenvalue problem: finding the smallest λ such that the normalization condition is satisfied and

$$Ax = \lambda Bx \ . \tag{1.47}$$

The methods reviewed in earlier sections (multigrid and algebraic multigrid) are applicable to this eigenproblem, together with the general rule mentioned above for treating global conditions and unknowns; namely, the *eigenvalue* (the Lagrange multiplier λ) is frozen, and the normalization condition need not be relaxed, at fine levels. In the linear case, due to homogeneity, the enforcement of the normalization condition needs in fact be done only once, at the very end.

If a good first approximation to λ and x is not yet available (e.g., at fine levels, before visiting the coarsest levels), $\lambda = 0$ can be used in relaxing (1.47). A very suitable relaxation to use in this process is the Gauss Seidel scheme.

Additional constraints often lead to the need to compute *several*, say K, lowest eigenvectors, i.e., K different solutions to the eigenproblem

$$Ax^{(k)} = \lambda_k Bx^{(k)} \ , \qquad \frac{1}{2}x^{(k)T}Bx^{(k)} = 1 \ , \qquad (k = 1, \ldots, K) \tag{1.48}$$

with the lowest eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_K$. A typical cycle of a possible approach, developed in [Brandt *et al.*, 1983], is to apply a multigrid cycle to each of K approximate eigenvectors to obtain new approximations which together will therefore span an improved approximation X to the K-dimensional *subspace* spanned by the lowest eigenvectors. This is then followed by a *Ritz projection*, which finds a basis in X closest to the sought set of eigenvectors, by calculating vectors $\widetilde{x}^{(1)}, \ldots, \widetilde{x}^{(K)} \in X$ and values $\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_K$ such that the orthogonal projection on X of each $A\widetilde{x}^{(k)} - \widetilde{\lambda}_k B\widetilde{x}^{(k)}$ vanishes $(k = 1, \ldots, K)$.

This "classical" approach is suitable for calculating several low eigenvectors in a geometrical environment which allows using a straightforward multi-polynomial coarse-to-fine interpolation. In many cases such interpolation is not available. In AMG solvers the interpolation is to be *found*. In fact, calculating low eigenvectors and finding a suitable interpolation depend on each other (see Sec. 3, the BAMG processing in particular). Moreover, even in geometrical multigrid with well structured grids, standard interpolation is not suitable for *high* eigenvectors, where the needed coarse-level correction is oscillatory, as in highly indefinite problems (cf. Sec. 5). In all these and other situations, especially where interpolation should be designed as part of the solution process, the following general scheme seems most suitable.

Exact Interpolation Scheme (EIS). The general EIS approach for coarsening the constraints optimization problem (1.43)-(1.46) is first to adapt the interpolation operator $\uparrow_c^f$ so that $\widetilde{x} = \uparrow_c^f \widetilde{x}^c$ *exactly*, where $\widetilde{x}$ is the *current* fine-grid approximation and $\widetilde{x}^c$ is its coarse-level representation, and then to derive the coarse equations from the resulting coarse minimization problem, namely, the minimization of $E(\uparrow_c^f x^c)$ under the constraints $\{f_l(\uparrow_c^f x^c) = 0\}_l$ and $\{g_m(\uparrow_c^f x^c) \geq 0\}_m$.

In particular, in the case of the linear eigenproblem (1.48), one can choose $\uparrow_c^f$ to be a *linear* (not just affine as in FAS) operator such that $\widetilde{x}^{(k)} = \uparrow_c^f \widetilde{x}^{(k)c}$ *simultaneously* for all the current approximate eigenvectors ($k = 1, \ldots, K$; see the discussion below about large K). This yields the coarse-level eigenproblem

$$A^c x^{(k)c} = \lambda_k B^c x^{(k)c} , \qquad \frac{1}{2} x^{(k)cT} B^c x^{(k)c} = 1 , \qquad (k = 1, \ldots, K) \quad (1.49)$$

where $A^c = \uparrow_c^{f^T} A \uparrow_c^f$ and $B^c = \uparrow_c^{f^T} B \uparrow_c^f$. Unlike classical CS and FAS multigrid, no residuals are transferred here from fine to coarse, and the coarse level problem has the same natural form of a generalized eigenproblem. At convergence, orthonormality at the coarse level, in the sense that $\{\frac{1}{2} x^{(l)T} B^c x^{(k)} = \delta_{kl}\}_{k,l=1}^K$, automatically implies orthonormality at the fine level.

Following relaxation of (1.49), interpolation from a still coarser level to the first coarse level can be designed to fit the current approximation to $\{x^{(k)c}\}_{k=1}^K$, analogously yielding eigen-equations for that second coarse level, and so on to a coarsest level whose eigenproblem can inexpensively be solved directly. The coarsest-level solution yields orthonormality and approximate eigenvalues, which are then inherited by successively finer levels, each obtained from the next coarser level by the designed interpolation followed by one or several relaxation sweeps. This complete an EIS cycle. No Ritz projection in needed.

Many low eigenvectors. In order to have an interpolation that is simultaneously stationary for (i.e., satisfied by) a large number K of current approximate eigenvectors, a correspondingly large number of interpolation points must be used ($|N_i|$, the number of points in the neighborhood N_i in (1.17), should be at least K, for every i, in contrast to the condition stated there). In extensive problems (where n is large while the effective underlying (e.g., physical) dimension is low) there can be a large number of low eigenvectors which satisfy, to a good approximation, the same interpolation rule (e.g., all of them are smooth, hence closely satisfy one regular second-order polynomial interpolation). This interpolation is good enough for the first EIS cycle. To boost accuracy in the next cycle, the numbers $|N_i|$ can be raised. As they increase, the interpolation effectively acquires progressively higher orders, hence it can fit *very* closely even a much larger number ($K \gg |N_i|$) of low eigenvectors. If absolute algebraic accuracy (vanishing residuals) is desired without further increasing $|N_i|$, some relaxation sweeps or even cycles can be added in which a (very small) FAS-like affine part is appended to the constructed joint linear interpolation. This however may well be unnecessary, since errors smaller than the discretization errors are obtained as soon as the effective interpolation order exceeds the discretization order.

Most eigenvectors that can be considered low at the fine level (in the above sense of satisfying to a good approximation a joint interpolation rule) may no longer be low at the scale of some coarser level. This brings them at that level to the situation discussed next.

Higher eigenvectors. As K gets even larger (or the level coarser), a point is reached where the K eigenvectors cannot all satisfy the same interpolation. The EIS is still applicable, except that different interpolation operators should be adapted to different (not necessarily disjoint) subsets of eigenvectors. Also the eigensystem (1.48) for high eigenvalues λ_k is highly indefinite, so the rules of Sec. 5 enter: a suitable relaxation scheme (such as Kacmarz or least-square) should be employed, together with a multiple-interpolation coarsening (1.20)-(1.22). Note that that interpolation is designed so that the coarse-level representations are smooth, even when the approximate fine-level eigenvectors are oscillatory. This EIS multigrid is appropriate for computing high eigenvectors even *without* calculating lower ones.

Multiscale Eigen-Base (MEB). The above description indicates that even when a large number of eigenstates is of interest, only few of them need be represented at the finest level: as few as the number of distinct interpolation operators necessary to fit to a good approximation all eigenvector subsets. At the coarser level a *larger* number of eigenvectors should be separated out, and so on: for linear systems of local equations it is expected that the total number of eigenstates that should be separated out at each level is inversely proportional to the number of variables on that level. To obtain this collective representation – the so called *multiscale eigenbase* (MEB) – it is not necessary to orthogonalize each eigenvector with respect to each other: On each level each representative eigenvector need be locally orthogonalized only with respect to its “siblings” in this tree-like collective structure.

A work along these ideas is in progress. As a preliminary demonstration, a kind of MEB structure has been developed for general discretized linear *ordinary* differential operators [Livne, 2000], [Livne and Brandt, 2000]. It has been shown that the MEB construction for K eigenvectors costs only $O(n\log K)$ computer operations and computer storage, and that with a similar amount of work typical information of interest can be extracted from this compact structure. In particular, it costs again only $O(n\log K)$ *operations to expand a given function in terms of the K eigenvectors*. This constitutes a vast generalization of the Fast Fourier Transform (FFT) efficiency, whose basis functions are the eigenstates of discretized differential operators with *constant* coefficients, *periodic* boundary conditions with 2^l *uniformly* spaced gridpoints. The new $O(n\log K)$ expansion is in terms of the eigenfunctions of a general variable-coefficient operator with general boundary conditions and a general number of unevenly spaced gridpoints.

Inequality constraints. One of the earliest computational fields to employ multilevel procedures is the field of *production planning*, notably in the Soviet Union. Such procedures were quite naturally introduced there, since the multilevel structure was already explicit in the problems themselves, due to the hierarchical division of the economy into sectors and its pyramidal management. This led to the introduction of iterative "aggregation/disaggregation" (a/d) algorithms, starting already in the mid sixties [Dudkin and Yershov, 1965] and growing in the seventies to extensive Russian literature on iterative a/d procedures for large linear programming problems (see [Vakhutinsky *et al.*, 1979]).

Multigrid methods have led to the realization that rather than the explicit hierarchy exploited by those methods, greater efficiency and generality can be obtained by introducing as many intermediate levels as possible, as long as this does not substantially increase the total number of variables on all levels. So far, not many inequality-constrained problems have been solved this way, though.

For discretized continuum problems, a fairly general multigrid solver can be based on projected Gauss Seidel (PGS) relaxation together with full-approximation-scheme (FAS) coarsening. In PGS, each unknown x_i in its turn is changed so as to minimize $E(x)$ as far as possible *in the feasible range* (the range where the constraints are satisfied). The FAS coarsening (see Sec. 7) is convenient for expressing the inequalities on the coarser level. On interpolating the calculated correction from coarse to fine, a slightly infeasible approximation may be produced, but subsequent relaxation should easily fix that, and the algorithm is stationary at the feasible minimum.

In [Brandt and Cryer, 1983], this approach was used to minimize the Dirichlet integral $\int_\Omega [\frac{1}{2}(\nabla \cdot u(\xi))^2 + f(\xi) \cdot u(\xi)] d\xi$, suitably discretized, given $f(\xi)$ in Ω and $u(\xi)$ on the boundary of Ω, under the constraint

$u(\xi) \geq 0$ for all $\xi \in \Omega \subseteq R^2$. Without this constraint the problem is equivalent to the second-order elliptic Poisson equation. The PGS-FAS multigrid solved the constrained problem in essentially the same "textbook" efficiency at which the Poisson equation is solved.

Another possible, but little tried, approach to optimization problems with equality+inequality constraints is the EIS mentioned above.

Experimental studies of multigrid-like solvers to the linear programming *transportation problem* is reported in [Kaminsky, 1989] and [Nilo, 1986].

9. Non-deterministic systems

A general ingredient in methods to escape false attraction basins in search for the global minimum is borrowed from physical processes associated with non-deterministic finite-temperature systems. We will therefore first review these systems and the conventional and multiscale methods for treating them, then, in the next section, proceed to the issues of global minimization.

Indeed the energy minimization problem can be viewed as the zero-temperature limit of non-deterministic finite-temperature problems. According to the theory of statistical mechanics, if a physical system with variables $x = (x_1, \ldots, x_n)$ and a potential-energy functional $E(x)$ has a thermal contact with a heat reservoir at absolute temperature T over a sufficiently long time period, then the probability (or probability density) $P_E(x)$ to obtain any particular configuration x at any given moment is given by

$$P_E(x) = \frac{1}{z(T)} e^{-E(x)/k_B T} \;, \tag{1.50}$$

where k_B is the Boltzmann constant and $z(T)$ is a constant determined by the requirement that the sum (or integral) of $P_E(x)$ over all possible configurations x is 1; namely, $\sum_x P_E(x) = 1$. Clearly, as $T \to 0$ the only probable configurations are those with the lowest energy.

Large systems (large n) of this kind arise as central problems in many areas of physical sciences, including elementary particles theory, quantum mechanics and molecular dynamics. The aim of computations is to calculate various statistical averages of various "observables", such as $M(x) = \sum_i x_i$ and $(M(x))^2$, $E(x)$ and many others, where the average of an observable $O(x)$ is defined by $< O >= \sum_x P_E(x) O(x)$ (or integration instead of summation when x_i are continuous variables).

Monte Carlo. A general method to calculate such averages is the *Monte Carlo* (MC) *method*, whose aim is to produce a sequence ("chain") of configurations $x^{(1)}, x^{(2)}, \ldots$ with the physical probability distribution (i.e., the probability of each $x^{(k)}$ to be a certain configuration x is $P_E(x)$), so that $< O >$ can be approximated by averaging over this sequence

$$< O > \approx \frac{1}{K} \sum_{k=1}^{K} O(x^{(k)}) \ . \tag{1.51}$$

To obtain such a fair-sampling sequence of configurations a typical MC sweep scans the variables $(x_1, \ldots, x_n)$ in some order, changing one (or few) of them at a time, much similar to relaxation (point-by-point minimization) in deterministic models. But instead of lowering $E(x)$ as much as possible in each such step, the MC step *simulates* $P_E(x)$, i.e., each choice of a new value x'_j to replace a current value x_j is decided randomly by an *ergodic* procedure that keeps P_E *stationary*. "Ergodic" means that every configuration x can (with positive probability) eventually (in a finite number of sweeps) be reached. "Stationary" means that *if* before the step the probability distribution happens to be P_E, then after the step the distribution will still be P_E. A usual way to obtain stationarity of P_E is by maintaining *detailed balance* at each step, that is, satisfying for any pair of configurations x and x' the condition

$$P_E(x) P(x \to x') = P_E(x') P(x' \to x) \ , \tag{1.52}$$

where $P(x \to x')$ is the *transition probability* of the step, i.e., the probability of obtaining x' after the step given that the configuration before the step is x. A general simple way to change one x_j while maintaining detailed balance is by the *Metropolis* procedure [Metropolis *et al.*, 1953], consisting of the following two steps:

(i) Let the current configuration be $x^\alpha = (x_1^\alpha, \ldots, x_n^\alpha)$; choose a *candidate* $x^\beta = (x_1^\beta, \ldots, x_n^\beta)$ to replace x^α, where $x_i^\beta = x_i^\alpha$ for $i \neq j$, and x_j^β is chosen in a *symmetric* way; i.e., the probability of choosing x_j^β to have a certain value γ given that x_j^α has the value γ' is the same as that probability with the roles of γ and γ' interchanged. For example, choose $x_j^\beta = x_j^\alpha + \delta$, where δ is a random number uniformly distributed in a symmetric interval $[-q, q]$.

(ii) If $E(x^\beta) \leq E(x^\alpha)$ then x^β is *accepted*, i.e., it replaces x^α as the next configuration of the MC chain. If however $E(x^\beta) > E(x^\alpha)$, then x^β is only accepted in probability ρ, where

$$\rho = P_E(x^\beta)/P_E(x^\alpha) = e^{E(x^\alpha) - E(x^\beta)/k_B T} \ , \tag{1.53}$$

while in probability $1 - \rho$ the candidate x^β is *rejected*, i.e., x^α remains to serve as the next configuration.

Note that if the Jacobian $\partial^2 E / \partial x_i \partial x_j$ is sparse, the calculation of $E(x^\alpha) - E(x^\beta)$, and hence of ρ, is inexpensive. For efficiency, the size q

above is adjusted so that acceptances and rejections are roughly equally frequent.

Monte Carlo slowness. By the theory of *Markov chains*, if the MC chain of configuration $x^{(1)}, x^{(2)}, \ldots$ is ergodic and stationary (e.g., by employing Metropolis), then $\lim_{k\to\infty}\text{Prob}(x^{(k)} = x) = P_E(x)$, and hence $\lim_{K\to\infty}\frac{1}{K}\sum_{k=1}^{K} O(x^{(k)}) =< O >$, as desired. However, convergence can be painfully slow, because many successive configurations in the chain are very similar to each other, so they do not add new statistics to average out deviations from the average. By the Central Limit Theorem of probability, the error $< O > -\frac{1}{K}\sum_{k=1}^{K} O(x^{(k)})$ is proportional to $i_K^{-1/2}$, where i_K is the number of essentially *independent* samples in the chain $x^{(1)}, x^{(2)}, \ldots, x^{(K)}$. Many MC sweeps are usually needed to produce one new independent configuration. This MC slowness (called Critical Slowing Down (CSD) in statistical mechanics of systems in critical temperatures) has a nature similar to the relaxation slowness (cf. Introduction): Due to the localness of the MC process (e.g., treating one x_j at a time), large-scale features (e.g., smooth components) in x are slow to change (requiring many MC sweeps to sample a new amplitude). In addition, and potentially more serious, again similar to the deterministic case, there is the problem of *attraction basins* of the energy functional. Although unlike relaxation the MC process is never absolutely stuck in a basin since it assigns a positive probability even to energy-increasing steps, in practice the probability to accumulate enough local steps to escape a large-scale attraction basin can be exponentially small, and the MC process will fail to sample other basins. In fact, in non-deterministic models, the distinction is blurred, between the slowness of sampling large-scale features and the slow switching between large-scale attraction basins. Both are very detrimental to the MC statistics, since it is exactly the *largest* fluctuations in $O(x)$ which are usually associated with the largest-scale changes in x, so they are slowest to be averaged out.

Cluster methods. A well-known early technique to eliminate or reduce the critical slowing down of certain models in statistical mechanics is the *cluster method*, introduced by [Swendsen and Wang, 1987] and further developed by [Wolff, 1989]. The approach is to enhance the MC simulation by constructing large-scale changes that maintain detailed balance. The method has revolutionized the computational investigation of the relevant models (e.g., the Ising spin model and models in which the Ising model can be embedded), but it is not general enough and cannot attain statistical optimality (defined below).

Interpolation-based multiscale methods. Similar to the Galerkin coarsening of deterministic problems (cf. Sec. 3), the energy functional $E(x)$ can automatically be defined on increasingly coarser levels by recursively specifying, level after level, coarse-to-fine interpolation rules. Also, the same type of multigrid (or algebraic multigrid) cycles can be used, except that the relaxation sweeps are replaced by MC sweeps. The cycle index γ (see Sec. 2) in such statistical multigrid algorithms, and the amount of MC passes in each region at each level, can be chosen so that changes associated with larger fluctuations of the calculated observable O are sampled suitably more often.

Different versions of interpolation-based multigrid MC algorithms were independently introduced by a number of authors [Brandt *et al.*, 1986], [Goodman and Sokal, 1986], [Mack and Pordt, 1988]. In [Brandt, 1992], [Galun, 1992] and [Brandt *et al.*, 1994] it was shown that such algorithms not only accelerate the MC simulation at the fine level, but can also very cheaply average-out large-scale fluctuations by calculating many samples at the coarsest levels of the multigrid hierarchy (employing higher cycle indices). In fact, for *infinite-lattice Gaussian models*, i.e., models with quadratic energy functionals $E(x)$ and $n \to \infty$, it has been shown in [Galun, 1992], [Brandt *et al.*, 1994], [Brandt and Galun, 1996], [Brandt and Galun, 1997] that the algorithms are "*statistically optimal*", i.e., they calculate $< O >$ to accuracy ε in just $O(\sigma^2\varepsilon^{-2})$ computer operations, where $\sigma =< (O- < O >)^2 >^{1/2}$ is the standard deviation of the observable O. This achieves for observables associated with infinitely extended systems the same order of computer work as needed to calculate, by statistical sampling and to a similar accuracy, any simple "pointwise" average, such as the frequency of "heads" in coin tossing.

Less successful has been the extension of the interpolation-based approach to non-Gaussian models, although partial reductions of critical slowing down were demonstrated [Galun, 1998], [Brandt and Galun, 1998], [Shmulyian, 1999]. This led to the multiscale approach described next, which is much more generally applicable.

Renormalization multigrid (RMG). The RMG method combines multigrid techniques with the coarsening approach of the renormalization group (RG) method developed in theoretical physics (e.g., [Wilson, 1983] and [Fisher, 1998]) and other multiscale ideas. It has already yielded statistically optimal performance for models in statistical mechanics (see [Brandt and Ron, 2001]) and was successfully applied to diverse models, from simple liquids [Brandt and Iliyn, 2000] to simple macromolecules [Bai and Brandt, 2000]. The following is a brief summary of its principles. (Compare it to "systematic upscaling" in the deterministic case – in Sec. 7.)

Coarse level variables are each defined in terms of a small local set of next-finer-level variables. Examples: (i) If the next finer level consists of a discrete function (e.g., defined on a grid of points) each coarse level variable may be a weighted local average of several fine variables, or just the *sign* of such an average (e.g., in Ising spin problems), etc. (ii) If the fine variables represent atom positions of a macromolecule (e.g., polymer), the coarse variables may represent positions of pseudo atoms, each located at the mass center of several fine atoms. (iii) If the fine variables are atom positions of a fluid, the coarse variable can be defined on a grid, where the values at each gridpoint represent local fluid statistics, such as its total mass in a cell around the gridpoint, its total dipole moment, etc., depending on the context. And so on. Although the choice of coarse variables is not unique and requires physical insight, there is a general criterion (presented below) to decide whether a given choice is adequate.

Compatible Monte Carlo (CMC) is a MC process on the fine level which is restricted to the subset of fine-level configurations compatible with (i.e., whose coarsening coincides with) one given, fixed coarse-level configuration. The *equilibration* (or *decorrelation*) *time* of CMC is the number of sweeps the CMC needs to obtain a fine-level configuration which is essentially independent of its initial (compatible) fine-level configuration. A *consistently fast* CMC equilibration (i.e., CMC with very short average decorrelation time, averaging being over an ensemble of the fixed coarse configuration) implies that the fine-level equilibrium can be produced from the coarse-level equilibrium just by a short local processing.

A general measure for the adequacy of the set of coarse variables is the average speed of CMC equilibration. A similar *deterministic* measure, the speed of compatible *relaxation*, is defined in Sec. 3 above. Analogously to that case, a consistently fast CMC equilibration entails the *near locality* property, implying that the *conditional* probability distribution of any coarse variable, given fixed states of all other coarse variables, depends mainly on its closest neighborhood, with exponentially weak remnant dependence on coarse variables outside that neighborhood. This makes it possible to derive, just by local simulations, coarse-level *transition probabilities*, in terms of which effective coarse-level MC simulations can then be conducted.

The coarse-level transition probabilities are defined either in terms of a branching table of conditional probabilities, obtained by accumulating statistics during fine-level simulations (as in [Brandt and Ron, 2001]), or in terms of a coarse-level Hamiltonian, obtained iteratively by comparing statistics of coarse simulations with fine ones (as in [Bai and Brandt, 2000]).

Interpolation. Whenever desired, any coarse level configuration can quickly be "interpolated" to the fine level, by running to equilibrium a corresponding CMC.

Multilevel coarsening. With MC simulations now available at the coarse level, derivation of the *next coarser* level, first its variables and then its transition probabilities, can be conducted in a similar manner. A sequence of increasingly coarser levels can be derived in this way, leading for example to a fixed point (see below) or to *macroscopic simulations* of a material whose microscopic description was given. Usually this microscopic description was given in the form of highly *repetitive* laws, so that simulations at fine levels can be restricted to small representative regions. The form of the emerging macroscopic "equations" (or transition probabilistic rules) comes out purely numerical, which offers much greater generality than closed-form analytical expressions.

Slowness-free Monte-Carlo simulations at all levels of this system comes naturally: to obtain a new independent equilibrium on a given finite domain, an equilibrium is first obtained on that domain at a coarse enough, inexpensive level, then an equilibrium configuration is interpolated to the next finer level (by fast-equilibrating CMC), and so on until the target level is reached. At each interpolation level, if the coarse level transition probabilities are not fully accurate, the CMC equilibration should be followed by a small number of regular MC sweeps, a process called "*post relaxation*".

A particular advantage of this equilibration process is the ability to cheaply produce very far regions of the same equilibrium configuration, without having to produce (at the fine levels) all the regions in between. This yields very efficient ways to calculate *far correlations*.

Windows. In fact, usually one should not produce full configurations of the fine levels; rather, only some representative windows of increasingly finer levels will be generated, concentrating on regions from which more fine-level statistics is needed.

Fast iterations. Since the derivation of the coarse transition probabilities depends on efficient simulations at the finer levels, which in turn depend on the coarser levels for fast equilibration (or for supplying a rich enough collection of windows), iterating back and forth between the levels is in principle needed. However, these iterations converge very fast, since only local equilibration inside representative windows is needed at each level. The entire multilevel system settles quickly into a self-consistent equilibrium.

Fixed-point calculations. In many systems of statistical mechanics there exists a *critical temperature* T_c below which the sequence of coarsening (renormalization) steps tend to a fully deterministic system, and above which it tends to a completely random system. The calculations of most interest are at T_c, where all scales of the problem strongly interact with each other and the successive coarsening (the renormalization group) tends to an interesting *fixed point*. Supplemented with "criticalization" projections to curb divergence away from the critical surface, the RMG techniques can very efficiently calculate the fixed point and associated quantities, such as the critical exponents (see [Brandt and Ron, 2001], [Ron *et al.*, 2002a], [Ron *et al.*, 2002b]).

Non-local interactions and non-equilibrium systems. The RMG techniques can be extended to non-local (e.g., electrostatic) interactions, by transferring directly to a coarse level the smooth part of those interactions (as in Sec. 6 above), so that only their local part remains to be expressed at the coarse level, which can be done by the local procedures presented in this section. Also, these techniques are expandable to non-equilibrium systems. The conditional-probability representation of such systems can be coarsened in both space and time, at various space/time coarsening ratios, yielding long-time and large-scale dynamics of the system.

Low temperature algorithms. Attractions basins are particularly inefficiently sampled at low temperatures. However, the multigrid algorithm can efficiently get into equilibrium even at low temperature by employing an *adaptive annealing process*. In this process the temperature is reduced step by step. At each step, upon reducing the temperature from a previous T to a new T', a first approximation to the transition probabilities of T' at all levels are obtained from those of T (by applying the same Hamiltonian or by raising each term in the conditional probability tables to the power T/T'). Then, in just few multilevel cycles, the transition probabilities can be made accurate — provided the quality of the set of coarse *variables* has not been deteriorated.

Indeed, the type of coarse-level variables appropriate at low temperatures does generally differ from that at high temperatures. In calculation of fluids, for example, at high temperatures the average density is an adequate coarse-level variable. At low temperatures, e.g., at the appearance of liquid drops in a gas or at the onset of piecewise crystallization, other coarse-level variables should be added, such as the average crystal direction, and/or the average density of holes, and/or the location of mass centers. Thus, in the annealing process one should monitor the quality of coarse variables by occasionally checking the CMC equilibration speed at all levels. When this speed starts to deteriorate at some level, additional variables should be added at that level, with a corresponding extension of the transition probability tables (e.g., adding terms to the Hamiltonian). Candidate new variables can be found by physical understanding and/or by suitably blocking highly-correlated variables at the next-finer level; then the new variables should be admitted provided they pass the CMC-equilibration-speed test. Some of the old variables may be removable, as judged again by CMC equilibration tests.

In fact, unlike the classical *simulated annealing* method (see Sec. 10), the chief purpose of annealing here is the gradual *identification* of the degrees of freedom that should be employed at increasingly coarser levels. At the zero-temperature limit these procedures can also yield powerful multiscale minimization procedures, as described later in the next section.

10. Global optimization: Multilevel annealing

A general method to escape false attraction basins is to replace the strict point-by-point minimization by a process that still accepts each candidate change which lowers the energy ($\delta E < 0$), but also assigns a positive probability, proportional for example to $\exp(-\beta \cdot \delta E)$, for accepting a candidate step that *increases* the energy ($\delta E > 0$). This is similar to a Monte Carlo simulation of the system at a finite temperature T, where $\beta = (k_B T)^{-1}$ and k_B is the Boltzmann constant. This is indeed the very way by which natural materials escape various attraction basins and advance toward lower energies (cf. Sec. 9).

To have a reasonable chance to escape *wide* attraction basins or basins within *high* energy barriers, in a tolerable amount of computational time, a low value of β, or a high temperature, must of course be applied. This however makes it improbable to hit the true minimum. A general approach therefore is the *gradual* decrease of temperature, hoping first to escape false high-energy attraction basins, then lower-energy ones, etc. This process is called *simulated annealing* [Kirkpatrick *et al.*, 1983], since it simulates the common industrial process of "annealing" — obtaining low-energy materials (such as less brittle glass) by careful gradual cooling. Variations on the theme include various procedures of alternate heating and cooling.

The simulated annealing algorithms are extremely inefficient for many physical problems, requiring exponentially slow temperature decrease to approach the true minimum. This is usually due to the multiscale structure of the attraction basins: small-scale basins reside within larger-scale ones, which reside within still-larger-scale ones etc. The small-scale basins correspond to *local* structures in the physical space; larger-scale basins correspond to larger physical structures. When the temperature is high enough to enable transition between large-scale attraction basins it would completely randomize finer-scale features, even when they have already settled into low-energy local structures (by a previous cooling).

In such cases, the transitions between basins at various scales should be better coordinated. It should employ much lower temperatures in switching between large-scale basins, which can be achieved only if well orchestrated large-scale moves are constructed. This is done by what we will generally call "*multilevel annealing*", whose main features are described below. Its first, incomplete version appeared in [Brandt *et al.*, 1986].

In multilevel annealing, the main role of the gradual cooling is to *identify* increasingly larger-scale degrees of freedom that are acceptable to simulation at progressively lower temperatures. There are two approaches to go about it, one in terms of coarse-level variables and the other in terms of large-scale moves.

Coarse level variables These are variables that are coupled to each other through temperature-dependent conditional probability (CP) tables, as in the RMG method (cf. Sec. 9). Gradually, as the temperature is lowered, new coarse-level variables are generally introduced, checked by the CMC-equilibration test. The procedure is like that of Monte Carlo simulation at low temperatures (described in Sec. 9), except that it can be executed without strict adherence to statistical fidelity ("detailed balance"). In many cases a low-temperature-like simulation is actually more realistic than strict minimization, either because the minimization task is fuzzy anyway (see Sec. 12), or simply because the material whose minimal energy is sought has in reality a *finite* temperature.

Note the similarity of this procedure to the BAMG approach in Sec. 3, in which increasingly coarser (large-scale) variables and interpolation rules associated with increasingly lower eigenvalues (corresponding to lower temperatures here) are gradually revealed, through a process that uses coarser levels already accessible by the current interpolation rules to accelerate relaxation (or the Monte Carlo simulation here) at finer levels.

Large-scale moves. Note that in the approach just described, each coarse level configuration corresponds to the *equilibrium* of all fine-level configurations that are compatible with it. When the temperature is lowered, this equilibrium narrows down to the vicinity of few specific fine-level configurations. Another approach then is to work explicitly with the fine level, and to identify on it increasingly larger-scale *moves* that can be done with progressively lower temperatures. If an efficient simulation has already been obtained at some temperature T, it can be employed to identify suitable moves for a lower temperature T', assuming $T - T' \ll T'$. Indeed, the moves already identified for T are at a scale close to those required for T', hence each suitable T'-move is approximately a linear combination of just a small number of T-moves. Such combinations can be identified by calculating correlations between neighboring T-moves during Monte Carlo simulations with the temperature T. Each combination can then be "revised" into more precise T'-move by optimizing around it (see below).

The work in terms of large-scale *variables* is perhaps preferable whenever the system is *highly repetitive*, so that the same coarse-level variables and CP tables can be used at all (or many) subdomains, as in the case of atomistic systems. The tables then can be derived in just representative small *windows* of the fine-scale system (see the description of windows in Sec. 9). On the other hand, the identification of *explicit large-scale moves* is often more practical for systems that have different specific structures at different neighborhoods, making it too expensive to derive place-dependent CP tables. However, the explicit moves are not flexible enough, requiring the device discussed next.

Nested revision algorithm. Any preassigned large-scale move is likely to bring about a substantial energy increase since its fine details would not generally quite fit the fine details produced by other large scale moves. In other words, in switching to a new large-scale attraction basin one does not generally immediately hit the lowest-energy configurations of that basin; since in the *previous* basin a process of minimization *has* already taken place, the new configuration is likely to exhibit a much higher energy. Thus, only rarely the large-scale move will be accepted in a low-temperature simulation, even if the new attraction basin does harbor lower energy configurations. Therefore, before applying the acceptance test to a large-scale move, one should "revise" (or "*reshape*") the move, or "optimize around it", by employing in the neighborhood around it a Monte Carlo simulation of smaller-scale moves. Each of these smaller-scale moves may itself need "revision" by local simulations around it at still finer scales. And so on. Such nested revision processes are needed when the energy landscape has nested attraction basins. Each of these processes generally should itself employ a kind of annealing (see details in [Brandt *et al.*, 1986]).

Working with the difficult discrete optimization problem of spin glasses, it was shown already in [Brandt *et al.*, 1986] that such multiscale nested optimization techniques (together with the taming technique discussed below) work reasonably well even *without* any prior identification of specialized moves at all scales. However, the amount of work in that case turned out to increase at least quadratically as a function of the number of spins in the system, due to the excessive nested revision processes that were required. Much shorter revision procedures will suffice with more specialized moves. Also, as mentioned above, the revision procedure can be used to optimize the specialized moves themselves, prior to their use in the T' simulations.

Note that the revision procedure (unless confined only to the prior identification of moves) does not satisfy the statistical detailed balance. It is very efficient in the search for a minimum, but cannot be used for obtaining accurate finite-temperature statistics.

Revision process can of course be very beneficial also in the annealing process at each *single* level of the multiscale algorithm. Namely, any attempted move can be followed by some neighboring moves *at the same level*, which revise the move so as to lower the energy as far as possible, before deciding whether to accept the (revised) move.

Taming local excitations. In any sufficiently large-scale problem with local couplings (i.e., its objective functional is a sum of terms each of which depends only on a *local* set of variables, in some low dimensional space), there is a large accumulation of likelihood that any stochastic simulation, even with a low temperature, will always have somewhere some small-scale local excitations (fluctuations away from the minimum), frustrating the chance to identify *the* global minimum. Since these excitations are indeed likely to be local, one can eliminate them by the following simple procedure.

Keep in memory one or several of the best-so-far (BSF) configurations. Once in a while (e.g., whenever the stochastically-evolving current configuration yields a particularly low energy) compare the current configuration with each of the BSF configurations. The two compared configurations will generally have spots of just *local* disagreement, i.e., *disconnected* (or nearly disconnected) subsets where the values of the two configurations differ, but outside which the configurations coincide. Hence, for each such subset, separately from all other subsets, one can decide whether or not to replace the BSF values by those of the current configuration, depending which option would yield at that spot the lower energy. Before deciding, if the subsets are not fully disconnected, one should of course relax around the replaced values, to lower the energy as far as locally possible (somewhat similar to the "revision" processes described above). From the two configurations, their so called *Lowest Common Configuration (LCC)* is thus produced. In this way all the BSF configurations can be replaced by better ones. The current configuration should continue its evolution from its previous value, in search for new optima. At the end, the BSF configurations can be compared to choose the best among them.

This device should apply not only to the main optimization process, but also to each of the auxiliary "revision" processes defined above, as successfully demonstrated in [Brandt *et al.*, 1986] and [Ron, 1989].

Population algorithms. Analogous devices can be used even for more general problems (not just locally coupled). The general approach can be described as a combination of multilevel annealing with genetic-type algorithms. In the case of processing in terms of *large-scale moves* (as described above), instead of one minimization process, a population of such processes evolve in parallel. Once in a while one of the evolving configurations (a "parent") chooses another (a "partner"), from which it borrows a combination of large-scale moves, revising them using its own finer multiscale moves, then (and only then) deciding whether to adopt the resulting configuration (accept it as an addition to the population or as a replacement). Each of the revision processes can itself be done in terms of several evolving children, and so on recursively. "Fitness" parameters can be defined in terms of the low-energy levels attained by the evolving configuration and its relatives. The choice of "partner" can be based on its fitness and criteria of compatibility with the choosing "parent".

In the case of processing in terms of *coarse-level variables*, processing at very coarse levels is very inexpensive, allowing the use of much larger populations of parallel processes there. Upon each switch to a finer level, the population can be sharply reduced, by repeatedly replacing a pair of *similar* configurations by their LCC. (For each configuration, a list of similar configurations can inexpensively be prepared at the coarsest level, then transferred to (and trimmed down at) finer levels, where progressively stricter "similarity" criteria would be applied.)

In short, one can marry the ideas of multiscale optimization with those of genetic algorithms and study the fitness of their evolving offsprings... The success is likely to be especially high for problems dominated by a multitude of local couplings.

11. Graph and hypergraph problems

Notation. A *hypergraph* is a pair of sets V and $\mathcal{E}$, where V is a set of $n = |V|$ *vertices*, or *nodes*, denoted $V = \{1, 2, 3, \dots, n\}$, and $\mathcal{E}$ is a set of subsets of V, called *hyperedges*. An (undirected) *graph* is the special case where each subset includes only two vertices, say i and j, forming a simple *edge*, or *link*, denoted $\{i,j\}$. In a *weighted* (hyper)graph, each (hyper)edge has a weight, and also to each vertex i there may be assigned a *volume* (or *area*, or *length*, depending on context), denoted v_i. In a simple graph, the weight of the edge $\{i,j\}$ will be denoted w_{ij}. The non-weighted hypergraph can be regarded as the special case where $v_i = 1$ ($i = 1,\ldots,n$) and all hyperedge weights are also 1. In the algorithms below, even if the original problem is non-weighted, the emerging coarse-level problems are generally weighted.

Optimization problems. There are many kinds of graph/hypergraph optimization problems. The following are some popular examples.

(1) *Linear arrangement (minLA)*: Find a permutation φ of the graph nodes such that $\sum_{i,j} w_{ij}|\varphi(i)-\varphi(j)|$ is minimal. The generalized form of this problem is to minimize $\sum_{i,j} w_{ij}|x_i - x_j|$ where $x_i = \sum_{\varphi(j)\le\varphi(i)} v_j$. The minLA problem is a typical example of *graph layout problems*; see [Díaz *et al.*, 2002] for a survey. It is also related to the next kind of problems.

(2) *Low-dimensional graph embedding*: For each node i assign a d-dimensional vector $\vec{x}_i$ such that the Euclidean distances $r_{ij} = \| \vec{x}_i - \vec{x}_j \|$ will minimize an objective functional such as $\sum_{\{i,j\}\in\mathcal{E}}(r_{ij} - w_{ij})^2$. Here each node may represent some high-dimensional data point and the edge weights represent distances in that high-dimensional space. Often such distances are given only between close neighbors. The dimension d is low, so that the positions $\vec{x}_i$ yield good parameterization or visualization of the large multivariate data set, or reveal similarity between such sets. (see [Cox and Cox, 1994]). The problem of drawing a general graph in some low dimension was formulated as such in [Hall, 1970]. Another promising possible objective, the locally linear embedding proposed in [Roweis and Saul, 2000], is $\sum_i \| \vec{x}_i - \sum_{(i,j)\in\mathcal{E}} W_{ij}\vec{x}_j \|$, where $\{W_{ij}\}_j$ is the set of weights for representing the data point i as a weighted average of some neighboring data points j.

(3) *Graph/hypergraph partition* is the problem of dividing V into a (usually small) number of disjoint subsets $R_1, R_2, \ldots, R_K$, called *regions*, each having total volume between two given bounds $v^{\min}$ and $v^{\max}$, so that the total weight of all inter-region hyperedges (*cutsize*) is minimal. The image segmentation problem can be regarded as a special case (see Sec. 12). For other major variant formulations of the partitioning problem see [Alpert and Kahng, 1995].

We will not discuss here for any of these problems theoretical complexity issues, such as lower and upper bounds for their solution cost. We are not interested here in the worst possible cases, which are extremely non-representative. Our focus is on *practical high-performance* algorithms, such that in most practical cases would yield a good approximation to the optimum at low computational cost. Typically, the multilevel algorithms exhibit linear complexity, i.e., the computational cost in most practical cases is proportional to $|V| + |\mathcal{E}|$.

To achieve such high practical performance, many of the optimization strategies discussed above, throughout this article, can be helpful, as will be outlined below.

AMG eigensolvers. For various graph problems, practical high performance algorithms have in the past taken the form of *spectral algorithms*, meaning algorithms based on calculating several of the lowest eigenvectors of the so called *graph Laplacian* A, whose terms are defined by

$$a_{ij} \;=\; \begin{cases} -w_{ij} & \text{for} \quad \{i,j\} \in \mathcal{E} \;,\; i \neq j \\ 0 & \text{for} \quad \{i,j\} \notin \mathcal{E} \;,\; i \neq j \\ \sum_{k \neq i} w_{ik} & \text{for} \quad i = j \end{cases} . \tag{1.54}$$

It has been shown by [Hall, 1970] that the second eigenvector of A gives the optimal solution for a real-valued quadratic formulation similar to minLA, i.e., minimize $\sum_{i,j} a_{ij}(x_i - x_j)^2$ subject to the constraints $\sum_i x_i^2 = 1$ while $\sum_i x_i = 0$. An approximation to the minLA ordering is the order of vertices in that eigenvector. Graph drawing on a plane can be similarly carried out by using the second and third eigenvectors of A. Extensions to partitioning and other applications are surveyed in [Alpert and Kahng, 1995].

For most graphs appearing in practice, the AMG-EIS eigensolver (see Sec. 8) is very efficient and has linear complexity. There are exceptions, such as random graphs and other expanders, but for them the spectral approximation is not likely to be good anyway, nor is the optimal solution likely to be significantly different from many arbitrary others. For most problems, since the graph Laplacian is a zero-sum M matrix, the simpler, *classical* AMG solvers will usually be suitable. (Parts of such an AMG eigensolver were used for graph drawing and minLA in [Koren *et al.*, 2002] and in [Koren and Harel, 2002].) However, the natural correspondence between the BAMG solver and the eigenproblem would possibly make it a more appropriate and certainly more general choice.

Multigrid paradigm. Rather than using it for solving a related eigenproblem, the general multigrid paradigm, combining recursive coarsening and relaxation at each level, can be *directly* applied to the graph problems. In this paradigm, the general purpose of each *coarsening* is to create an analogous but substantially smaller graph problem, whose solution would yield an approximate (or improved) solution to the given graph problem.

The *relaxation* in this paradigm has actually two different roles, sometimes entailing two different kinds of schemes. The first is to improve the approximate solution obtained after returning from the coarse level, a process called *post-relaxation*. The second is to help prepare the coarsening by yielding a good coarse-to-fine correspondence, for example, through smoothing the error, as in classical multigrid, or by supplying test vectors to define good coarse-to-fine interpolation, as in BAMG (see Sec. 3). This process is called *pre-relaxation*.

The multigrid paradigm has been used successfully for drawing graphs by [Koren *et al.*, 2002], for the graph/hypergraph partitioning problem (see [Karypis, 2002], [Walshaw, 2002] and references therein) and for other related problems discussed in this book. Essential aspects of applying this paradigm to graph problems are discussed below.

Weighted aggregation. The AMG coarsening (Sec. 3) will be interpreted here as a process of *weighted aggregation* of the graph nodes to define the nodes of a coarser graph. In a *strict* aggregation process (also called edge contraction or matching of vertices) the nodes are blocked in small disjoint subsets, called *aggregates*. Two nodes i and j would usually be blocked together (put in the same aggregate) only if their coupling is *strong*, meaning that w_{ij} is comparable to $\min\{\max_k w_{ik}, \max_k w_{kj}\}$. In *weighted* aggregation, each node can be divided into *fractions*, and different fractions belong to different aggregates. In both cases, these aggregates will form the nodes of the *coarser level*, where they will be blocked into larger aggregates, forming the nodes of a *still coarser level*, and so on. As AMG eigensolvers have shown, *weighted*, instead of *strict*, aggregation is important in order to express the *likelihood* of nodes to belong together; these likelihoods will then accumulate at the coarser levels of the process, automatically reinforcing each other where appropriate. Strict aggregation, by contrast, may run into a conflict between the local blocking decision and the larger-scale picture.

The rules for creating a weighted aggregation of a graph can indeed be based on AMG principles, as follows. Each aggregate is *seeded* by one fine-level node, where the set C of all seeds is a subset of V chosen so that the compatible relaxation of the graph Laplacian has a fast convergence rate (see Sec. 3). Then, the fraction P_{ij} of a node i that belongs to an aggregate j can be chosen by the rule (1.16). To curb complexity, the number of fractions of each node should be restricted, either directly or by a proper choice of the thresholds $\alpha_i^{(\nu)}$.

A weighted aggregation can also be designed for *hypergraphs*. The seeds and fraction sizes can in this case be determined by a pre-relaxation process, as described below.

The coarse graph. Having determined the aggregates, which are the *nodes* of the coarse graph, the associated coarse *minimization problem* should be formulated so that upon disaggregation its minimum will approximate the minimum of the original (or the next-fine-level) problem. This formulation is usually quite straightforward.

For example, in the (hyper)graph partition problem, the coarse problem is again a partition problem, with the same $v^{\min}$ and $v^{\max}$ constraints, defining the volume of each coarse node I to be $\sum_i v_i f(i,I)$, where $f(i,I)$ is the fraction of the fine-node i that belong to I. Similarly, the edge that connects two coarse aggregates I and J is assigned with the weight $\sum_{i,j} w_{ij} f(i,I) f(j,J)$. In the image segmentation problem these weights are subsequently modified (see Sec. 12).

In minLA, the coarse problem will be a *generalized* minLA, with coarse "volumes" (here actually standing for *lengths*) and weights defined as in the partition problem.

Disaggregation. Having solved the coarse problem, an approximate solution to the original (or the next-finer-level) problem is obtained by *disaggregation*, the analog of the coarse-to-fine multigrid interpolation. This approximation is subsequently improved by several *post-relaxation* sweeps (first compatible, then regular: see below).

The design of disaggregation is quite straightforward, too. For example, in the partition problem, having obtained a partition of the aggregates, a simple disaggregation process is to put fine-level node i in region R_k in probability $\sum_{I \in R_k} f(i,I)$. If the obtained partition is infeasible (violate the volume constraints), it will be made feasible by the post-relaxation. In minLA, the disaggregation involves two steps: first one positions node i along the real line at $y(i) = \sum_I f(i,I) x_I^c$, where x_I^c is the position of aggregate I. Then the position is changed to $x_i = \sum_{y(j) \le y(i)} v_j$, thus retaining order but taking volume (length) into account.

This simple disaggregation is not likely to be accurate enough, though. It should therefore be followed by several sweeps of *compatible* relaxation, which is like the post relaxation described below, but avoids changing the positions of the seeds (cf. the compatible relaxation in Sec. 3). This will produce much improved disaggregation, still compatible with the coarse solution, before changing it by *regular* post relaxation.

Post relaxation, since done in the multilevel framework, can be restricted to just *local* exchanges of nodes, each exchange being aimed at lowering the objective (energy) functional, possibly with added stochasticity to "oil" the passage to lower energies (see "annealing" below). By "local" exchanges we will generally mean switching or adjusting positions only between several neighboring or nearly neighboring nodes, where neighborhood relations are determined in some natural way: in minLA it is determined by the current linear arrangement; in the partition problem – by the topology induced by the current hierarchical aggregation.

Pre-relaxation is a similar *local* process, but with a possibly different objective functional and, especially, different constraints. For example, in the *hypergraph* partition problem, the objective functional can remain the same, but the volume constraints are much smaller , e.g., $v^{\max} = 4$ and $v^{\min} = 1$, so that the aim is partitioning into a host of small regions, which we will call *clusters*. Once optimized, at least locally, these clusters can be used as the aggregates of a *strict* aggregation. Alternatively, during the sweeps of pre-relaxation one can count the number of times any two nodes, i and j, share the same cluster, and use these counts as the coupling a_{ij} serving in a *weighted* aggregation. (This is somewhat similar to finding interpolation weights from relaxation in Bootstrap AMG, as in Sec. 3, or to the identification of large-scale movements, as in Sec. 10). Note that this approach is applicable when there is no Laplacian, as in the case of *hypergraphs*, or when the Laplacian is not very useful for obtaining weights, as in the case of very *aggressive* coarsening (large $|V_{\text{fine}}|/|V_{\text{coarse}}|$, sometimes needed to reduce the overall complexity).

Linearizations. The graph Laplacian yields a good coarsening (the AMG coarsening) when the problem is associated with, or approximated by, the problem of minimizing the quadratic functional $\sum_{i,j} w_{ij}(x_i - x_j)^2$. A better quadratic formulation to a non-quadratic minimization problem can usually be obtained in terms of a *current approximation*, in the spirit of Newton linearization (see Sec. 7). The main property of such an approximate quadratic formulation is *stationarity*, i.e., the quadratic formulation will reproduce the current approximation if the latter happens to be already the solution to the *original* (non-quadratic) problem. For example, given a current approximation $\{\widetilde{x}_i\}$, a stationary quadratic approximation to the (generalized) minLA problem is

$$\text{minimize} \qquad \sum_{i,j} \frac{w_{ij}}{|\widetilde{x}_i - \widetilde{x}_j|^\alpha}(x_i - x_j)^2 \ , \qquad \text{with } \alpha = 1 \ . \qquad (1.55)$$

At each level of the multiscale minLA solver, several cycles to coarser levels can thus be performed, using first the original ($\alpha = 0$) quadratization, then in subsequent cycles increasing α toward 1. Using a certain value of α means here to employ $a_{ij} = w_{ij}/|x_i - x_j|^\alpha$ instead of the original Laplacian in forming the aggregation seeds and weights. Note, however, that (1.55) is stationary only for the real-number approximation to minLA, it is not stationary when the requirement that $(x_1, \ldots, x_n)$ has to be a permutation of $(1, \ldots, n)$ is added.

Multilevel annealing, revision and population processes. Both post- and pre-relaxation, at all levels, can benefit greatly from incorporating annealing processes, in which Monte Carlo simulations with progressively lower temperatures replace the deterministic relaxation. At each level the annealing can start at relatively high temperature, e.g., such that would give 90% acceptance to about half the local exchanges at that level. Then, at each subsequent Monte-Carlo sweep, the temperature should be lowered substantially, e.g., to 0.75 times its previous value. The localness of these exchanges and the rapid cooling guarantee the preservation of large-scale solution features inherited from the coarser levels. Repeated re-heating and cooling can be useful.

As the algorithm proceeds to ever more evolved configurations, further improvement may depend on employing nested revisions of each move that seems promising; that is, whether to accept the move is decided only after optimizing around it by additional moves either at the same level or at finer levels (see Sec. 10). Another alternative: If recursive revisions are not extensively used, processing at the coarsest level is very inexpensive, hence a large *population of parallel optimization* processes can be afforded there. Upon switching to a finer level, the population should be cut down by repeated LCC replacements for pairs of similar configurations (see Sec. 10), so as to keep the overall work at each level suitably bounded.

The multiscale paradigm, employing extensive annealing at all levels has been implemented in [Safro, 2002] for a sequence of model minLA problems. Although not yet employing other devices mentioned above, the test in nearly all cases have produced a solution with energy lower than that reported in [Petit, 1998] and [Koren and Harel, 2002].

12. Multilevel formulation

In many, perhaps most, global optimization problems, the objective functional E is not uniquely determined by direct physical laws, but is concocted, somewhat arbitrarily, to impart a precise meaning to a practical problem, whose original form is more fuzzy.

This, for example, is the formulation of ill-posed problems, like *inverse PDE problems* (system identification or data assimilation, as in Sec. 8).

The solution of such problems is often uniquely and stably fixed with the aid of *regularization*, which recasts the problem into a minimization task. The same is true in formulating *optimal control* problems, in which a performance index is added to the given mechanical system. In all these cases, the objective, or the sense in which one solution is considered to be better than another, is not exactly apriori given; it is *chosen*, with somewhat arbitrary form and parameters.

Another typical example is the problem of reconstructing pictures from blurred or noised data. It is often recast as the problem of minimizing an energy functional which is the sum of penalty terms, penalizing the reconstruction for various unwanted features, such as (i) its distance from the data; (ii) non-smoothness, except across lines recognized as "edges"; (iii) proliferation of such edges; (iv) non-smoothness of edges; etc. This combination of penalty terms creates a monstrous minimization problem, with many nested attraction basins at all scales. It is extremely difficult to solve — *and unnecessarily so*: The difficulty largely arises from taking too seriously a set of arbitrary choices. Indeed, the form and the numerical coefficients of the various penalty terms are quite arbitrarily chosen; a picture which is slightly better than another according to one choice may well be worse according to many other, equally reasonable choices.

More generally, unnecessary computational difficulties often arise from our tradition to cast fuzzy tasks into "stationary" formulations, that is, to define as a solution a configuration which satisfies (exactly or approximately) one well-defined criterion, such as minimizing a certain functional under specified constraints. A more universal, and often far easier way is to admit a solution which is just the end product of a suitable *numerical process*, not necessarily designed to satisfy, even approximately, any *one* governing criterion. In reconstructing pictures, for example, features like edges and segments can be captured very satisfactorily by very inexpensive processes (see below); the results may well fit our perception even *better* than the true or approximate minimizer of the objective functional mentioned above. Similarly, for many other fuzzy problems, a numerical process can inexpensively yield excellent solutions, whose only "fault" is our inability to say what stationary objective functional they (at least approximately) optimize.

While this may be fairly obvious, one can argue that the objective-functional formulation is still in principle the "true" one: if fully carefully chosen, it would precisely reflect what one would *want* to obtain, complicated or impractical as it may be. However, even this is often not the case: a numerical process can incorporate a host of driving directives that are *impossible* to include in one stationary criterion. *Ex-*

amples: (i) The process for detecting curved edges can employ different completion-field parameters at different scales (see [Sharon *et al.*, 2000a]). (ii) The process for detecting picture segments can introduce new affinities between emerging intermediate aggregates, based on their internal statistics (see below). (iii) In solving inverse PDE problems one can apply *multiscale* regularizations, which use different penalty terms at different scales, based on the fact that known properties of the solution, as well as both the nature of measurement errors and the size of numerical errors, all may be very dissimilar at different scales.

It can be seen from these examples that an important tool in *formulating* various problems is to have different, sometimes even somewhat conflicting, objectives at different scales of the problem. The multiscale processing is thus not just a method to accelerate convergence and escape false attraction basins (as discussed above), but can often also be essential for an improved *definition* of the problem.

Incidentally, even for *linear* problems multi-scale *formulations* are sometimes needed. An example is the case of wave equations with radiation boundary conditions: such conditions are most appropriately formulated at the coarsest levels of the wave/ray algorithm (see Sec. 5 above), while the differential equations themselves are discretized at the finest level.

VLSI placement obviously has quite fuzzy multiple objectives. For example, one objective (objective I) may simply sum the total length of connections, while another objective (objective II) may also take into account the frequency of using each connection. Objective II may be more important than objective I at *short* ranges. The multilevel algorithm can accommodate such a situation by applying objective II at the lower (finer) levels, while more heavily weighting objective I in defining increasingly-coarser-level hyperedges.

Picture segmentation. As a more detailed example, we briefly present the multiscale picture segmentation algorithm of [Sharon *et al.*, 2000b], [Sharon *et al.*, 2001]. The decomposition of a given picture into meaningful segments is a basic task in pattern recognition. The criteria for blocking two picture elements into the same segment include similarity in color levels, absence of separating edges, etc. Quantitatively, these can be expressed in terms of *coupling* coefficients between neighboring pixels, thus translating the picture into a graph. The iterative weighted aggregation of this graph (see Sec. 11) can be regarded as a hierarchical segmentation of the picture. Salient segments are those coarse-level aggregates whose external couplings (to other aggregates of the same level) are weak compared to their average internal couplings (a quantity that can be accumulated during the iterative aggregation process).

This segmentation process is very fast: only several dozen operations per pixel are invested before switching to coarse levels where the work is much smaller. More important, the multiscale weighted aggregation is free to apply new types of couplings at different levels. The coupling between larger-scale aggregates, instead of (or in combination with) being induced by the fine-scale couplings (as in the AMG process), they can employ new criteria. Such criteria can include for example similarity in the *average* color levels of the aggregates. More generally, all kinds of other intra-aggregate "*observables*" can be used: the aggregate's center of mass, its diameter, principal orientation, texture measures (being, e.g., averages and variances of sizes and orientations of sub-aggregates at various smaller-scale levels), etc., with the number of observables per aggregate increasing at coarser levels. Strong couplings should for example be established between neighboring aggregates which have similar texture measures. Strong affinity should also be assigned between two (not necessarily neighboring) aggregates whose principal orientations align with the direction of the line connecting their centers of mass; or between two neighboring aggregates whose boundaries seem to *continue* each other; etc. These kinds of couplings can be established even between quite *distant* aggregates, promoting the appearance of *disconnected* segments, presumably signifying partly occluded objects.

All these higher-level segmentation processes could not of course be reflected in a single objective functional. Our experiments with a series of real images show very successful segmentations of difficult examples. Not only the detected segments are much more accurate and realistic than could be produced by previous methods (that did not use multiscale principles in the aggregation process), but the computer run-time has been dramatically reduced from many minutes to just few seconds, even at the current research stage. Importantly, the *hierarchical* segmentation produced in this way is in a form directly usable by potential storage/retrieval recognition systems, since various statistics are accumulated from level to level and then normalized, so that each large-scale segment emerges with a vector of numbers representing textures, standardized shapes, sub-segments with their own vectors of numbers and other identifying features.

Acknowledgements. The research has been supported by grants No. 295/01 from the Israel Science Foundation, US Air Force Office Of Scientific Research contract F33615-97-D5405, US Air Force EOARD contract F61775-00-WE067 and by the Carl F. Gauss Minerva Center for Scientific Computation at the Weizmann Institute of Science.

References

Alpert, C.J. and Kahng, A.B. (1995). Recent directions in netlist partitioning: a survey. *INTEGRATION* the VLSI journal, 19:1–81.

Bai, D. and Brandt, A. (1987). Local mesh refinement multilevel techniques. *SIAM J. Sci. Stat. Comput.*, 8:109–134.

Bai, D. and Brandt, A. (2000). Multiscale computation of polymer models. In [Brandt *et al.*, 2000], pp. 250–266.

Bank, R.E. and Mittelmann, H.D. (1986). Continuation and multigrid for nonlinear elliptic systems. In: *Multigrid Methods, II*, Lecture Notes in Mathematics 1228 (Hackbusch, W. and Trottenberg, U., eds.), Springer, Berlin, pp. 23–37.

Bolstad, J.H. and Keller, H.B. (1986). A multigrid continuation method for elliptic problems with turning points. *SIAM J. Sci. Comput.*, 7:1081–1104.

Brandt, A. (1973). Multi-level adaptive technique (MLAT) for fast numerical solutions to boundary value problems. In *Proc.* 3^{rd} *Int. Conf. on Numerical Methods in Fluid Mechanics* (Cabannes, H. and Temam, R., eds.), Lecture Notes in Physics **18**, Springer-Verlag, pp. 82–89.

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

Brandt, A. (1980). Stages in developing multigrid solutions, in: *Proc.* 2^{nd} *Int. Congr. on Numerical Methods for Engineers* (Absi, E., Glowinski, R., Lascaux, P. and Veysseyre, H., eds.), Dunod, Paris, pp. 23–43.

Brandt, A. (1982). Guide to multigrid development. In *Multigrid Methods* (Hackbusch, W. and Trottenberg, U., eds.), Springer-Verlag, pp. 220–312.

Brandt, A. (1984). Multigrid Techniques: 1984 Guide, with Applications to Fluid Dynamics, 191 pages, 1984, ISBN-3-88457-081-1, GMD Studien Nr. 85. Available from GMD-AIW, Postfach 1316, D-53731, St. Augustin 1, Germany.

Brandt, A. (1986). Algebraic multigrid theory: The symmetric case. *Appl. Math. Comput.*, 19:23–56.

Brandt, A. (1991a). Rigorous local mode analysis of multigrid. In *Preliminary Proc.* 4^{th} *Copper Mountain Conf. on Multigrid Methods*, Copper Mountain, Colorado, April 1989. An updated version appeared as Weizmann Institute Report, 1991. The first part has appeared as [Brandt, 1994].

Brandt, A. (1991b). Multilevel computations of integral transforms and particle interactions with oscillatory kernels, *Comput. Phys. Comm.*, 65:24–38.

Brandt, A. (1992). Multigrid methods in lattice field computations. *Nuclear Phys. B Proc. Suppl.*, 26:137–180.

Brandt, A. (1994). Rigorous quantitative analysis of multigrid: I. Constant coefficients two level cycle with L_2 norm. *SIAM J. Numer. Anal.*, 31:1695–1730.

Brandt, A. (1997). The Gauss Center research in multiscale scientific computation. *Electr. Trans. Numer. Anal.*, 6:1–34.

Brandt, A. (1998). Barriers to Achieving Textbook Multigrid Efficiency in CFD. ICASE Interim Report No. 32, NASA/CR-198-207647. Appears as Appendix C in [Trottenberg *et al.*, 2000].

Brandt, A. (2000). General highly accurate algebraic coarsening schemes. *Electr. Trans. Num. Anal.*, 10:1–20.

Brandt, A. (2001). Multiscale Scientific Computation: Review 2001. In, Barth, T., Haimes, R., and Chan T., eds.: *Multiscale and Multiresolution Methods*, Springer Verlag. (Proceeding of the Yosemite Educational Symposium, October 2000).

Brandt, A., Bernholc, J., and Binder, K., eds. (2000). *Multiscale Computational Methods in Chemistry*, NATO Science Series, Computer and System Sciences, Vol. 177, IOS Press, Amsterdam.

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

Brandt, A. and Galun, M. (1996). Optimal multigrid algorithms for the massive Gaussian model and path integrals. *J. Stat. Phys.*, 82:1503–1518.

Brandt, A. and Galun, M. (1997). Optimal multigrid algorithms for variable-coupling isotropic Gaussian models. *J. Stat. Phys.*, 88:637–664.

Brandt, A. and Galun, M. (1998). Statistically optimal multigrid algorithm for the anharmonic crystal model, Gauss Center Report WI/GC–9.

Brandt, A., Galun, M., and Ron, D. (1994). Optimal multigrid algorithms for calculating thermodynamic limits. *J. Statist. Phys.*, 74:313–348.

Brandt, A. and Iliyn, V. (2000). Multilevel approach in statistical physics of liquids. In [Brandt *et al.*, 2000], pp. 187–197.

Brandt, A. and Livshits I. (1997). Wave-ray multigrid methods for standing wave equations, *Electr. Trans. Numer. Anal.*, 6:162–181.

Brandt, A. and Lubrecht, A. (1990). Multilevel matrix multiplication and the fast solution of integral equations, *J. Comput. Phys.*, 90(2):348–370.

Brandt, A., McCormick, S., and Ruge, J. (1982). Algebraic multigrid (AMG) for automatic multigrid solution with application to geodetic computations. Institute for Computational Studies, POB 1852, Fort Collins, Colorado.

Brandt, A., McCormick, S., and Ruge, J. (1983). Multi-grid methods for differential eigenproblems. *SIAM J. Sci. Statist. Comput.*, 4:244–260.

Brandt, A., McCormick, S., and Ruge, J. (1984). Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and its Applications* (Evans, D.J., ed.), Cambridge University Press, Cambridge, pp. 257–284.

Brandt, A. and Mikulinsky, V. (1995). Recombining iterants in multigrid algorithms and problems with small islands. *SIAM J. Sci. Comput.*, 16:20–28.

Brandt A. and Ron, D. (2001). Renormalization Multigrid (RMG): Statistically optimal renormalization group flow and coarse-to-fine Monte Carlo acceleration. *J. Stat. Phys.*, 102:231–257. Also appeared with modification in [Brandt *et al.*, 2000], pp. 163–186.

Brandt, A., Ron, D., and Amit, D.J. (1986). Multi-level approaches to discrete-state and stochastic problems. In: *Multigrid Methods, II* (Hackbusch, W. and Trottenberg, U., eds.), Springer-Verlag, pp. 66–99.

Brandt, A. and Venner, C.H. (1998). Fast evaluation of integral transforms with asymptotically smooth kernels. *SIAM J. of Sci. Comput.*, 19:468–492.

Brandt, A. and Venner, C.H. (1999). Multilevel evaluation of integral transforms on adaptive grids. In: *Multigrid Methods V*, Lecture Notes in Computational Science and Engineering **3** (Hackbusch, W. and Wittum, G., eds.), Sprinter Verlag, Berlin, pp. 20–44.

Brezina, M., Cleary, A.J., Falgout, R.D., Henson, V.E., Jones, J.E., Manteuffel, T.A., McCormick, S.F., and Ruge, J.W. Algebraic Multigrid Based on Element Interpolation (AMGe). LLNL Technical Report UCRL-JC-131752, submitted to *SIAM J. Sci. Comput.*

Briggs, W.L., Henson, V.E., and McCormick, S.F. (2000). *A Multigrid Tutorial*, 2nd Ed., SIAM.

Cox, T. and Cox, M. (1994). Multidimensional Scaling. Chapman and Hall, London.

Díaz, J., Petit, J., and Serna, M. (2002). A survey on graph layout problems. ACM Computing Surveys. To appear.

Dudkin, L.M. and Yershov, E.B. (1965). Interindustries input-output models and the material balances of separate products. *Planned Economy*, 5:54–63.

Ewald, P.P. (1921). Die Berechnung Optischer und elektrostatischer Gitterpotentiale. *Ann. Phys.*, 64 p. 253.

Fedorenko, R.P. (1964). The speed of convergence of an iterative process. *USSR comp. Math. Math. Phys.*, 4(3):227–235.

Fisher, M.E. (1998). Renormalization group theory: Its basis and formulation in statistical physics. *Rev. Mod. Phys.*, 70(2):653–681.

Galun, M. (1992). Optimal Multigrid Algorithms for Model Problems in Statistical Mechanics. M.Sc. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Galun, M. (1998). Multigrid Algorithms for Optimal Computations in Statistical Physics. Ph.D. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Gandlin, R. (2002). Final Report for Ph.D. Thesis. Weizmann Institute of Science, Rehovot, Israel.

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

Greengard, L. (1994). Fast algorithms for classical physics. *Science*, 265:909–914.

Hackbusch, W. (1985). Multigrid Methods and Applications. Springer, Berlin.

Hall, K.M. (1970). An r-dimensional Quadratic Placement Algorithm. *Management Science*, 17:219–229.

Kaminsky, R. (1989). Multilevel Solution of the Long Transportation Problem. M.Sc. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Karypis, G. (2002). Multilevel hypergraph partitioning. This book.

Keller, H.B. (1962). Geometrical theory of diffraction, *J. Optical Soc. Am.*, 52.

Keller, H.B. (1977). Numerical solution of bifurcation and nonlinear eigenvalue problems. In *Applications of Bifurcation Theory*, (Rabinowitz, P., ed.), Academic Press, New York, pp. 359–384.

Kirkpatrick, S., Gelatt, C.D., Jr., and Vecci, M.P. (1983). Optimization by simulated annealing, *Science*, 220 p. 671.

Koren, Y., Carmel, L., and Harel, D. (2002). ACE: A Fast Multiscale Eigenvectos Computation for Drawing Huge Graphs. In *IEEE Symposium in Information Visualization*, Boston. To appear.

Koren, Y. and Harel, D. (2002). A Multi-scale Algorithm for the Linear Arrangement Problem. Tech. Rep. MCS02-04, Faculty Maths. Comp. Sci., Weizmann Institute of Science, Rehovot, Israel.

Livne, O.E. (2000). Multiscale Eigenbasis Algorithms. Ph.D. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Livne, O.E. and Brandt, A. (2000). $O(N \log N)$ multilevel calculation of N eigenfunctions. In [Brandt *et al.*, 2000], pp. 112–136.

Mack, G. and pordt, A. (1988). Convergent perturbation expansions for Euclidean quantum field theory. G. t'Hooft *et al.*, eds. Plenum Press, New York, p. 309.

Mandel, J., Brezina, M., and Vanek, P. (1998). Energy optimization of algebraic multigrid bases, UCD/CCM Report 125.

Metropolis, N, Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., and Teller, E. (1953). Equation of state calculation by fast computing mechines, *J. Chem. Phys.*, 21 N6 p. 1087.

Mittelmann, H.D. (1982). Multigrid methods for simple bifurcation problems. In *Multigrid Methods*, Lecture Notes in Mathematics 960 (Hackbusch, W. and Trottenberg, U., eds.), Springer, Berlin, pp. 558–575.

Nilo, B. (1986). The Transportation Problem: A Multilevel Approach. M.Sc. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Petit, J. (1998). Approximation heuristics and benchmarkings for the MinLA problem. In *Alex '98 — Building bridges between theory and applications*, (Battiti, R. and Bertossi, A., eds.) Universite di Trento, pp. 112–128.

Ron, D. (1989). Development of Fast Numerical Solvers for Problems in Optimization and Statistical Mechanics. Ph.D. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Ron, D., Swendsen, R.H., and Brandt, A. (2002a). Inverse Monte Carlo Renormalization Group Transformations for Critical Phenomena. Submitted.

Ron, D., Swendsen, R.H., and Brandt, A. (2002b). Computer simulations at the fixed point using an inverse renormalization group transformation. Submitted.

Roweis, S.T. and Saul, L.K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science* 290:2323–2326.

Ruge, J. and Stüben, K. (1987). Algebraic multigrid. In *Multigrid Methods* (McCormick, S. F., ed.), SIAM, Philadelphia, pp. 73–130.

Safro, I. (2002). Multiscale methods for the Minimum Linear Arrangment of graphs. M.Sc. Thesis, Weizmann Institute of Science, Rehovot, Israel.

Schröder, J., Trottenberg, U, and Reuterssberg, H. (1976). Reuktionsverfahren fuer differenzengleichungen bei randwertaufgaben II. *Num. Math.*, 26:429–459.

Schwichtenberg, H. (1985). Erweiterungsmöglichkeiten des Programmpaketes M-G01 auf nichtlineare Aufgaben. M.Sc. Thesis, University of Bonn, West Germany.

Sharon, E., Brandt, A., and Basri, R. (2000a). Completion Energies and Scale. *IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR–97), Puerto Rico, 1997. Report CS97–19, Weizmann Institute of Science, Rehovot Israel,. *IEEE Trans. on Pattern Anal. and Machine Intelligence* 22:1117–1131.

Sharon, E., Brandt, A., and Basri, R. (2000b). Fast multiscale image segmentation. *IEEE Conf. on Computer Vision and pattern recognition*, South Carolina, pp. 70–77.

Sharon, E., Brandt, A., and Basri, R. (2001). Segmentation and boundary detection using multiscale intensity measurements. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Hawaii.

Shmulyian, S. (1999). Toward Optimal multigrid Monte Carlo Computations in Two-Dimensional $O(N)$ Non-Linear σ-models, Ph.D. Thesis, Weizmann Institute of Science, Rehovot, Israel.

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

Stüben, K. (2000). Algebraic multigrid (AMG): An introduction with applications. Guest appendix in [Trottenberg *et al.*, 2000]. A review of algebraic multigrid, *J. Comp. Appl. Math.*, 128:(1–2), 2001.

Swendsen, R.H. and Wang, J.S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Phys. Rev. Lett.*, 58:86–88.

Tanabe, K. (1971). Projection methods for solving a singular system if linear equations and its applications. *Num. Math.*, 17:203–214.

Trottenberg, U, Oosterlee, C.W., and Schüller, A. (2000). *Multigrid.* Academic Press, London.

Vanek, P., Mandel, J., and Brezina, M. (1994). Algebraic multigrid on unstructured meshes. University of Colorado at Denver, UCD/CCM Report No 34.

Vanek, P., Mandel, J., and Brezina, M. (1996). Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56:179–196.

Vanek, P, Brezina, M, and Mandel, J. (1998). Convergence of algebraic multigrid based on smoothed aggregation. UCD/CCM Report 126. Submitted to *Numer. Math.*

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

Venner, C.H. and Lubrecht, A.A. (2000). *Multilevel Methods in Lubrication.* Elsevier (Tribology Series, 37), Amsterdam.

Walshaw, C. (2002). An exploration of multilevel combinatorial optimization. This book.

Wan, W.L., Chan, T.F., and Smith, B. (1998). An energy minimization interpolation for robust multigrid methods. Department of Mathematics, UCLA, UCLA CAM Report 98-6.

Washio, T. and Oosterlee, C.W. (1997). Krylov subspace acceleration for nonlinear multigrid schemes. *Electr. Trans. Num. Anal.*, 6:271–290.

Weinands, R. (2001). Extended Local Fourier Analysis for Multigrid: Optimal Smoothing, Coarse Grid Correction and Preconditioning. Ph.D. Thesis, Universität zu Köln.

Wilson, K.G. (1983). The Renormalization Group and Critical Phenomena. *Rev. Mod. Phys.*, 55:583–600. (1982, Nobel Prize Lecture)

Wolff, U. (1989). Collective Monte Carlo updating for spin systems. *Phys. Rev. Lett.*, 62:361–364.

Yavneh, I. and Dardyk, G. A multilevel nonlinear method. In preparation.

Zimare, H. (1980). *Beweis der sternenkonvergenz und untersuchung der stabilitaet beim verfahren der totalen reduktion*. Ph.D. Thesis, University of Cologne.