Chapter 13

Dealgebraization of Multigrid

An interesting line in the development of multigrid can be viewed as a gradual "dealgebraization", a gradual liberation from algebraic concepts, and the development of methods that increasingly exploit the underlying *differential* nature of the problems. We'd like to briefly trace this line here, so as to bring out some concepts useful in practical implementations.

As the first step of dealgebraization we can regard the **replacement of "acceleration" by "smoothing"**. The early two-grid and multi-grid approach viewed coarse-grid corrections mainly as a tool for accelerating the basic iterative process - the fine-grid relaxation. Only later it became clear that the only role of relaxation is to smooth the error. (Cf. §12, where a further "dealgebraization" of the smoothing concept is described.) This slight shift in understanding revolutionized the multigrid practice: It made it clear that the fine-grid process is basically local, hence analyzable by local mode analysis. This understanding, together with that analysis, produced the truly efficient multigrid cycles, in which very few sweeps are made on each grid before switching to coarser ones, and in which the fine-to-coarse meshsize ratio assumes the (practically) optimal value of 1 : 2.

The next dealgebraization steps are related to the trivial understanding that we are not primarily interested in solving the algebraic equations (obtaining u^h), but we are interested in approximating the differential solution u. First, this implies that we have to solve the algebraic equations only "to the level of truncation errors", i.e., only to the point that our calculated solution $\tilde{u}^h$ satisfies $\|\tilde{u}^h - u^h\| \approx \|u - u^h\|$; further reduction of $\tilde{u}^h - u^h$ is futile.

This implies that the asymptotic convergence rate of the multigrid cycle is not important by itself. What counts is the amount of work we need in an FMG algorithm in order to reduce the error from its original value on grid $2h$, which is approximately $\|\mathbb{I}_{2h}^h u^{2h} - u\| \approx \|u^{2h} - u\|$, to the desired level $\|u^h - u\|$. This is a reduction by a modest factor, which can usually be achieved in one cycle. (The fundamental reason for this is again non-algebraic: See §7.3.) Evidently it is then more relevant to think in terms of the FMG analysis (§7.4) than in terms of asymptotic rates.

Even the later viewpoint, that we want to reduce the errors to the level of truncation errors, is too algebraic-oriented. It is tied too much to one given discretization on one given grid. The optimal moment of switching from a certain currently-finest grid H to a new, finer grid $h = H/2$ is not necessarily when $\|\tilde{u}^H - u^H\| \approx \|u - u^H\|$. Rather, it is determined by comparing H-cycles to h-cycles in their efficiency at driving $\tilde{u}^h$ closer to u (see §7.2). what really counts is the behavior of the differential error $E = \|\tilde{u}^h - u\|$ as a function of the total accumulated computational work W.

We want $E(W)$ to be as fast-decreasing as possible.

From this as our objective we can derive correct switching criteria, i.e. decide when to establish a new finer grid. The next step is to realize that criteria based on $E(W)$ can be applied *locally*, to decide not only *when* to have a finer grid, but also *where* to have it. This naturally brings us to grid-adaptation (§9.5). Indeed one can integrate the switching and self-adaption criteria (discussed in §6.2, 7.2, 9.5) into a total **multi-level adaptive algorithm**, where switching between levels and creating new, or extending existing, levels are all governed by the same exchange-rate criteria (see §9.6).

Another step away from fixed algebraic concepts is to allow **variable discretization schemes**, i.e., schemes which can be changed throughout the algorithm to promote faster decreasing $E(W)$. This includes the use of higher-order, variable-order and adaptable-order schemes, governed again by E(W) criteria (see §9.5 and [Bra79b, §3.6, §4.3]). Using different discretization schemes in relaxation and in residual transfers (§10.2) is a further step in that direction.

By now we have gone quite far beyond the notion of multigrid as just a fast algebraic solver, toward viewing it as a **total treatment of the original problem**. This is proved to be a very beneficial general principle: Always think of multigrid in terms of as original a problem as possible: For example, instead of using Newton iterations, employing multigrid as a fast solver of the linearized problems, apply multigrid directly to the non-linear problem (§8.3). Instead of solving an eigenproblem by the inverse power method, with multigrid as the fast inverter, you can multigrid directly the original eigenvalue problem (§8.3.1). Instead of using multigrid for solving each step in some outer iterative process - be it a continuation process, a time-dependent evolution, a process of optimizing some parameters or solving an inverse problem, etc. - apply it directly to the originally given problem (cf. §8.3.2, 15, 16). Instead of a grid adaptation process where the discrete problem on each grid configuration is completely solved (by multigrid, say) and then used to decide on an improved grid configuration, the whole adaptation process can be integrated into a multigrid solver (§9.6); and so on.

An illustration to this approach is the solution of **optimization problems**, where the parameter to be optimized is some continuum *function* on which the solution u depends. This "parametric function" may for example be the shape of the boundary (e.g., the shape of an airplane section which we want to optimize in some sense), or a certain coefficient of the differential equations (e.g., in inverse problems, where one tries to determine this coefficient throughout the domain so that the solution will best fit some observational data), etc. Multigridding the original problem means that we solve it by some FMG algorithm, where already at the coarser FMG stages we treat the given *optimization* problem, by optimizing a coarser representation of the parametric function. On the finer grids, incidentally to relaxing the equations, we optimize that function locally (when this makes sense), and then we introduce smooth corrections to the function during the coarse-grid correction cycles. Instead of using the multigrid solver many times, we may end up doing work only modestly larger than just *one* application of that solver.

13.1 Reverse trend: Algebraic multigrid

Contrary to the above line of dealgebraization, there is a recent trend to develop purely Algebraic Multi-Grid (AMG) algorithms. By this we mean a multi-level algorithm without any geometry, without grids. An algebraic (linear or nonlinear) system of equations is given. To solve it fast, a sequence of increasingly "coarser" levels is created. A coarser level in this context is a related, but much smaller, algebraic system. The choice of the coarse-level *variables* and of the coarse-to-fine interpolation I_H^h, is based not on geometric positions but on the algebraic equations themselves: The coarse variables are chosen so that each fine-level variable is strongly coupled to one or more of them, and each I_H^h coefficient can for example be chosen to be proportional to the corresponding coupling strength. The fine-to-coarse transfer and the coarse-level matrix are then constructed by prescriptions like (4.12) and (4.11), respectively. The theoretical background directing the various choices is developed in [Bra86].

Generally, the efficiency that can be achieved by such algebraic algorithms is below that of algorithms built to exploit the geometric information, let alone the further efficiency obtainable by further dealgebraization. On the other hand these algebraic solvers may be used as black boxes for larger classes of problems. They may especially be useful in cases where the geometrical information is too complicated, such as finite-element equations based on arbitrary partitions, or various problems which are not differential in their origin but still lend themselves for fast multi-level solutions. Also, there are cases of finite-difference equations on a uniform grid, in which the usual geometric choice of coarse-grid variables is not good, since too many finegrid variables happen to depend too weakly on the coarse-grid variables (cf. e.g., [ABDP81, §8]). Algebraic multigrid can then perform better. Because of its sensitive coarsening, there is in AMG no need for special relaxation schemes, in varying block and marching directions (cf. §3.3); simple Gauss-Seidel is for example used for all definite problems. Experiments on a wide range of problems, including discretization of regular and degenerate second-order elliptic equations as well as problems with no continuous origin, show that the typical multigrid convergence rates are robustly obtained [BMR84], [Stü83]. The AMG set-up time is expensive, but still comparable to the set-up time required by any Galerkin coarsening (4.11). Work is underway to generalize AMG to other classes of matrices, such as those arising in discretizing *systems* of differential equations.