2.2 Wavelet and Multigrid Methods for Convection–Diffusion Equations

Thomas Gerstner, Frank Kiefer, Angela Kunoth

May 2001

Abstract

Multiscale methods such as wavelet and multigrid schemes are efficient algorithms for the numerical solution of large classes of partial differential equations. In this chapter we illustrate the basic ideas behind iterative methods for the solution of the resulting large linear systems which arise from the discretization of PDEs. Besides describing the concepts of wavelet preconditioning, we summarize the main ingredients of multigrid methods. Examples from computational geosciences will show the importance of multiscale methods for the simulation of large and complex systems.

The simulation of physical processes typically requires the numerical solution of various types of partial differential equations that model the processes, see e.g. [HHN]. Discretizing the equations on realistic domains in three dimensions, one finally faces the problem to solve very large but sparsely populated systems of linear equations. While the amount of unknowns no longer allows for direct solution techniques to decompose the system matrices, classical iterative methods converge only very slowly. This is due to the fact that standard (single–scale) discretizations lead to system matrices with condition numbers depending on the discretization in such a way that they grow very fast as the discretization step size becomes finer. These large condition numbers in turn are the reason for the slow convergence of iterative methods.

In contrast, wavelet and multigrid methods provide the solution of very large and sparse systems up to prescribed accuracy quickly in a *fixed* number of iterations. The key to their success is the fact that the differential equations are discretized with respect to *different* discretization step sizes or *grids* providing a hierarchical decomposition of the domain in order to capture the multiscale behavior of the solution. The solutions on the different scales are then appropriately catenated to build the solution of the initial system. This can be viewed as *preconditioning* the linear system of equations: by applying these multilevel methods one can achieve what is asymptotically the best possible, namely, that the convergence speed of the methods does not depend any more on the number of unknowns.

The necessary details for describing wavelet techniques are collected in Chapter 1.2. We briefly explain here how wavelets are used for preconditioning.

The basic mathematical ideas for multigrid methods are described in terms of *intergrid operators* called restriction and prolongation operators that map matrices and vectors living on one grid into structures on a grid of another length scale. Below we describe the construction and usage of both classical and wavelet–based multigrid methods for process simulation. However, before we do so, we derive a standard discretization for a (partial) differential equation and discuss classical iterative methods in order to be able to precisely describe the effects of applying the multilevel techniques.

2.2.1 Discretization of Partial Differential Equations

There is indeed a variety of multilevel methods that can be viewed rather as a methodology than a single method. However, they all rely on a few basic principles that can be sufficiently described in terms of the simplest example of process modeling, a stationary (not time–dependent) one–dimensional partial differential equation. Namely, consider the boundary value problem

$$\begin{aligned} -u''(x) &= f(x), \qquad 0 < x < 1, \\ u(0) = u(1) &= 0. \end{aligned} \tag{2.2.1}$$

In order to solve this problem by a numerical method, one introduces a grid on the underlying domain $\Omega := (0,1)$ (which is an open interval here) and then computes the solution u of (2.2.1) approximately with respect to the grid points. Namely, decompose Ω into $N+1$ subintervals by defining (equidistant) grid points $x_j := jh$ where $h = (N+1)^{-1}$ is the grid size or resolution. We use the notation $\Omega_h := \{x_j\}_{j=0,\dots,N+1}$ to refer to the grid.

In order to derive a discretization of the differential equation (2.2.1) relative to the grid Ω_h, there are different methods available. Modern methods are often based on deriving a weak formulation of the differential equation which then allows for using for instance finite elements or wavelets. Since we want to describe the main ideas, here we apply a classical difference method. That is, one approximates the second partial derivative by a (second order) difference quotient, evaluated at the (interior) grid points,

$$-u''(x_j) \approx \frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2}, \qquad j = 1, \dots, N. \tag{2.2.2}$$

By setting $u_j := u(x_j)$ and observing that the boundary values in (2.2.1) give $u_0 = u_{N+1} = 0$, one obtains a linear system of N equations for the N unknowns $v := (u_1, \ldots, u_N)^T$ that approximate u at the grid points x_j,

$$Av = b. \tag{2.2.3}$$

Here the vector $b \in I\!R^N$ is just the right hand side $f(x)$ of (2.2.1) evaluated on Ω_h, and A is a $N \times N$–matrix with entries determined by the differences in (2.2.2), that is,

$$A = \frac{1}{h^2} \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \end{pmatrix}. \tag{2.2.4}$$

Since A is nonsingular, the system (2.2.3) has for any given b a unique solution v. Note that A is a tridiagonal matrix which has nonzero entries only on the diagonal and its side diagonals.

In order to describe different methods for solving the system (2.2.3), it is more instructive at this point to consider the two–dimensional version of the differential equations (2.2.1) and its discretizations. To this end, consider the Poisson problem on the unit square, involving a second order partial differential equation,

$$\begin{aligned} -\Delta u(x,y) := -\left(\tfrac{\partial^2}{\partial x^2} + \tfrac{\partial^2}{\partial y^2}\right) u(x,y) &= f(x,y), \qquad (x,y) \in \Omega := (0,1)^2, \\ u|_{\partial\Omega} &= 0, \end{aligned} \tag{2.2.5}$$

where $\partial\Omega$ denotes the boundary of Ω. For discretization, associate with Ω again a grid Ω_h that consists of subdividing the x and the y axis into $N+1$ subintervals, yielding $\Omega_h := \{(x_i, y_k),\ 0 \le i, k \le N+1\}$. The discretization of (2.2.5) is then performed analogously to the one–dimensional case, approximating now the second order partial derivatives by second order differences in the x and y direction each. To set up the corresponding linear system of equations, the grid points (x_i, y_k) are ordered (for instance) lexicographically, yielding the problem to determine $n := N \times N$ unknowns $v_j \approx u(x_i, y_k)$ that solve

$$Mv = b \tag{2.2.6}$$

where

$$M = \begin{pmatrix} A & -\tilde{I} & & & \\ -\tilde{I} & A & -\tilde{I} & & \\ & -\tilde{I} & A & -\tilde{I} & \\ & & \ddots & \ddots & \ddots \\ & & & -\tilde{I} & A \end{pmatrix}. \tag{2.2.7}$$

Here A is the matrix from the univariate problem (2.2.4), $\tilde{I} := h^{-2}I$ where I is the identity matrix of size N and b contains the values of the right hand side f evaluated on Ω_h. Like A above, the matrix M is nonsingular, admitting for any given right hand side a unique solution v^*. Note that M has an amount of entries which is proportional to the number of unknowns $n = N^2$. Such matrices are called *sparse*. However, in contrast to A, the matrix M has in addition to the tridiagonal pattern two side diagonals which are 'far away' from the main diagonal, its distance being determined by the number of unknowns in the respective spatial direction. These always occur in some form or another (using any ordering of the unknowns) when the spatial dimension is larger than one. Of course, if one increases the grid refinement and enlarges N, the size of the matrix grows quadratically. Sparse matrices are usually stored in a sparse format, that is, one stores only the nonzero entries and the necessary information on their position, therefore requiring $\mathcal{O}(n)$ amount of storage. If now the linear system $Mv = b$ is solved by any direct method like Gaussian elimination or QR decomposition, this creates *fill–in*, that is, all the diagonals consisting of zeros between the main diagonals and the diagonal 'far away' fill up with non–zero entries, requiring a much higher amount of storage. In three spatial dimensions, this effect is even more dramatic.

2.2.2 Iterative Methods

For that reason, one usually solves sparse systems of the form (2.2.6) by iterative methods. These require in principle only asymptotically the same amount of storage. Also by the discretization of a boundary value problem (2.2.5) one already introduces a *discretization error*. Thus, also the linear system needs to be solved only up to an accuracy determined by the discretization error. Using a direct method instead which would solve $Mv = b$ exactly (up to machine precision) would be a waste of time.

Next we describe some basic iterative methods for the solution of linear systems $Av = b$, see e.g. [Ha2]. For simplicity, for the remainder of this chapter, we consider the system stemming from the one–dimensional equation (2.2.1). The basic iterative methods are derived from writing the (scaled) equations (2.2.3) as a fixed point equation $v = v + C(b - Av)$ from which the iteration

$$v^{(i+1)} = (I - CA)v^{(i)} + Cb, \qquad i = 0, 1, 2, \dots \tag{2.2.8}$$

follows. Here $C \in \mathbb{R}^{n \times n}$ is some nonsingular matrix determined later and $I \in \mathbb{R}^{n \times n}$ is the identity. That is, starting from some initial guess $v^{(0)}$, one obtains by the assignment (2.2.8) the next iterate $v^{(1)}$, uses this to determine $v^{(2)}$, and so on, until a prescribed tolerance is met. Any iteration is called convergent if the iterates $\{v^{(i)}\}$ converge for an arbitrary initial guess $v^{(0)}$ to the (unique) solution v^* of (2.2.6). The iteration (2.2.8) converges if and only if

$$\rho(I - CA) < 1,$$

that is, the modulus of all eigenvalues of $I - CA$ is strictly smaller than 1. For convergent iterations, the value of $\rho(I - CA)$, the *convergence factor*, determines the *speed* of convergence. Roughly speaking, it is the factor by which the error $e^{(i)} := v^{(i)} - v^*$ is reduced in each iteration sweep. From this, one can also determine the amount of iterations that are required to reduce an initial error $e^{(0)} = v^{(0)} - v^*$ by a factor of 10^{-c} as follows. For some vector norm $\|\cdot\|$ let m be the smallest integer such that the error in the i–th iterate satisfies

$$\|e^{(i)}\| \leq 10^{-c}\,\|e^{(0)}\|,$$

which is approximately the case if

$$(\rho(I - CA))^m \leq 10^{-c}.$$

This is satisfied if

$$m \geq -\frac{c}{\log_{10}(\rho(I - CA))}.$$

The quantity $\log_{10}(\rho(I - CA))$ is called the *convergence rate*. It is the amount of iterations which are required to reduce the error by one decimal digit. Note that the closer $\rho(I - CA)$ is to 1, the smaller the convergence rate and thus the slower the convergence is. Of particular importance are such methods where $\rho(I - CA)$ is *independent* of the discretization, i.e., independent of the grid size h. In such cases one can achieve a prescribed accuracy in a *fixed* amount of iterations independent of the discretization.

Up to this point, we have not yet specified the role of the matrix C in (2.2.8). If $C = A^{-1}$, the inverse of A, then $I - CA = 0$ and the iteration (2.2.8) terminates after one step. But determining A^{-1} explicitly is just the problem that is to be avoided. Nevertheless, it gives an idea how one should choose C, namely, to approximate A^{-1} as good as possible at ideally low cost. Different possibilities stem from splitting A additively into

$$A = D + L + U, \tag{2.2.9}$$

where D is the diagonal, L the strict lower triangular and U the strict upper triangular part of A. Choosing now $C = D^{-1}$ is the approximation to A^{-1} which is easiest to compute, but also has mostly only very limited effects. This method (possibly with an appropriate weighting term) is called *Jacobi method*. Observing that triangular matrices can also be easily inverted (by forward or backward substitution), a more sophisticated choice which is still quickly computable is to take $C = (D + U)^{-1}$. This variant is commonly called *Gauss-Seidel method*. Both Jacobi and Gauss-Seidel methods are nowadays seldom used by themselves since their convergence (provided that they converge) is usually too slow. However, they are used as one of the basic ingredients in multigrid methods where these basic iterative methods are usually referred to as *smoothing* iterations. This is due to the fact that one can observe e.g. by Fourier mode analysis that small oscillations in the errors are quickly 'smoothed' out by such iterations.

Up to this point, the description of the iterative methods and the remarks on convergence apply to any linear system of equations. For the particular case (2.2.6), the matrix stemming from the discretization of the partial differential equation (2.2.1) or (2.2.5) has particular properties. Namely, A (and also M defined in (2.2.7)) is symmetric and positive definite, thus, A has only positive real eigenvalues. For such matrices, both the Jacobi and the Gauss-Seidel iterations converge. Their convergence speed depends on the (spectral) *condition number* of A which is given in terms of the largest and smallest eigenvalues of A,

$$\kappa_2(A) := \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

Essentially the spectral radius for each of these methods satisfies

$$\rho(I - CA) \;\approx\; \frac{\kappa_2(A) - 1}{\kappa_2(A) + 1} = 1 - \frac{2}{\kappa_2(A) + 1}$$

which is strictly smaller than one. However, for matrices A stemming from the discretization of the partial differential equation (2.2.5), the condition number is proportional to h^{-2},

$$\kappa_2(A) = \mathcal{O}(h^{-2}) \tag{2.2.10}$$

resulting in a spectral radius which is very close to 1. Thus, if one wants to decrease the grid size h in order to obtain a better approximation to the solution of the partial differential equation, this results in growing condition numbers and exceedingly slow convergence of such methods. Here one speaks of an *ill–conditioned* matrix A.

2.2.3 Preconditioning

The other extreme is when the matrix C in the iteration (2.2.8) can be found in such a way that the spectral radius of $I - CA$ does *not* depend on the grid spacing h at all. There are essentially two strategies which can achieve that. They can by summarized as *preconditioning techniques* since they massively improve the condition number of the system matrix. The first strategy is denoted by *additive preconditioning technique*. There are basically two techniques that make up this class, namely, the *Fast Wavelet Transform* and the *BPX Preconditioner* [DK]. The first technique consists of applying an appropriately scaled version of the Fast Wavelet Transform described in Chapter 1.2, yielding a transformed matrix $\tilde{A}$ which satisfies $\kappa_2(\tilde{A}) = \mathcal{O}(1)$ *independent* of $h \sim 2^{-J}$.

Here we present some numerical results from [KPV]. In Table 1 we have discretized the one–dimensional problem $-u''+u = 1$ with periodic boundary conditions $u(0) = u(1)$ by means of periodized biorthogonal spline–wavelets which are based on the piecewise linear hat function (indicated by $d = 2$) and coarsest level 0. The second column labelled `no prec.` shows the effect (2.2.10) on the condition number with growing finest level J. ('x' indicates that the condition number has not been computed.) Using the wavelet preconditioner indicated by D^{-1}, one sees in the third column that the condition number stays uniformly bounded independent of J. A further reduction in the absolute values of the condition numbers is achieved by taking into account a scaling by the inverse of the diagonal of A in terms of the wavelet representation instead of the diagonal scaling, listed in the last table and denoted by $A_{\Psi^J}^{-1}$.

Level J	$d=2,\ j_0=0$		
	no prec.	D^{-1}	$A_{\Psi^J}^{-1}$
3	2.73e+02	5.17e+00	1.62e+00
4	1.11e+03	5.59e+00	1.95e+00
5	4.43e+03	6.13e+00	2.27e+00
6	1.77e+04	6.50e+00	2.51e+00
7	7.08e+04	6.83e+00	2.72e+00
8	2.83e+05	7.09e+00	2.88e+00
9	x	7.32e+00	3.01e+00

Table 1: Condition numbers for $-u'' + u = 1$ using periodized biorthogonal spline–wavelets.

For many applications, periodic boundary conditions are not realistic. Therefore we have also tested the effect of using wavelets which are adapted to bounded domains, which is an interval here, described in Chapter 1.2, see also [DKU1]. The effect of the boundary adaption on the absolute values of the corresponding condition numbers can be seen in Table 2. Nevertheless, asymptotically they still remain uniformly bounded. It is known that these numbers can be further reduced by more sophisticated strategies to construct wavelets adapted to intervals, see [Bar, DKU2]. Another influence comes here from the fact that the finest level is 3 (for technical reasons in the construction due to the fact that then the functions living at each end of the boundary do not overlap).

The other strategy for preconditioning is the class of *multigrid methods* that can also be viewed as *multiplicative preconditioning techniques*. Both strategies are *optimal* in the sense that they provide the solution in a fixed amount of steps *independent* of the grid spacing h.

2.2.4 Multigrid Methods

In the following, we will describe the basic ingredients of multigrid methods for the efficient solution of sparse linear systems which arise from the discretization of elliptic partial differential equations (see [BHM] for an introduction and [Br, Ha1]). Multigrid methods consist of two complementary components,

Level J	$d=2,\ j_0=3$		
	no prec.	D^{-1}	$A_{\Psi^J}^{-1}$
4	1.81e+02	4.59e+01	2.34e+01
5	7.40e+02	5.51e+01	2.48e+01
6	2.96e+03	6.12e+01	2.58e+01
7	1.19e+04	6.66e+01	2.61e+01
8	4.75e+04	7.07e+01	2.64e+01
9	1.90e+05	7.41e+01	2.67e+01

Table 2: Condition numbers for $-u'' + u = 1$, $u(0) = u(1) = 0$, using boundary adapted wavelets.

- the *smoothing iteration*,
- and the *coarse grid correction*.

If one applies classical iterative methods of the form (2.2.8) (e.g. a Gauss–Seidel or Jacobi method) for the solution of such linear systems, these traditional solvers strongly damp out high frequency (*rough*) parts of the error between the discrete solution and the iterates with only a few iterations. However, low frequency (*smooth*) error components will be reduced only slowly and the error reduction is minimal as soon as the error essentially consists of only smooth components. In this sense traditional iterative methods are called *smoothers*. Now, low frequency error components may be represented on a coarser scale without loss of accuracy but with considerably less degrees of freedom. Thus, the central idea is to transport the smoothed error onto a coarser grid to reduce it there with the help of an appropriate coarse grid problem, implying lower computational costs. Since ideally the problem on the coarse grid has the same structure as the original problem on the fine grid, this procedure may be continued recursively, resulting in what is called *multigrid methods*.

In order to describe the coarse grid correction in somewhat more detail, it will be sufficient to consider only *two* different grids.

The grid Ω_h introduced in Section 2.2.1 will play the role of the *fine grid*. In addition, we assume that there is a second grid Ω_H with respect to a coarser discretization $H > h$ and such that the grid points of Ω_H are also grid points of Ω_h. Here we consider for simplicity

$$H = 2h.$$

We denote by $V_h = I\!R^{N_h}$ and $V_H = I\!R^{N_H}$ the spaces of corresponding grid functions, respectively. For transporting vectors and matrices from one grid to the other, one needs two linear *intergrid operators.* The one that takes vectors from the fine to the coarse grid is called *restriction operator*,

$$R_{h\to H} : V_h \to V_H; \tag{2.2.11}$$

its counterpart *prolongation operator*,

$$P_{H\to h} : V_H \to V_h. \tag{2.2.12}$$

For instance, one chooses $R_{h\to H}$ and $P_{H\to h}$ of the following form

$$R_{h\to H} = \frac{1}{2}\begin{pmatrix} \frac{1}{2} & 1 & \frac{1}{2} & & & & & \\ & & \frac{1}{2} & 1 & \frac{1}{2} & & & \\ & & & & \frac{1}{2} & 1 & \frac{1}{2} & \\ & & & & & \ddots & \ddots & \ddots \end{pmatrix} \in I\!R^{N_h \times N_H}, \tag{2.2.13}$$

and

$$P_{H\to h} = \begin{pmatrix} \frac{1}{2} & & & \\ 1 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ & 1 & & \\ & \frac{1}{2} & \frac{1}{2} & \\ & & 1 & \\ & & \frac{1}{2} & \frac{1}{2} \\ & & \ddots & \ddots & \ddots \end{pmatrix} \in I\!R^{N_H \times N_h}. \tag{2.2.14}$$

The prolongation and restriction operators are usually chosen such that they satisfy the *consistency conditions* $P_{H\to h} = c\, R^T_{h\to H}$ with some constant c which is 2 here.

Now the coarse grid correction can be described as follows. Let $v_h \in V_h$ be some approximation of the solution of the linear system $A_h v = b_h$ discretized with respect to the fine grid Ω_h. Denote by $r_h := b_h - A_h v_h$ the *residual*. The residual is transported onto the coarse grid Ω_H by applying the restriction operator $R_{h\to H}$ to r_h, yielding the vector $r := R_{h\to H} r_h$. Then the equation $A_H e = r$ is solved up to some accuracy by a basic iterative method, giving an approximation e_H to e. Finally one transports e_H back to the fine grid by applying the prolongation operator $P_{H\to h}$ and adds this to the previous approximation v_h, giving a new approximation

$$\tilde{v}_h := v_h + P_{H\to h} e_H. \tag{2.2.15}$$

The terminology 'coarse grid correction' stems from the fact that $\tilde{v}_h$ can formerly also be expressed as

$$\tilde{v}_h := v_h + P_{H\to h} A_H^{-1} R_{h\to H}(b_h - A_h v_h) \tag{2.2.16}$$

where, of course, instead of the explicit computation of A_H^{-1} on the coarse grid the linear system is solved iteratively.

From the viewpoint of the wavelet terminology of Chapter 1.2 and subdivision schemes in Computer Aided Geometric Design, the intergrid operators $R_{h\to H}$ and $P_{H\to h}$ can be interpreted also as *mask matrices* [DkK].

If one restricts the smoothing procedure to those degrees of freedom which are represented by grid points that do not belong to the coarser grid, one obtains the *Hierarchical basis Two–Grid Method* [Ba, BDY]. From the point of view of subspace correction methods [X], the corresponding smoothing iteration can be considered as a correction step which is performed only with respect to a hierarchical complement space. This complement space is spanned by fine grid functions that are located in grid points which do not belong to the coarser grid anymore. Note that the Hierarchical Basis Two-Grid Method may also be interpreted as a block Gauss–Seidel method for the two scale hierarchically transformed linear system. In this sense it can be seen as an approximate block LU–factorization of the hierarchically transformed discrete operator.

Working with more than two grids and applying the above procedures recursively, one can immediately generate (*Hierarchical Basis*) *Multigrid Methods*. There are many different variants when working with several grids. Depending on how one proceeds inbetween grids, starting from the finest and descending through all grids to the coarsest ('V-cycle') or moving up and down between intermediate grids ('W-cycle') are some of the possibilities.

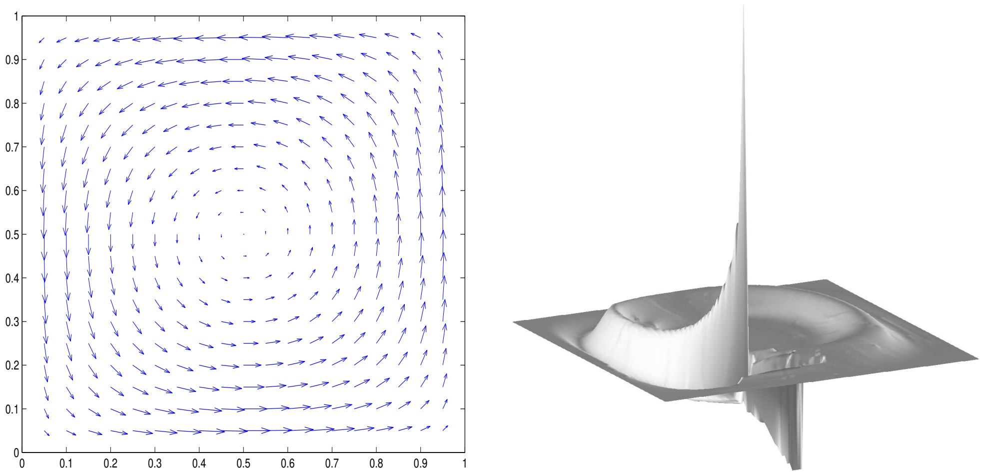

Figure 1: Left picture: circular convective vector field; right picture: AMGlet generated by a wavelet–like stabilization of the corresponding AMG–based hierarchical basis decomposition of the trial space.

If one chooses wavelet–like basis functions to span the complement spaces within the hierarchical basis multigrid method, one obtains *Multiscale Methods* which can be viewed as wavelet–like multigrid or generalized hierarchical basis methods [GK, Ki, P, VW2]. They can be interpreted as ordinary multigrid methods which use a special kind of *multiscale smoother* [Ki]. Furthermore, one can show an optimal convergence behavior similar to classical multigrid [VW1] for 'nicely' elliptic problems. From a theoretical point of view this is due to the enhanced stability properties of the corresponding subspace decompositions [CDP, O]. From the multigrid interpretation the reason for this is the stronger effect of the respective smoothing iteration compared to the hierarchical one. However, since wavelet–based multiscale smoothers still compute corrections only for parts of the unknowns, it is rather difficult to construct *robust* multiscale solvers which do not depend on the mesh size used for discretization *and* the coefficients of the partial differential operator involved. One means to construct robust solvers based on direct splittings of the underlying function spaces is a smart choice of the multiscale decompositions itself. In [Ki] a rather general Petrov–Galerkin multiscale concept is developed. Here, problem–dependent coarsening strategies known from robust multigrid techniques (matrix–dependent prolongations, algebraic multigrid (AMG)), [ABDP, BHM, De1, De2, Gr, HKWZ, Oe, RS, St, Z1, Z2] are applied together with specific wavelet–like and hierarchical multiscale decompositions of the trial and test spaces on the finest grid. The main idea is to choose one complement space hierarchically (to implicitly generate physically meaningful coarse-grid operators) and the other one wavelet–like (to stabilize the multiscale setting with respect to the mesh size, and to strengthen the smoother). The numerical results in [Ki] show that by this choice wavelet–type multiscale solvers can be constructed which result in robust and efficient solvers. There also tables of multigrid convergence rates can be found for the following example. On the left of Figure 1 a circular convective vector field is shown. It belongs to the singularly perturbed convection–diffusion operator

$$A = -\Delta - 4 \cdot 10^6 \, (x(x-1)(1-2y) \, \partial_x - y(y-1)(1-2x) \, \partial_y)$$

subject to homogeneous Dirichlet boundary conditions on the unit square. Since the wavelet–type basis functions are implicitly generated by the AMG–approach, we call them *AMGlets*. On the right of Figure 1, one can see that apparently the AMGlet follows the convective flow and takes positive *and* negative values as a wavelet should.

References

[ABDP] R. E. Alcouffe, A. Brandt, J. E. Dendy, J. W. Painter. *The multigrid method for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Comput. 2, 1981, 430–454.

[Ba] R. E. Bank, *Hierarchical bases and the finite element method*, Acta Numerica 5, 1996, 1–43.

[BDY] R. E. Bank, T. F. Dupont, H. Yserentant, *The hierarchical basis multigrid method*, Numer. Math. 52, 1988, 427–458.

[Bar] T. Barsch, *Adaptive Multiskalenverfahren für elliptische partielle Differentialgleichungen — Realisierung, Umsetzung und numerische Ergebnisse* (in German), IGPM, RWTH Aachen, Dissertation, 2001.

[BHM] W. L. Briggs, V. E. Henson, S. F. McCormick, *A Multigrid Tutorial*, SIAM, Phiadelphia, 2000.

[Br] D. Braess, *Finite Elements: Theory, Fast Solvers and Applications in Solid Mechanics*, 2nd. ed., Cambridge University Press, Cambridge, 2001.

[CDP] J.M. Carnicer, W. Dahmen, J.M. Peña, *Local decomposition of refinable spaces*, Appl. Comp. Harm. Anal. 3, 1996, 127–153.

[DkK] S. Dahlke, A. Kunoth, *Biorthogonal wavelets and multigrid*, in: Adaptive Methods — Algorithms, Theory and Applications, Proceedings of the 9th GAMM Seminar, W. Hackbusch, G. Wittum (eds.), NNFM Series Vol. 46, Vieweg, Braunschweig, 1994, 99–119.

[DK] W. Dahmen, A. Kunoth, *Multilevel preconditioning*, Numer. Math. 63, 1992, 315–344.

[DKU1] W. Dahmen, A. Kunoth, K. Urban, *Biorthogonal spline-wavelets on the interval — Stability and moment conditions*, Appl. Comp. Harm. Anal. 6, 1999, 132–196.

[DKU2] W. Dahmen, A. Kunoth, K. Urban, *Wavelets in numerical analysis and their quantitative properties*, in: Surface Fitting and Multiresolution Methods, A. Le Méhauté, C. Rabut, L.L. Schumaker (eds.), Vanderbilt University Press, Nashville, TN, 1997, 93–130.

[De1] J. E. Dendy, *Black box multigrid*, J. Comput. Phys. 48, 1982, 366–386.

[De2] J. E. Dendy, *Black box multigrid for nonsymmetric problems*, Appl. Math. Comput. 13, 1983, 261–283.

[Gr] M. Griebel, *Zur Lösung von Finite-Differenzen- und Finite–Element-Gleichungen mittels der Hierarchischen Transformations-Mehrgitter–Methode*, Technical Report 342/4/90 A, TU München, 1990.

[GDN] M. Griebel, T. Dornseifer, T. Neunhoeffer, *Numerical Simulation in Fluid Dynamics, a Practical Introduction*, SIAM, Philadalphia, 1998.

[GK] M. Griebel, F. Kiefer, *Generalized hierarchical basis multigrid methods for convection–diffusion problems*, Manuskript, IAM, Universität Bonn, 2001.

[GZ1] M. Griebel, G. Zumbusch, *Parallel multigrid in an adaptive PDE solver based on hashing and space-filling curves*, Parallel Computing 25, 1999, 827–843.

[GZ2] M. Griebel, G. Zumbusch, *Parallel adaptive subspace correction schemes with applications to elasticity*, Comput. Meths. Appl. Mechs. Engin. 184, 2000, 303–332.

[Ha1] W. Hackbusch, *Multigrid Methods and Applications*, Springer, New York, 1985.

[Ha2] W. Hackbusch, *Iterative Solution of Large Sparse Systems of Equations*, Springer, New York, 1994.

[HHN] I. Hattendorf, S. Hergarten, H.J. Neugebauer, *Local slope stability analysis*, in: Process Modelling and Landform Evolution, S. Hergarten, H.J. Neugebauer (eds.), Lecture Notes in Earth Sciences, Springer No. 78, 1999, 169–185.

[HKWZ] P. W. Hemker, R. Kettler, P. Wesseling, P. M. de Zeeuw, *Multigrid methods: Development of fast solvers*, Appl. Math. Comput. 13, 1983, 311–326.

[Ki] F. Kiefer, *Multiskalen–Verfahren für Konvektions–Diffusions Probleme* (in German), Dissertation, IAM, Universität Bonn, 2001.

[Kn] S. Knapek, *Matrix–dependent multigrid homogenization for diffusion problems*, SIAM J. Sci. Comput. 20, 1999, 515–533.

[KPV] A. Kunoth, A.-S. Piquemal, J. Vorloeper, *Multilevel preconditioners for discretizations of elliptic boundary value problems — Experiences with different software packages*, IGPM–Preprint #194, RWTH Aachen, July 2000.

[Oe] D. Oeltz, *Algebraische Mehrgitter–Methoden für Systeme partieller Differentialgleichungen* (in German), Diplomarbeit, IAM, Universität Bonn, 2001.

[O] P. Oswald, *Multilevel Finite Element Approximations*, Teubner Skripten zur Numerik, Teubner, Stuttgart, 1994.

[P] C. Pflaum, *Robust convergence of multilevel algorithms for convection-diffusion equations*, SIAM J. Numer. Anal. 37, 2000, 443–469.

[RS] J.W. Ruge, K. Stüben, *Algebraic multigrid*, in: *Multigrid Methods*, S. McCormick (ed.), SIAM, Philadelphia, 1987.

[St] K. Stüben, *Algebraic multigrid (AMG): An introduction with applications*, Technical Report 53, GMD–Forschungszentrum Informationstechnik GmbH, St. Augustin, 1999.

[VW1] P. S. Vassilevski, J. Wang, *Stabilizing the hierachical basis by approximate wavelets. I: Theory*, Numer. Lin. Algebra Appl. 4, 1997, 103–126.

[VW2] P. S. Vassilevski, J. Wang, *Stabilizing the hierachical basis by approximate wavelets, II: Implementation and Numerical Results.* SIAM J. Sci. Comput. 20, 1998, 490–514.

[X] J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Rev. 34, 1992, 581–613.

[Z1] P. M. de Zeeuw, *Matrix–dependent prolongations and restrictions in a black–box multigrid solver*, J. Comput. Appl. Math. 33, 1990, 1–27.

[Z2] P. M. de Zeeuw, *Acceleration of Iterative Methods by Coarse Grid Corrections*, PhD thesis, University of Amsterdam, 1997.