Numerical Algorithms **26:** 111–121, 2001.
© 2001 *Kluwer Academic Publishers. Printed in the Netherlands.*

Multilevel algorithms for finite difference discretizations on sparse grids

Frauke Sprengel

SCAI Institute for Algorithms and Scientific Computing,
GMD German National Research Center for Information Technology,
Schloss Birlinghoven, D-53754 Sankt Augustin, Germany
E-mail: frauke.sprengel@gmd.de

Received 9 June 2000; revised 1 August 2000
Communicated by G. Strang

We propose some algorithms to solve the system of linear equations arising from the finite difference discretization on sparse grids. For this, we will use the multilevel structure of the sparse grid space or its full grid subspaces, respectively.

Keywords: finite differences, sparse grids, multilevel algorithms

AMS subject classification: 65N06, 65N22, 65N55

1. Introduction

Sparse grids and related methods already have a long tradition in numerical quadrature and approximation theory (see, e.g., [2,3,12,16,18]). The reason why they became popular is the following: When applied to d dimensions, $d \geqslant 2$, all numerical methods using regular rectangular grids have one problem in common: *the curse of dimension*. If one refines the grid – for instance, by halving the mesh size n times – the number of grid points grows exponentially with the dimension, i.e., like N^d, where $N = \mathrm{O}(2^n)$ denotes the number of points per direction. One way out is to use *sparse grids*, where the number of points only grows like $N(\log N)^{d-1}$. Under certain conditions on the mixed derivatives of the function, the approximation accuracy degrades only by a logarithmic factor compared with the accuracy achieved on the regular grid with the same N, cf. [7,9,19]. During the last decade, since the introduction of sparse grid methods for elliptic boundary problems by Zenger [19], several authors (e.g., [1,4,7,8,14]) have contributed to this field. Most of these papers were concerned with finite element methods.

Recently, also finite difference methods for sparse grids have been developed [5,15]. For general variable coefficient equations on sparse grids, FD operators allow a more efficient operator evaluation than finite element operators. However, the structure of the FD operators is more complex. The corresponding matrix of the system of linear equations is ill-conditioned. So, it takes (even in the preconditioned version [15]) many iterations of an iterative solver to obtain a solution. The finite difference operator on a sparse grid described in [5,10,15] is operating on the space of piecewise linear functions on the sparse grid. If one considers these spaces for sparse grids of different levels one observes [8,17] that they form a (interpolatory) multiresolution analysis. On the other hand, the sparse grid space is a sum of full grid spaces (see section 2). For the finite element approach, these properties have been used to develop multiplicative subspace correction algorithms (see [6]). Of course, the question comes up how one can use one or both of these subspace properties to develop multilevel-type algorithms also for the finite difference approach.

We will describe two possible types of multilevel-type algorithms for finite difference discretizations on sparse grids. The first one is similar to multiplicative subspace correction algorithms for the finite element case using the full grids contained in the sparse grids. We present several versions of this algorithm and discuss the problems that rise. The second algorithm is a V-cycle of the sparse grid spaces of several levels, when the full grids of one level can be treated in parallel as in [13].

2. Notation

We want to solve elliptic PDEs with a finite difference approach using sparse grids. For our tests, we restrict ourselves to the model problem of Poisson's equation with homogeneous Dirichlet boundary conditions,

$$\begin{aligned} \Delta u &= f \quad \text{in } \Omega, \\ u|_{\delta\Omega} &= 0 \end{aligned} \tag{1}$$

on the cube $\Omega = (0,1)^3$ in 3D and a regular sparse grid.

We will use the usual multi-index notation $\mathbf{m} = (m_1, m_2, m_3) \in \mathbb{N}_0^3$ with $|\mathbf{m}| = m_1+m_2+m_3$. We choose dyadic mesh-widths $\mathbf{h_k} = (h_{k_1}, h_{k_2}, h_{k_3}) \in \mathbb{R}_+^3$ with $h_{k_i} = 2^{-k_i}$ to form dyadic (full) grids

$$\Omega_{\mathbf{k}}^+ = \{\mathbf{x_{k,j}} = \mathbf{j}\mathbf{h_k} = (j_1 h_{k_1}, j_2 h_{k_2}, j_3 h_{k_3})\}$$

and the sparse grid

$$\Omega_\ell^+ = \bigcup_{|\mathbf{k}|=\ell} \Omega_{\mathbf{k}}^+.$$

From the univariate hat function $\varphi(x) = \max(0, 1-|x|)$, we build the trilinear basis functions $\varphi_{\mathbf{k},\mathbf{j}}(\mathbf{x}) = \prod_{i=1}^3 \varphi(x_i/h_{k_i} - j_i)$. The corresponding spaces of piecewise trilinear functions are

$$V_{\mathbf{k}} = \operatorname{span}\{\varphi_{\mathbf{k},\mathbf{j}}\colon \operatorname{supp}(\varphi_{\mathbf{k},\mathbf{j}}) \subset \overline{\Omega}\}$$

for the full grid $\Omega_{\mathbf{k}}^+$ and

$$V_\ell = \sum_{|\mathbf{k}|=\ell} V_{\mathbf{k}}$$

for the sparse grid Ω_ℓ^+. Approximation results for these sparse and full grid spaces can be found, e.g., in [9].

A given continuous function $u \in C_0(\Omega)$ can be approximated by a function $u_\mathbf{k} \in V_\mathbf{k}$ by trilinear interpolation on the full grid $\Omega_\mathbf{k}^+$

$$u_\mathbf{k} = \sum_\mathbf{j} u(\mathbf{x}_{\mathbf{k},\mathbf{j}})\,\varphi_{\mathbf{k},\mathbf{j}}. \tag{2}$$

We do this for all full grids belonging to the sparse grid Ω_ℓ^+ and call the collection of (2) with $|\mathbf{k}| \leqslant \ell$ the *nodal* or *E-representation* of the approximation. The coefficients $u(\mathbf{x}_{\mathbf{k},\mathbf{j}})$ have to coincide if the evaluation points $\mathbf{x}_{\mathbf{k},\mathbf{j}}$ coincide on the different grids. So, a certain consistency is required and the E-representation is redundant. The approximations on all the full grids $\Omega_\mathbf{k}^+$ do not necessarily satisfy this consistency condition. In order to get a consistent representation one can use then hierarchical smoothing (cf. [13] or see section 4).

Another form to represent a given function on a sparse grid is the *hierarchical* or *H-representation* in terms of the hierarchical basis

$$u_\ell = \sum_{|\mathbf{k}|\leqslant\ell,\ \mathbf{j}\text{ odd}} a_{\mathbf{k},\mathbf{j}}\,\varphi_{\mathbf{k},\mathbf{j}}. \tag{3}$$

Here, $\mathbf{j}$ odd has the meaning that either j_i is odd or $k_i = 0$ (i.e., j_i lives on the coarsest grid in x_i-direction).

Of course, one obtains full grid parts (2) as partial sums

$$u_\mathbf{m} = \sum_{\mathbf{k}\leqslant\mathbf{m}} \left(\sum_{\mathbf{j}\text{ odd}} a_{\mathbf{k},\mathbf{j}}\,\varphi_{\mathbf{k},\mathbf{j}}\right) = \sum_{\mathbf{k}\leqslant\mathbf{m}} w_\mathbf{k}, \tag{4}$$

where $w_\mathbf{k}$ denotes the hierarchical surplus on grid $\Omega_\mathbf{k}^+$. For further purpose, we denote by $H_{\mathbf{k},\mathbf{m}}$ the projector of hierarchical surplus, realizing $H_{\mathbf{k},\mathbf{m}}u_\mathbf{m} = w_\mathbf{k}$.

For trilinear functions, one can use fast pyramid algorithms to convert an E-representation into an H-representation, and vice versa (e.g., [5]). This can be done in $\mathrm{O}(N)$ operations where N denotes the total number of degrees of freedom.

We discretize the Laplace operator by finite differences. On full grids $\Omega_\mathbf{k}^+$, this discretized operator $A_\mathbf{k}$ applied to an E-representation is the usual 7-point stencil. On the sparse grid Ω_ℓ^+, things are more complicated. The discretized operator A_ℓ has to be built from a combination of transformations between H- and E-representations and one-dimensional difference operators in each direction (cf. [5,15]). As an additional difficulty it turns out that the resulting matrix is not symmetric. An explicit formula for A_ℓ applied to the H-representation can be found in [10].

Example 2.1. We will apply the algorithms proposed in sections 3 and 4 to the following 3D-problem. Solve (1) with the right-hand side

$$f(\mathbf{x}) = -3\pi^2\left(\prod_{i=1}^{3}\sin\pi x_i + 64\prod_{i=1}^{3}\sin 8\pi x_i\right)$$

starting from the zero function $u_\ell^{(0)} \equiv 0$. We denote by $f_\ell = R_\ell f \in V_\ell$ the piecewise trilinear interpolant of the right-hand side on the sparse grid and by $u_\ell \in V_\ell$ the current approximation of the solution in the sparse grid space.

3. An algorithm with sequential treatment of the full grids

This section is devoted to algorithms similar to the multiplicative subspace correction methods for finite elements on sparse grids as described in [6]. That means, the discretized Laplacian A_ℓ on the sparse grid Ω_ℓ^+ will be approximated by the discretized Laplacians $A_\mathbf{k}$ on the full grids $\Omega_\mathbf{k}^+$. On the different full grids, subproblems with defect-corrected right-hand sides are solved visiting the full grids sequentially in order to obtain a solution of the discretized problem on the whole sparse grid. We propose three versions of such algorithms and present numerical results for one of them (the convergence rates are not so different).

Here and in the sequel, we denote $N_\ell = \#\{\mathbf{k};\ |\mathbf{k}| = \ell\}$, and $j = 1, \ldots, N_\ell$ belongs to a certain $\mathbf{k}$ with $|\mathbf{k}| = \ell$. Because $V_\mathbf{k} \subset V_\ell$ for $|\mathbf{k}| \leqslant \ell$, we can use the natural imbedding $P_{\ell,\mathbf{k}}: V_\mathbf{k} \to V_\ell$ as a prolongation. The restriction $R_{\ell,\mathbf{k}}: V_\ell \to V_\mathbf{k}$ to a full grid space $V_\mathbf{k}$ is realized by trilinear interpolation on the grid $\Omega_\mathbf{k}^+$. Both operations can be carried out easiest in H-representation.

With this, the multiplicative subspace correction algorithm would read as

$$\begin{aligned} v_\ell^{(0)} &= u_\ell^{(i)}, \\ v_\ell^{(j+1)} &= v_\ell^{(j)} - \omega\bigl(P_{\ell,\mathbf{k}}A_\mathbf{k}^{-1}R_{\mathbf{k},\ell}\bigl(A_\ell v_\ell^{(j)} - f_\ell\bigr)\bigr), \quad j = 0, \ldots, N_\ell - 1, \\ u_\ell^{(i+1)} &= v_\ell^{(N)}. \end{aligned} \tag{5}$$

Here and in the following algorithms, we built in damping parameters ω and (later) ω_0. Later we will discuss why this is useful and necessary.

In algorithm (5), the defect correction step is carried out within the loop. But the evaluation of the sparse grid finite difference operator is by far the most expensive step here. So, we may think of doing the defect correction of the right-hand side always with the same residual which is computed before the loop

$$\begin{aligned} v_\ell^{(0)} &= u_\ell^{(i)}, \\ v_\ell^{(j+1)} &= v_\ell^{(j)} - \omega\bigl(P_{\ell,\mathbf{k}}A_\mathbf{k}^{-1}R_{\mathbf{k},\ell}\bigl(A_\ell u_\ell^{(i)} - f_\ell\bigr)\bigr), \quad j = 0, \ldots, N_\ell - 1, \\ u_\ell^{(i+1)} &= u_\ell^{(i)} + \omega_0\bigl(v_\ell^{(N)} - u_\ell^{(i)}\bigr). \end{aligned} \tag{6}$$

One can simplify this further by carrying out the complete defect correction step before the loop what gives the more complicated looking algorithm

$$\begin{aligned} v_\ell^{(0)} &= u_\ell^{(i)}, \\ v_\ell^{(j+1)} &= v_\ell^{(j)} + P_{\ell,\mathbf{k}} R_{\mathbf{k},\ell}\big(u_\ell^{(i)} - v_\ell^{(j)}\big) - \omega\big(P_{\ell,\mathbf{k}} A_{\mathbf{k}}^{-1} R_{\mathbf{k},\ell}\big(A_\ell u_\ell^{(i)} - f_\ell\big)\big), \\ &\qquad j = 0, \ldots, N_\ell - 1, \\ u_\ell^{(i+1)} &= u_\ell^{(i)} + \omega_0\big(v_\ell^{(N)} - u_\ell^{(i)}\big). \end{aligned} \tag{7}$$

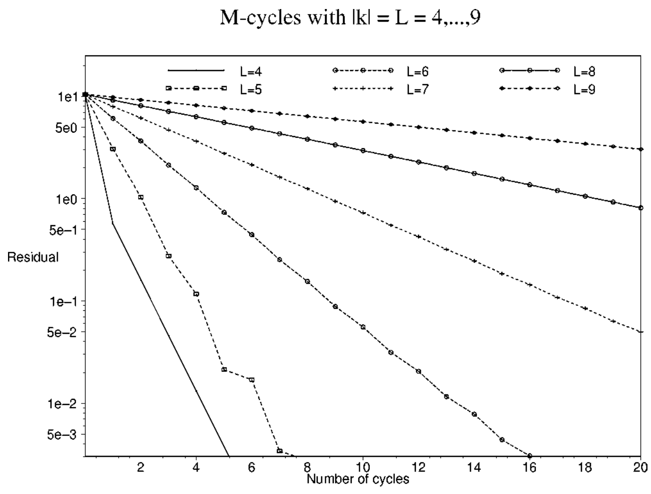
In more algorithmic form, it becomes more clear that the defect correction step is outside the loop:

```
$r_\ell := f_\ell - A_\ell u_\ell$
$u_{\mathbf{k}} := R_{\mathbf{k},\ell} u_\ell$
$g_{\mathbf{k}} := A_{\mathbf{k}} u_{\mathbf{k}} + R_{\mathbf{k},\ell} r_\ell$
$v_\ell := u_\ell$
for $\forall \mathbf{k}$, $|\mathbf{k}| = \ell$
do  $v_{\mathbf{k}}^{\text{old}} := R_{\mathbf{k},\ell} v_\ell$
    solve $A_{\mathbf{k}} v_{\mathbf{k}}^{\text{new}} = g_{\mathbf{k}}$
    $v_\ell := v_\ell + \omega(P_{\ell,\mathbf{k}}(v_{\mathbf{k}}^{\text{new}} - v_{\mathbf{k}}^{\text{old}}))$
enddo
$u_\ell := u_\ell + \omega_0(v_\ell - u_\ell)$.
```

In the following examples, we have set $\omega = 1$ (no damping within the loop) to see what is a proper damping parameter ω_0. The results are given for ω_0 (chosen experimentally) as big as possible such that the algorithm converges. We did not try to really optimize for the best possible damping parameter. The linear systems for the full grids are solved using BiCGStab.

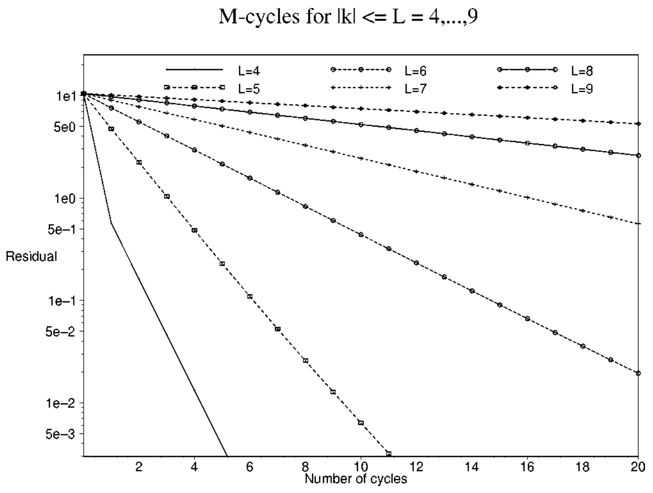
Example 3.1. We apply algorithm (7) to example 2.1.

Level ℓ	Damping ω_0
4	1.000
5	0.667
6	0.385
7	0.215
8	0.110
9	0.056



Example 3.2. We apply algorithm (7) to example 2.1, taking the loop over all levels $|\mathbf{k}| \leqslant \ell$ instead of over $|\mathbf{k}| = \ell$.

Level ℓ	Damping ω_0
4	1.000
5	0.500
6	0.250
7	0.125
8	0.062
9	0.031



We see that already for our well-behaved example, the convergence rate and the necessary damping depend very much on the level ℓ of the sparse grid. The damping parameters become fairly small with growing level. The reason is the following: The finite difference operators for the Laplacian on full grids depend on the grid but not on the point of evaluation. On sparse grids, this is quite different, here the step sizes for finite differences in different directions depend on the evaluation point (cf. [10]). For this reason, the full grid matrices can not be written as Galerkin approximations of the sparse grid matrix. In this sense, the finite difference operators on the full grids are no good approximations of the finite difference operators on the sparse grid. If we have a closer look [10] on the difference operators for the second derivative in x_ν-direction (full grid: $A^\nu_{\mathbf{k}}$, sparse grid: A^ν_ℓ) at the point $\mathbf{x}_{\mathbf{j},\mathbf{m}}$ being hierarchical on grid $\Omega^+_{\mathbf{j}}$, then we see the reason for the damping: Beside restriction and prolongation, we have to scale the row responsible for $\mathbf{x}_{\mathbf{j},\mathbf{m}}$ by the factor $2^{k_\nu - j_\nu - \ell + |\mathbf{j}|}$ to obtain $A^\nu_{\mathbf{k}}$ from A^ν_ℓ. The damping parameters have to meet the needs of all points and all directions, so they have to be small and decrease with growing level ℓ of the sparse grid. In example 3.2, we used the same damping parameter ω_0 for all grids on the different levels, i.e., for an M-cycle with highest level ℓ, ω_0 depends only on ℓ. So we had to use even smaller damping parameters (here we chose: $\omega_0 = 2^{4-\ell}$) than in example 3.1 and we also got a slower convergence.

We do not have such a phenomenon in the Galerkin approach using the hierachical basis or the generating system of the nodal bases for the sparse and the full grids. There, the stiffness matrices for the full grids are submatrices of the stiffness matrix for the sparse grid by construction. In this case [6], the multiplicative subspace correction methods can be interpreted as block iteration methods for the sparse grid stiffness matrix.

4. A V-cycle with parallel treatment of the full grids on a level

In this section, we use the multilevel structure of the sparse grid spaces $V_0,\ V_1,\ V_2,\ldots$ to establish some kind of V-cycle for finite differences on sparse grids. This means, that we use approximations from coarser grids to improve the approximation of the solution on the finest grid. If we would work with a V-cycle in the closer sense of the word we would do iterations with the sparse grid operator A_ℓ and try to improve it with solutions from $A_{\tilde\ell},\ \tilde\ell < \ell$. This approach is discussed elsewhere [10,11].

What we present here is in the spirit of [13, section 5.4] . On each level, we do not iterate with the operators $A_{\tilde\ell}$ on sparse grids but on each of its grids (of this level) separately. Of course, for this approach we need different restriction and prolongation strategies because starting with an E-representation, we obtain a non-consistent representation by correcting on the grids. We make the following choice. For the restriction, we take mean values from the 3 possible finer grids (one per direction, either full weighting or function values). The prolongation is even more delicate. Here, we use the hierarchical smoothing procedure described, e.g., in [13]. This means, that $u_\ell := P_{\ell,\ell}(\{u_{\mathbf{k}},\ |k| = \ell\})$ stands for hierarchical smoothing of the given solutions $u_{\mathbf{k}}$ on level ℓ, following the algorithm

```
**for** all levels $m = \ell - 1(-1)0$
**do** $u_{\mathbf{k}} := (1/3)\sum_{s=1}^{3} R_{\mathbf{k},\mathbf{k}+\mathbf{e}_s} u_{\mathbf{k}+\mathbf{e}_s}\ (\forall\mathbf{k},\ |\mathbf{k}| = m)$
    $w_{\mathbf{k}} := H_{\mathbf{k},\mathbf{k}}\, u_{\mathbf{k}}\ (\forall\mathbf{k},\ |\mathbf{k}| = m+1)$        (8)
**enddo**
$u_\ell := u_0 + \sum_{0<|\mathbf{k}|\leqslant\ell} w_{\mathbf{k}}$,
```

where the projection $H_{\mathbf{k},\mathbf{k}}$ is described in section 2. The hierarchical smoothing routine delivers a consistent approximation u_ℓ for the sparse grid Ω_ℓ^+. If we use the notation $P_{\ell+1,\ell}(\{u_{\mathbf{k}}, |k| = \ell\})$ we mean hierarchical smoothing on level ℓ and then interpolation to grid $\Omega_{\ell+1}^+$, i.e., in the last line of algorithm (8), the left-hand side has to be replaced by $u_{\ell+1}$. With this, we obtain a consistent approximation $u_{\ell_{\max}}$ after each cycle.

Note, that on each level, we apply the iterative solver on all full grids separately (this means that on each level the iterations can be done in *parallel*). Because of the relative inconsistency, it is not necessary to accurately solve the systems on each full grid, a few iterations (pre- or postrelaxations) are sufficient. This is quite different from the algorithms presented in the previous section. There, the full grids are treated sequentially and is was necessary to really *solve* the systems on the full grids sufficiently accurate to obtain convergence.

Below we describe the full algorithm, given a lowest level $\ell_{\min}$ and a highest level $\ell_{\max}$ of sparse grids, the number ν_1 of prerelaxations and ν_2 of postrelaxations, and a damping parameter ω_0. We start with a current approximation $u_{\ell_{\max}}^{\text{old}} \in V_{\ell_{\max}}$. Then, one cycle is given by

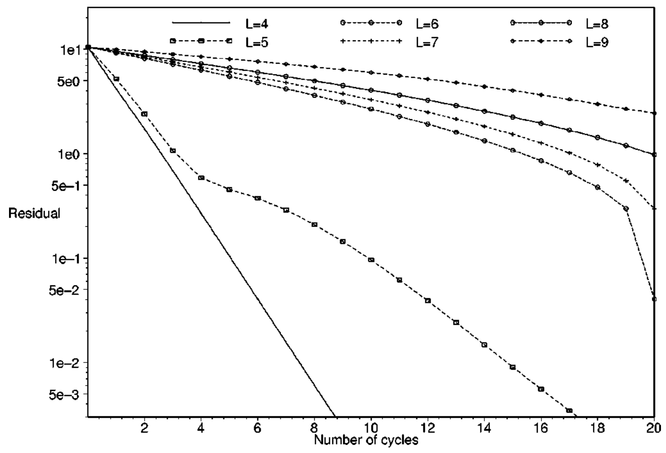
```
$r_{\ell_{\max}} := f_{\ell_{\max}} - A_{\ell_{\max}} u^{\text{old}}_{\ell_{\max}}$
$u_{\mathbf{k}} := R_{\mathbf{k},\ell_{\max}} u_{\ell_{\max}}$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell_{\max})$
$g_{\mathbf{k}} := A_{\mathbf{k}} u_{\mathbf{k}} + R_{\mathbf{k},\ell_{\max}} r_{\ell_{\max}}$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell_{\max})$
**for** all levels $\ell = \ell_{\max}(-1)\ell_{\min} + 1$
**do** $\nu_1$ steps of a smoother for $A_{\mathbf{k}} u_{\mathbf{k}} = g_{\mathbf{k}}$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell)$
    **for** $\forall \mathbf{k}, |\mathbf{k}| = \ell - 1$
    **do** $u_{\mathbf{k}} := (1/3) \sum_{s=1}^{3} R_{\mathbf{k},\mathbf{k}+\mathbf{e}_s} u_{\mathbf{k}+\mathbf{e}_s}$
        $g_{\mathbf{k}} := A_{\mathbf{k}} u_{\mathbf{k}} + R_{\mathbf{k},\ell_{\max}} r_{\ell_{\max}}$                    (9)
        $u^{\text{old}}_{\mathbf{k}} := u_{\mathbf{k}}$
    **enddo**
**enddo**
**Solve** on level $\ell_{\min}$: $A_{\mathbf{k}} u_{\mathbf{k}} = g_{\mathbf{k}}$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell_{\min})$
**for** all levels $\ell = \ell_{\min}(+1)\ell_{\max} - 1$
**do** $u_{\mathbf{k}} := u_{\mathbf{k}} + R_{\mathbf{k},\ell} P_{\ell+1,\ell}(\{(u_{\mathbf{n}} - u^{\text{old}}_{\mathbf{n}}), |\mathbf{n}| = \ell\})$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell + 1)$
    $\nu_2$ steps of a smoother for $A_{\mathbf{k}} u_{\mathbf{k}} = g_{\mathbf{k}}$  $(\forall \mathbf{k}, |\mathbf{k}| = \ell + 1)$
**enddo**
$u_{\ell_{\max}} := P_{\ell_{\max},\ell_{\max}}(\{u_{\mathbf{k}}, |\mathbf{k}| = \ell_{\max}\})$
$u_{\ell_{\max}} := u^{\text{old}}_{\ell_{\max}} + \omega_0 (u_{\ell_{\max}} - u^{\text{old}}_{\ell_{\max}})$
```

In the following examples, we used Gauß–Seidel iteration as a smoother and BiCGStab as a solver on the coarsest grids. The same damping parameter ω_0 has been used after each cycle. Again we need this damping parameter which has to go down with the level for exactly the same reason as discussed in the previous section.

Example 4.1. We start with a two-level algorithm using only two levels $\ell_{\max}$ and $\ell_{\max}-1$ in algorithm (9). We use it with $\nu_1 = 1$ prerelaxation and $\nu_2 = 1$ postrelaxation obtaining the following damping parameters and convergence behaviour.

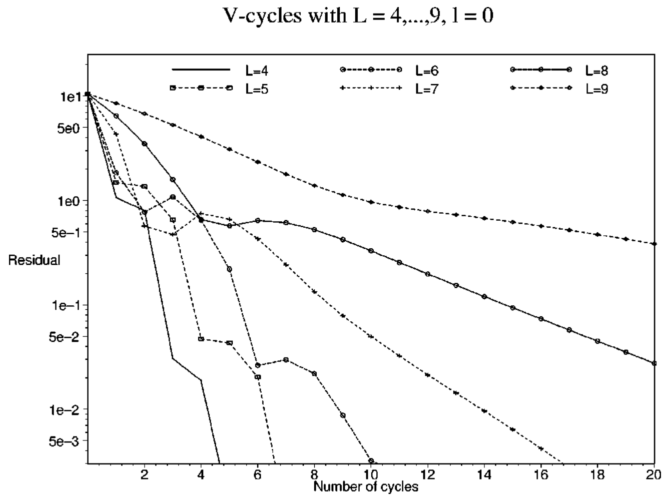
Level $\ell_{\max}$	Damping ω_0
4	1.000
5	0.350
6	0.085
7	0.080
8	0.070
9	0.040

2-cycles with L = 4,...,9, l = L–1

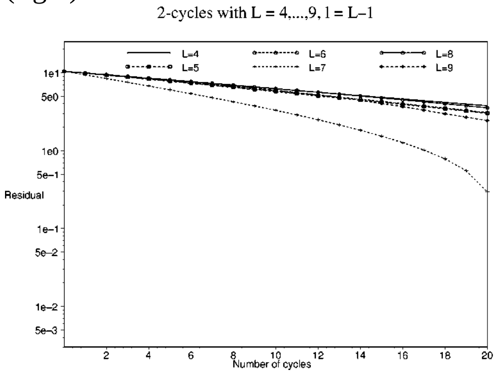


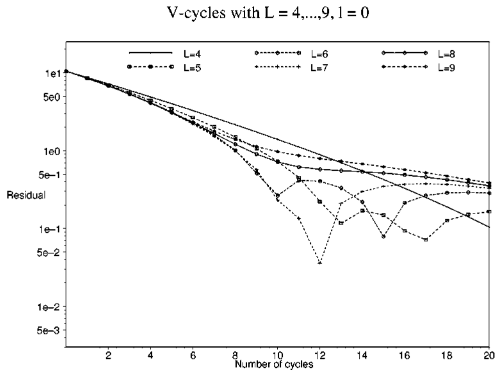
Example 4.2. In the following example, we apply a V-cycle using all levels from $\ell_{\max}$ down to $\ell_{\min} = 0$ in algorithm (9). We use it with no ($\nu_1 = 0$) prerelaxation and $\nu_2 = 1$ postrelaxation. Then we obtain the following damping parameters and convergence behaviour.

Level $\ell_{\max}$	Damping ω_0
4	0.900
5	0.750
6	0.600
7	0.450
8	0.300
9	0.150



Example 4.3. For better comparison, we now use the same damping parameter for all levels (the one for the highest level $\ell = 9$) and repeat the two-level algorithm from example 4.1 with $\omega_0 = 0.040$ (left) and the V-cycles from example 4.2 with $\omega_0 = 0.150$ (right).





The convergence behaviour depends much on the choice of a good damping parameter which has to go down with growing level. Again, we see that the speed of convergence slows down with growing level (as it seems mainly because of the necessary damping).

At last we give a comparison for the algorithms under consideration and the multiplicative cycle with the adapted full grid operators from [10]. These adapted full grid operators are Galerkin approximations of the sparse grid operator. We compare the number of cycles and the computing times (relative to the fastest algorithm) necessary for level $\ell_{\max} = 9$ to reach a residual which is (in L_2-norm) less than $5 \cdot 10^{-3}$ (i.e., approximately the discretization error for level 9). We did not exploit parallelization for the V-cycle in example 4.2.

Algorithms	Cycles	Time units
Example 3.1 (one-level multiplicative algorithm)	123	3.25
Example 3.2 (multilevel multiplicative algorithm)	223	8.44
Example 4.2 (V-cycle, $\nu_1 = 0,\ \nu_2 = 1$)	60	1.00
One-level multiplicative algorithm (26) from [10]	28	23.87
Multilevel multiplicative algorithm (27) from [10]	17	23.22

We see, that the use of adapted full grid operators results in less iteration cycles, compared to the multiplicative algorithms with damped finite difference operators on the full grids. But the algorithms presented here in section 3 are less expensive per cycle so that they are more efficient. The V-cycle from example 4.2 is the best of our algorithms. We need the least number of cycles (compared only with the algorithms treated in this paper) and the best time (compared with the algorithms from this paper and from [10]).

References

[1] H.-J. Bungartz and T. Dornseifer, Sparse grids: Recent developments for elliptic partial differential equations, in: *Multigrid Methods V*, eds. W. Hackbusch and G. Wittum, Lecture Notes in Computational Science and Engineering, Vol 3 (Springer, Berlin, 1998).

[2] F.-J. Delvos and H. Posdorf, Nth order blending, in: *Constructive Theory of Functions of Several Variables*, eds. W. Schempp and K. Zeller (Springer, Berlin, 1977) pp. 53–64.

[3] F.-J. Delvos and W. Schempp, *Boolean Methods in Interpolation and Approximation*, Pitman Research Notes in Mathematics Series, Vol. 230 (Longman Scientific & Technical, Harlow, 1989).

[4] M. Griebel, *Multilevelmethoden als Iterationsverfahren über Erzeugendensystemen*, Teubner Skripten zur Numerik (Teubner, Stuttgart, 1994).

[5] M. Griebel, Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences, Computing 61 (1998) 151–179.

[6] M. Griebel and P. Oswald, Tensor product type subspace splitting and multilevel iterative methods for anisotropic problems, Adv. Comput. Math. 4 (1995) 171–206.

[7] M. Griebel, M. Schneider and C. Zenger, A combination technique for the solution of sparse grid problems, in: *Iterative Methods in Linear Algebra*, eds. R. Beauwens and P. de Groen, IMACS (Elsevier, Amsterdam, 1992) pp. 263–281.

[8] P. W. Hemker, Sparse-grid finite-volume multigrid for 3D-problems, Adv. Comput. Math. 4 (1995) 83–110.

[9] P. W. Hemker and C. Pflaum, Approximation on partially ordered sets of regular grids, Appl. Numer. Math. 25 (1997) 55–87.

[10] P.W. Hemker and F. Sprengel, On the representation of functions and finite difference operators on adaptive dyadic grids (2000) submitted (CWI Report MAS-R9933).

[11] P.W. Hemker and F. Sprengel, Experience with the solution of a finite difference discretization on sparse grids (2000) submitted (GMD report 98).

[12] N.M. Korobov, Approximate calculation of multiple integrals with the aid of methods in the theory of numbers, Dokl. Akad. Nauk SSSR 115 (1957) 1062–1065 (in Russian).

[13] J. Noordmans and P.W. Hemker, Convergence results for 3D sparse grid approaches, Numer. Linear Algebra Appl. 5 (1999) 363–376.

[14] P. Oswald, *Multilevel Finite Element Approximation. Theory and Applications*, Teubner Skripten zur Numerik (Teubner, Stuttgart, 1994).

[15] T. Schiekofer, Die Methode der Finiten Differenzen auf dünnen Gittern zur Lösung elliptischer und parabolischer partieller Differentialgleichungen, Ph.D. thesis, Universität Bonn (1998).

[16] S.A. Smolyak, Quadrature and interpolation formulas the classes W_s^a and E_s^a, Dokl. Akad. Nauk SSSR 131 (1960) 1028–1031 (in Russian); English transl.: Soviet Math. Dokl. 1 (1963) 384–387.

[17] F. Sprengel, Interpolation and wavelet decomposition of multivariate periodic functions, Ph.D. thesis, University of Rostock (1997).

[18] V.N. Temlyakov, *Approximation of Periodic Functions* (Nova Science, New York, 1993).

[19] C. Zenger, Sparse grids, in: *Parallel Algorithms for Partial Differential Equations*, ed. W. Hackbusch, Notes on Numerical Fluid Mechanics, Vol. 31 (Vieweg, Braunschweig, 1991) pp. 297–301.