NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2010; **1**:1 *Prepared using* nlaauth.cls *[Version: 2002/09/18 v1.02]*

Validation and Optimization of the Convergence Rate on Semi-structured Meshes using the LFA

Björn Gmeiner[1,*], Tobias Gradl[1], Francisco Gaspar[2], Ulrich Rüde[1]

[1] *Department of Computer Science, System Simulation, Cauerstr. 6, 91058 Erlangen, Germany*
[2] *Department of Applied Mathematics, University of Zaragoza. Maria de luna 3, 50018 Zaragoza, Spain*

SUMMARY

The local Fourier analysis (LFA) is an excellent tool to estimate asymptotic convergence factors for iterative solvers. The assumptions for the LFA incorporate structured grids, which makes the LFA hard to apply for arbitrary finite element meshes. Semi-structured meshes at least partly fulfill this requirement. In this paper we compare measured convergence rates from the finite element program Hierarchical Hybrid Grids (HHG) with predictions of the LFA for a two-grid multigrid cycle. We show a good accordance for four iterative smoothers between a structured region of HHG and the LFA for differently shaped tetrahedral elements. Further we want to demonstrate how predictions of these regions can improve the overall convergence by redistributing the smoothing steps over the domain.

KEY WORDS: multigrid, local fourier analysis, convergence rates, smoothers

1. Introduction and terminology

Although Multigrid (MG) methods have optimal computational complexity for solving many numerical problems, their performance in solving individual problems varies. The speed of convergence towards a solution depends on the numerical properties of the underlying problem, e. g. the type of a differential equation and the method used for discretizing the equation. Besides that, the user can choose from a variety of algorithms for the components of the MG method, most prominently the smoother, the restriction, and the prolongation. Choosing the appropriate components for a specific problem has a great impact on the solver performance, too.

The convergence speed of iterative solvers like MG can be measured with the convergence rates

$$\rho_i = \frac{e^i}{e^{i-1}} \text{ and} \tag{1}$$

$$\rho = \rho_i \text{ for } i \to \infty\,, \tag{2}$$

*Correspondence to: Björn Gmeiner, Department of Computer Science, System Simulation, Cauerstr. 6, 91058 Erlangen, Germany, bjoern.gmeiner@cs.fau.de

where e^i is the error norm after the ith iteration of the solver, and ρ is the asymptotic convergence rate reached after a large number of iterations. The error is the difference between the numerical solution and the solver's current approximation of that solution. For determining the convergence rate in a production run of an iterative solver, i. e. when the numerical solution is not known in advance, the residual norm can be substituted for the error norm.

The programmers of MG solvers need means for verifying the quality of their implementations. One means is comparing their convergence rates to those of other implementations. For standard cases, like 9-point finite difference discretizations of elliptic partial differential equations (PDEs) on Cartesian grids, the convergence rates that can be achieved by MG solvers are well known and reported in the literature. The authors of the present paper are working with the Hierarchical Hybrid Grids (HHG) solver, which operates on 15-point stencils resulting from a finite element (FE) discretization on semi-structured three-dimensional tetrahedral meshes. To verify the convergence rates achieved with this more unusual discretization, the local Fourier analysis (LFA) [3] is a useful tool.

While MG can work with unstructured meshes, the LFA assumes a structured grid. Thus, the first goal of this paper is to show that the LFA confirms the convergence rates achieved by HHG on structured grids with different anisotropies.

When unstructured FE meshes are used, the convergence rate varies across the domain, depending on the shapes of the individual elements. Numerical properties of the underlying problem, like singularities, can cause locally varying convergence rates, too. The second goal of this paper is to present strategies for locally adapting a MG solver to varying convergence rates. It will turn out that the semi-structured meshes used in HHG help in this process. There are several strategies for implementing local convergence rate adaptivity. One of them is, again, the LFA.

The model problem used throughout this paper is the Laplace equation with homogeneous Dirichlet boundaries

$$\begin{aligned} \Delta u &= 0 \text{ in } \Omega\,, \\ u &= 0 \text{ on } \partial\Omega\,, \end{aligned} \tag{3}$$

where u is the solution.

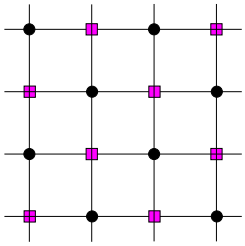
Before the paper presents the numerical experiments and their results, it introduces the tools used in the experiments. Section 2 elaborates on the role of the smoother in a MG method and introduces the smoothers used in the experiments. Sections 3 and 4 provide basic information on HHG and the LFA. In Section 5, we first compare the convergence rates predicted by the LFA to the ones measured with HHG for different element shapes. After that, we show two possibilities for improving the convergence rate on semi-structured meshes.

2. Smoothers for multigrid algorithms

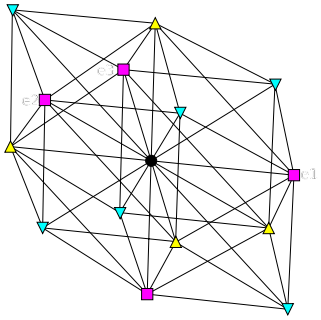
In a multigrid algorithm, the smoother has the task of reducing the high-frequent error components, while the coarse grid correction reduces the low-frequents components. Both smoother and coarse grid correction can be tuned to optimally suit the numerical problem, i. e. the differential equation and its discretization. In this paper, we focus on optimizing the smoother for anisotropic discretizations. We compare the widely used Jacobi and Gauss-Seidel smoothers to a four-color smoother and a line smoother. Besides choosing the smoother type,

Figure 1. Stencil colorings.

(a) Coloring of a 5-point stencil.



(b) Coloring of a 15-point stencil.



two more parameters can be changed to achieve optimal smoothing results: the under-/over-relaxation parameter ω and the number of pre-/post-smoothing steps per multigrid cycle ν_1 and ν_2. These parameters can even be adapted locally, if the numerical properties change strongly across the domain. Implementing local adaptivity can be quite easy and very effective, as Section 5 will show. The remainder of this section introduces the investigated smoothers.

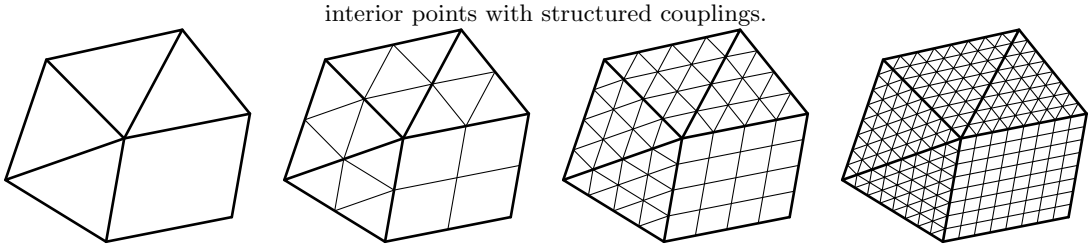
In the *Jacobi* algorithm, all updated values only depend on values from the previous iteration. On the one hand, this allows for excellent parallelization, on the other hand, it requires more memory to keep all values from the previous iteration. In a straight-forward implementation, the storage for two times the number of unknowns is required. The Jacobi algorithm is fast, but it has a bad convergence rate.

In a *Gauss-Seidel* iteration, values that have already been updated in the current iteration are used in the same iteration for updating dependent values. Thus, the unknowns have to be stored only once. If the grid points are updated in a *lexicogrphical order*, a grid point update depends on data from the directly preceding update. This data dependency inhibits a straight-forward parallelization.

The Gauss-Seidel algorithm usually smoothes much better than the Jacobi algorithm, but, due to its data dependencies, it is potentially much slower on current hardware. The data dependencies can be eliminated by grouping the grid points into "colors". Points of the same color are not connected to each other, so they can be updated in parallel. The discretization determines the required number of colors. In a 2D rectangular grid with a 5-point discretization stencil, for example, two colors are needed (cf. Fig. 1(a)). The 15-point stencil of HHG needs *four colors* (cf. Fig. 1(b)). Note that the relaxation parameter ω can be set individually for each of the colors.

Instead of updating single unknowns like in the previous cases, we can also update a *line of unknowns* at a time. Therefore, we have to solve a tridiagonal system of equations. This can be achieved either by repeated smoothing or by using a direct solver, like a modified Gauss elimination, applied to the line. While the lines can be updated in different orders, we will restrict ourselves to the lexicographic ordering in the remainder of this paper.

Figure 2. Regular refinement example for a two-dimensional input mesh. Beginning with the input mesh on the left, each successive level of refinement creates a new mesh that has a larger number of interior points with structured couplings.



3. Hierarchical Hybrid Grids

For solving elliptic PDEs, FE methods are often preferred over other discretization schemes, because they permit flexible, unstructured meshes. Among the MG methods, algebraic MG [6] also supports unstructured meshes automatically. Geometric MG, in contrast, relies on a given hierarchy of nested meshes. On the other hand, geometric MG achieves a significantly higher performance than algebraic MG in terms of unknowns computed per second.

The Hierarchical Hybrid Grids (HHG) software framework [1, 2] is designed to close this gap between FE flexibility and geometric MG performance by using a compromise between structured and unstructured grids. A coarse input FE mesh is organized into the grid primitives vertices, edges, faces, and volumes. The primitives are then refined in a structured way, as indicated in Figure 2.

The HHG data layout preserves the flexibility of unstructured meshes, while the regular internal structure of the primitives allows for an efficient implementation on current computer architectures, especially on parallel computers. In parallel runs on up to 16 384 cores HHG has demonstrated excellent performance on solving linear systems with up to $3{\cdot}10^{11}$ unknowns. The semi-structured mesh also subserves the local adaption of the smoothing parameters needed in Section 5.3.

4. Local Fourier analysis

It is well-known that LFA is a very useful tool to predict with high accuracy the asymptotic convergence factors of MG methods. For this reason, it is widely used to design efficient algorithms by choosing suitable MG components. This section gives a basic introduction into LFA and explains how it can be used to estimate the asymptotic convergence rate ρ defined by (2).

Given $\mathbf{h} = (h_1, h_2, h_3)$ a grid spacing, let $\mathcal{T}_h$ be a regular tetrahedral grid on a fixed coarse tetrahedron $\mathcal{T}$ and let $\{\mathbf{e}'_1, \mathbf{e}'_2, \mathbf{e}'_3\}$ be unit vectors indicating the direction of three of the edges of $\mathcal{T}$; see Fig. 1(b). We extend $\mathcal{T}_h$ to the infinite grid

$$G_h = \{\mathbf{x}' = x'_1\mathbf{e}'_1 + x'_2\mathbf{e}'_2 + x'_3\mathbf{e}'_3 |\, x'_i = k_i h_i, k_i \in \mathbf{Z}, i = 1, 2, 3\},$$

such that $\mathcal{T}_h = G_h \cap \mathcal{T}$. It is crucial to write the Fourier frequencies as $\boldsymbol{\theta}'' = \theta''_1\mathbf{e}''_1 + \theta''_2\mathbf{e}''_2 + \theta''_3\mathbf{e}''_3$, where $\{\mathbf{e}''_1, \mathbf{e}''_2, \mathbf{e}''_3\}$ is the reciprocal basis to $\{\mathbf{e}'_1, \mathbf{e}'_2, \mathbf{e}'_3\}$, i. e. $\mathbf{e}'_i \cdot \mathbf{e}''_j = \delta_{ij}$. Every bounded grid

function $u_h(\mathbf{x}')$ defined on G_h can be written as a formal linear combination of the discrete exponential functions called Fourier modes

$$\varphi_h(\boldsymbol{\theta}'', \mathbf{x}') = \mathrm{e}^{i(\theta_1'' x_1' + \theta_2'' x_2' + \theta_3'' x_3')}, \tag{4}$$

where the Fourier frequencies $\boldsymbol{\theta}'' = \theta_1'' \mathbf{e}_1'' + \theta_2'' \mathbf{e}_2'' + \theta_3'' \mathbf{e}_3''$ vary continuously in $\mathbf{R}^3$. Since $\varphi_h(\tilde{\boldsymbol{\theta}}'', \mathbf{x}') = \varphi_h(\boldsymbol{\theta}'', \mathbf{x}')$ for all $\mathbf{x}' \in G_h$ if and only if $\tilde{\theta}_i'' = \theta_i = (\mathrm{mod}\, 2\pi/h_i), i = 1, 2, 3$, the Fourier modes with associated frequency $\boldsymbol{\theta}''$ such that for some component $|\theta_i''| \geq \pi/h_i$, are not visible on G_h. Therefore, the Fourier space $\mathcal{F}(G_h) = \mathrm{span}\{\varphi_h(\boldsymbol{\theta}'', \cdot) \,|\, \boldsymbol{\theta}'' = (\theta_1'', \theta_2'', \theta_3'') \in \boldsymbol{\Theta}_h\}$, with $\boldsymbol{\Theta}_h = (-\pi/h_1, \pi/h_1] \times (-\pi/h_2, \pi/h_2] \times (-\pi/h_3, \pi/h_3]$, contains any bounded infinite grid function on G_h. If regular refinement is applied, we are considering the case of standard multigrid coarsening, $(H_1, H_2, H_3) = (2h_1, 2h_2, 2h_3)$, and therefore the infinite coarse grid G_H is defined as

$$G_H = \{x_1' \mathbf{e}_1' + x_2' \mathbf{e}_2' + x_3' \mathbf{e}_3' |\, x_i' = 2k_i h_i, k_i \in \mathbf{Z}, i = 1, 2, 3\}.$$

We distinguish high and low frequencies on G_h with respect to G_H. We define the subset $\boldsymbol{\Theta}_H$ of low frequencies as $\boldsymbol{\Theta}_H = (-\pi/H_1, \pi/H_1] \times (-\pi/H_2, \pi/H_2] \times (-\pi/H_3, \pi/H_3]$, and the subset of high frequencies $\boldsymbol{\Theta}_h \setminus \boldsymbol{\Theta}_H$. This definition is based on the fact that only low frequencies are visible on the coarse grid G_H. Each high-frequency component coincides with a certain low-frequency component on G_H. In particular we have $\varphi_h(\tilde{\boldsymbol{\theta}}'', \cdot) = \varphi_h(\boldsymbol{\theta}'', \cdot)$ on G_H with $\tilde{\theta}_i'' = \theta_i''(\mathrm{mod}\, \pi/h_i), i = 1, 2, 3$.

The discrete operators considered here are assumed to be linear with constant coefficients. Neglecting boundary conditions and/or connections with other neighboring tetrahedra on the coarsest grid, the discrete problem $L_h u_h = f_h$ can be extended to the whole grid G_h. For a fixed grid point $\mathbf{x}' = (x_1', x_2', x_3') \in G_h$, the corresponding equation reads in stencil notation [4] as

$$L_h u_h(\mathbf{x}') = \sum_{\mathbf{k} \in \mathcal{I}} s_{\mathbf{k}} u_h(x_1' + k_1 h_1, x_2' + k_2 h_2, x_3' + k_3 h_3) = f_h(\mathbf{x}'), \tag{5}$$

where $s_{\mathbf{k}} \in \mathbf{R}$ are constant coefficients and $\mathcal{I} \subset \mathbf{Z}^3$ is a finite index set. From (5), it can be deduced that the grid-functions $\varphi_h(\boldsymbol{\theta}'', \mathbf{x}')$ given in (4) are formal eigenfunctions of the discrete operator L_h. More precisely, the relation $L_h \varphi_h(\boldsymbol{\theta}'', \mathbf{x}') = \widehat{L_h}(\boldsymbol{\theta}'') \varphi_h(\boldsymbol{\theta}'', \mathbf{x}')$ holds, where

$$\widehat{L_h}(\boldsymbol{\theta}'') = \sum_{\mathbf{k} \in \mathcal{I}} s_{\mathbf{k}} \mathrm{e}^{i(\theta_1'' k_1 h_1 + \theta_2'' k_2 h_2 + \theta_3'' k_3 h_3)}$$

is the corresponding eigenvalue or Fourier symbol of L_h. The most common approaches of LFA are the Fourier one-grid (smoothing) and two-grid analysis. In order to investigate the interplay between relaxation and coarse grid correction, which is crucial for an efficient multigrid method, it is convenient to perform a two-grid analysis which takes into account the effect of transfer operators. Let u_h^m be an approximation of u_h. The error $e_h^m = u_h^m - u_h$ is transformed by a two-grid cycle as $e_h^{m+1} = M_h^H e_h^m$, where $M_h^H = S_h^{\nu_2} K_h^H S_h^{\nu_1}$ is the two-grid operator, $K_h^H = I_h - P_H^h (L_H)^{-1} R_h^H L_h$ the coarse grid correction operator and S_h is a smoothing operator on G_h with ν_1 and ν_2 indicating the number of pre- and post-smoothing steps, respectively. In the definition of K_h^H, L_H is the coarse grid operator and P_H^h, R_h^H are transfer operators from coarse to fine grids and vice versa. The two-grid analysis is the basis for the classical asymptotic multigrid convergence estimates, and the spectral radius $\rho(M_h^H)$ of the operator M_h^H indicates the asymptotic convergence factor of the two-grid method.

For simplicity in notation, in the following we will use $\mathbf{x} = (x_1, x_2, x_3)$ and $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ as the coordinate vectors in the bases $\{\mathbf{e}'_1, \mathbf{e}'_2, \mathbf{e}'_3\}$ and $\{\mathbf{e}''_1, \mathbf{e}''_2, \mathbf{e}''_3\}$, respectively. The Fourier space $\mathcal{F}(G_h)$ is decomposed into eight-dimensional subspaces $\mathcal{F}^8(\boldsymbol{\theta}^{000})$, known as $2h$-harmonics, which consist of one low-frequency harmonic $\boldsymbol{\theta}^{000} \in \boldsymbol{\Theta}_H$ with coordinates $\boldsymbol{\theta}^{000} = (\theta_1^{000}, \theta_2^{000}, \theta_3^{000})$ and seven corresponding high-frequency harmonics in $\boldsymbol{\Theta}_h \setminus \boldsymbol{\Theta}_H$:

$$\mathcal{F}^8(\boldsymbol{\theta}^{000}) = \operatorname{span}\{\varphi_h(\boldsymbol{\theta}^{\alpha_1\alpha_2\alpha_3}, \cdot) \mid \quad \alpha_1, \alpha_2, \alpha_3 \in \{0, 1\}\} \text{ with } \boldsymbol{\theta}^{000} \in \boldsymbol{\Theta}_H,$$
$$\boldsymbol{\theta}^{\alpha_1\alpha_2\alpha_3} = \boldsymbol{\theta}^{000} - (\alpha_1 \operatorname{sign}(\theta_1^{000})\pi/h_1, \alpha_2 \operatorname{sign}(\theta_2^{000})\pi/h_2, \alpha_3 \operatorname{sign}(\theta_3^{000})\pi/h_3).$$

In order to ensure that we deal with nonsingular Fourier symbols $\widehat{L_h}(\boldsymbol{\theta})$ and $\widehat{L_H}(2\boldsymbol{\theta})$, we restrict our considerations to the subspace

$$\mathcal{F}(G_h) \setminus \bigcup_{\boldsymbol{\theta}^{000} \in \boldsymbol{\Psi}} \mathcal{F}^8(\boldsymbol{\theta}^{000}),$$

with $\boldsymbol{\Psi} = \{\boldsymbol{\theta}^{000} \in \boldsymbol{\Theta}_H \,|\, \widehat{L_H}(2\boldsymbol{\theta}^{000}) = 0 \text{ or } \widehat{L_h}(\boldsymbol{\theta}^{\alpha_1\alpha_2\alpha_3}) = 0, \ \alpha_1, \alpha_2, \alpha_3 \in \{0, 1\}\}$. The crucial observation is that the coarse grid correction operator K_h^H leaves the spaces of 2h-harmonics invariant for an arbitrary Fourier frequency $\boldsymbol{\theta}^{000} \in \tilde{\boldsymbol{\Theta}}_H = \boldsymbol{\Theta}_H \setminus \boldsymbol{\Psi}$, i. e. $K_h^H : \mathcal{F}^8(\boldsymbol{\theta}^{000}) \longrightarrow \mathcal{F}^8(\boldsymbol{\theta}^{000})$. This same invariance property is true for the smoothers S_h presented in Section 2, i. e. $S_h : \mathcal{F}^8(\boldsymbol{\theta}^{000}) \longrightarrow \mathcal{F}^8(\boldsymbol{\theta}^{000})$. Therefore, the two-grid operator $M_h^H = S_h^{\nu_2} C_h^H S_h^{\nu_1}$ also leaves the $2h$-harmonic subspaces invariant. Hence, M_h^H is equivalent to a block-diagonal matrix consisting of 8×8 blocks denoted by $\widehat{M_h^H}(\boldsymbol{\theta}^{000}) = M_h^H \,|_{\mathcal{F}^8(\boldsymbol{\theta}^{000})}$, with $\boldsymbol{\theta}^{000} \in \tilde{\boldsymbol{\Theta}}_H$. For this reason, we can determine the spectral radius $\rho(M_h^H)$ by calculating the spectral radius of (8×8)-matrices

$$\rho(M_h^H) = \max_{\boldsymbol{\theta}^{000} \in \tilde{\boldsymbol{\Theta}}_H} \rho(\widehat{M_h^H}(\boldsymbol{\theta}^{000})). \tag{6}$$

5. Numerical experiments and results

5.1. Convergence rate measurement with HHG

The intention of this section is to describe the settings for HHG that are used to measure the asymptotic convergence rates in the following sections. The Laplace equation (3) is discretized on a single coarse grid tetrahedron which has been regularly refined 8 times. The solution u is initialized with random values between -10^6 and 10^6. The errors e_i for determining the convergence rates are measured with the discrete L2 norm.

The LFA assumes an infinitely large regular domain. In practice, such a domain can be approximated by a regular grid with a very large number of unknowns. In experiments, 8 levels of refinement, resulting in $3.3 \cdot 10^5$ unknowns, turned out to be sufficient. The LFA predictions have been calculated for a two-grid cycle. Performing the same cycle in HHG would require a high computational effort on the coarse grid. Prior numerical experiments showed, however, that the convergence rates of a W-cycle, which is much faster, are so close to those of a two-grid cycle that we can use the W-cycle in our experiments.

During the first 20 MG cycles the convergence rate increases strongly. After some more cycles it stays almost constant (cf. Fig. 3(b)). The asymptotic convergence rates are recorded after 150 cycles.

Table I. Tetrahedrons used for comprising HHG and LFA. The needle and wedge types are illustrated.

	$\mathbf{e}_1$	$\mathbf{e}_2$	$\mathbf{e}_3$
Regular	(1, 0, 0)	(0.5, 0.866, 0.0)	(0.5, 0.289, 0.816)
Unit	(1, 0, 0)	(0, 1, 0)	(0, 0, 1)
Needle 1	(1, 0, 0)	(0.5, 0.866, 0)	(0.5, 0.289, 5)
Needle 2	(1, 0, 0)	(0.5, 0.866, 0)	(0.5, 0.289, 10)
Wedge 1	(0.5, 0, 0)	(0.25, 0.968, 0)	(0.25, 0.451, 0.856)
Wedge 2	(0.25, 0, 0)	(0.125, 0.992, 0)	(0.125, 0.488, 0.863)

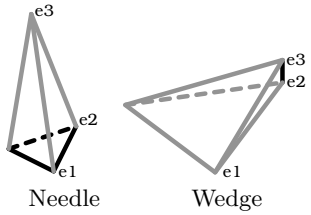


Table II. Comparison of asymptotic convergence rates between LFA and HHG for $\nu = (1, 0)$

	Regular	Unit	Needle 1	Needle 2	Wedge 1	Wedge 2
Jacobi (LFA)	0.640	0.760	0.986	0.996	0.817	0.940
Jacobi (HHG)	0.637	0.756	0.980	0.991	0.812	0.934
Lex. GS (LFA)	0.434	0.565	0.965	0.991	0.640	0.861
Lex. GS (HHG)	0.427	0.566	0.959	0.985	0.634	0.856
4-Color GS (LFA)	0.406	0.607	0.970	0.991	0.623	0.862
4-Color GS (HHG)	0.389	0.550	0.959	0.985	0.612	0.855
Line-wise (LFA)	0.423	0.541	0.943	0.985	0.324	0.327
Line-wise (HHG)	0.417	0.537	0.937	0.979	0.318	0.320

5.2. Asymptotic convergence rates of HHG compared to LFA predictions

To compare the convergence rates measured with HHG with the ones predicted by the LFA, six test cases are selected. The rates are compared on a grid of regular tetrahedrons, on a grid of unit tetrahedrons, and on four grids of more irregularly shaped tetrahedrons. Each tetrahedron type is defined by three vectors $\mathbf{e}_1$, $\mathbf{e}_2$, and $\mathbf{e}_3$, which are listed in Table I.

A regular tetrahedron is characterized by edges of equal length. In a Cartesian coordinate system, a unit tetrahedron can be constructed by choosing one point in the point of origin and one on each coordinate axis at the value 1. Two of the test tetrahedra are shaped like needles, with one point far away from the opposing face. They differ in the amount of anisotropy. The remaining two tetrahedra are shaped like wedges, with one edge much smaller than the other ones. The two wedges differ in the amount of anisotropy, too.

On all grids different numbers of pre-smoothing (ν_1) and post-smoothing (ν_2) steps have been tested. Tables II and III show the results for $\nu = (\nu_1, \nu_2) = (1, 0)$ and $\nu = (2, 2)$. Other values of ν have been tested with analog results. The Jacobi smoother is applied with a relaxation parameter of $\omega = 0.8$. The line smoother is always oriented parallel to the shortest edge of a tetrahedron, because this alignment yields the best convergence rates.

The Jacobi smoother has the worst convergence rates in all tests. In all HHG runs the four-color Gauss-Seidel smoother is at least as good as the lexicographic one; often, it is a little bit better. The LFA does not always predict this correctly, e. g. for the unit tetrahedron at $\nu = (1, 0)$. The line smoother is very successful on the wedges. This is due to the fact that two

Table III. Comparison of asymptotic convergence rates between LFA and HHG for $\nu = (2, 2)$

	Regular	Unit	Needle 1	Needle 2	Wedge 1	Wedge 2
Jacobi (LFA)	0.252	0.375	0.944	0.985	0.446	0.781
Jacobi (HHG)	0.250	0.375	0.938	0.979	0.444	0.775
Lex. GS (LFA)	0.143	0.217	0.867	0.964	0.157	0.550
Lex. GS (HHG)	0.141	0.214	0.861	0.956	0.150	0.545
4-Color GS (LFA)	0.125	0.205	0.884	0.966	0.144	0.545
4-Color GS (HHG)	0.125	0.210	0.860	0.956	0.131	0.536
Line-wise (LFA)	0.115	0.159	0.791	0.942	0.056	0.045
Line-wise (HHG)	0.115	0.157	0.783	0.935	0.053	0.043

values of the operator stencils are much smaller than the others [7]. Solving in this direction directly leads to this advantage. The line smoother has a higher computational cost than the other smoothers, of course, but for very anisotropic tetrahedra the gain from an improved convergence rate easily offsets this penalty.

We observe a good matching of HHG experiments and LFA predictions. The differences are always below 0.01, except in some four-color smoother cases. A source for mismatches between the two methods are slight oscillations in the measured convergence rate that remain even when the asymptotic convergence rate has been reached.

5.3. Locally adaptive convergence rate control on semi-structured meshes

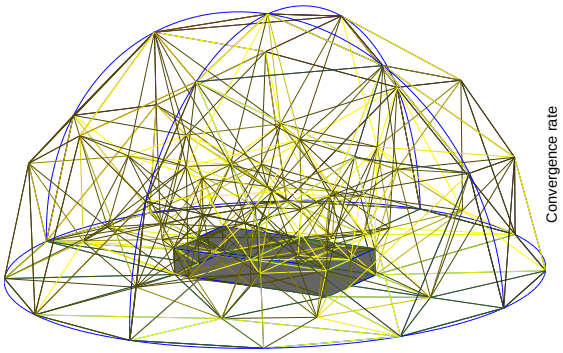
In order to achieve a good overall convergence of the solver, we try to distribute the computational work such that all regions of the domain receive a similar error reduction in each iteration. In our approach we use the convergence rate to control this process. Partial smoothing has already been promoted e. g. in [3].

With HHG's semi-structured meshes it is relatively easy to locally adapt smoothing parameters to mesh-induced convergence rate variations. Since each coarse mesh element gets refined in a regular way (cf. Fig. 2), all its sub-elements are similar, and the operator stencils of all interior grid points are the same. Therefore, one can determine the optimal smoothing parameters for each coarse grid element and use them for all its sub-elements. We introduce two approaches to estimate the convergence rate of a structured region. The first, heuristic approach uses look-up tables, the second one is based directly on LFA prediction.

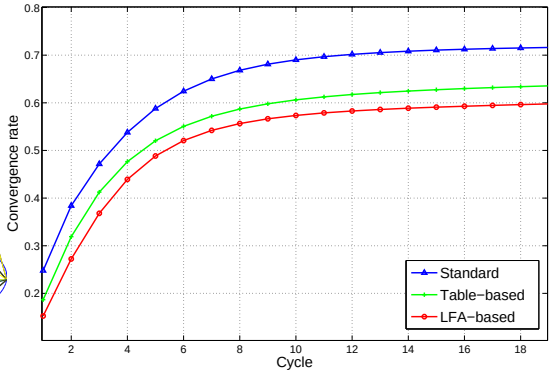
The look-up table approach first classifies tetrahedral elements by their shape. We use the classification into nine tetrahedron types according to edge length ratios. The generated types of degenerations are comparable to [5]. Two examples are the needle and the wedge shown in Section 5.2. In an off-line step the convergence rates of each type for different degrees of anisotropy are measured and stored in a look-up table. As a measure of the anisotropy the aspect ratio of the longest and the shortest edge is taken. At the start of a productive run HHG maps the tetrahedrons of an arbitrary input mesh to an entry of the look-up table. Depending on this information, the global number of smoothing steps is redistributed among the structured regions, in order to get similar convergence rates for each region.

Figure 3. Convergence rate optimization.

(a) Test domain: 293 elements on the coarsest mesh.



(b) Convergence rates for an unoptimized and two optimized runs on the test domain.



While the table approach gets its convergence rate by a look-up table, it is possible to predict these rates with a LFA calculation for each coarse grid element. In both methods the redistribution starts by setting the number of smoothing steps in all grid regions to $\nu = (1, 0)$. Then all convergence rates are predicted and the remaining smoothing steps are successively distributed to the element with the highest convergence rate. After each step the convergence rate of the updated element has to be recalculated. It can be reasonable to set an upper limit for the number of smoothing steps. In our case we took $\nu = (25, 25)$ for the LFA based prediction.

As a domain for testing the optimization we use a half sphere with a small box cut out at its bottom. The diameter of the half sphere is 6.0, the box has a size of $1.0 \times 2.0 \times 0.3$ (see Fig. 3(a)). The domain is discretized with 293 elements on the coarsest mesh. These elements are refined 8 times, which leads to $1.02 \cdot 10^8$ unknowns on the finest level. The lexicographic Gauss-Seidel smoother was used in all tests in this section. A well-known degeneracy measure for FE meshes is the ratio between inscribed and circumscribed radius α, another one is the ratio between the shortest and the longest edge β. The average degeneracy measures of our test mesh are $\alpha = 0.610$ and $\beta = 0.537$. The minimal and maximal values are $\alpha_{min} = 0.313$, $\alpha_{max} = 0.900$, $\beta_{min} = 0.214$, and $\beta_{max} = 0.830$.

For the unoptimized version $\nu = (4, 4)$ yielded the best time to solution. With this number of smoothing steps 20 V-cycles are required to reduce the error of a random initial solution below 10^{-6}. The table-based method needs 15 cycles to reach the same error bound. With the LFA based estimation we can either reduce the number of V-cycles to 12, or perform 20 V-cycles with $\nu = (2, 2)$. Figure Fig. 3(b) plots the convergence rates of all three methods against the number of V-cycles.

Using one core of an 2.8 GHz *Intel Core2 Quad* CPU, a V-cycle takes 16 seconds. The additional overhead of the table based method can be neglected. The setup phase of the LFA based approach takes 6 seconds. Thus, both optimization methods yield a big reduction of the time to solution.

The table based method predicted some convergence rates incorrectly. Thus, the LFA is a better choice, when the effort of its calculation is acceptable. The LFA also allows for further

optimizations like choosing the smoother or the relaxation parameters.

6. Conclusions and outlook

We were able to show a good correspondence between the LFA and HHG for a Jacobi, a lexicographic and a four-color Gauss-Seidel as well as for a line smoother. The variation between the measured results of HHG and the LFA predictions for the asymptotic convergence rate is, except for some cases with the four-color smoother, below 6% for six different structured tetrahedral grids. The discrepancy for the 4-color smoother is below 12%. Line-smoothing shows an especially good convergence for some tetrahedral shapes, e. g. for the wedge type.

The agreement between the LFA and HHG can be seen as a validation for both implementations. Furthermore, benefits of more complex solvers can first be tested with the LFA before implementing them in HHG. This prevents development effort that would be futile, if the solver turned out not to be suitable for HHG. Additionally, the LFA can help to find optimal relaxation parameters. In the case of a lexicographic successive over-relaxation (SOR) smoother this could be done experimentally in parameter studies. However, already for a four-color relaxation scheme this would be much more difficult, because each color can be relaxed with an individual relaxation parameter.

Because of the refinement strategy in HHG, child elements of a tetrahedron have the same anisotropy as the parent element. Thus, it can be worth doing some calculations in order to find out good smoothing parameters, since these parameters are applied to a huge number of tetrahedrons on the finer grids. In this paper we focused on the number of pre-/post-smoothing steps. It turned out that by redistributing the number of smoothing steps over a semi-structured test domain half of the number of smoothing steps can be saved.

An open question is how these improvments behave in a parallel execution, since they are currently only working locally on each compute node. We expect that the effects of the smoothing step redistribution decreases with increasing number of processors. However, the presented methods have a good potential for load balancing since they are based on a priori estimations.

REFERENCES

1. B. Bergen, T. Gradl, F. Hülsemann, and U. Rüde. A massively parallel multigrid method for finite elements. *Computing in Science & Engineering*, 8:56–62, November 2006.
2. B. Bergen, F. Hülsemann, and U. Rüde. Is 1.7×10^{10} unknowns the largest finite element system that can be solved today? In *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 5. IEEE Computer Society, 2005.
3. A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31(138):333–390, April 1977.
4. W. Hackbusch and U. Trottenberg. *Multigrid methods: Fundamental algorithms, model problem analysis and applications.* Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1982.
5. R. Heinzl and T. Grasser. Generalized comprehensive approach for robust three-dimensional mesh generation for tcad. In *Simulation of Semiconductor Processes and Devices, 2005. SISPAD 2005. International Conference on*, pages 211–214, Sept. 2005.
6. U. Meier Yang. Parallel algebraic multigrid methods — high performance preconditioners. In A.M. Bruaset and A. Tveito, editors, *Numerical Solution of Partial Differential Equations on Parallel Computers*, volume 51 of *Lecture Notes in Computational Science and Engineering*, pages 209–236. Springer, 2006.
7. C. Oosterlee U. Trottenberg and A. Schüller. *Multigrid.* Academic Press, London, 2001.