Adaptivity techniques in High Performance FEM Simulation

Dipl.Math. Matthias Grajewski, Prof. Dr. Stefan Turek

1. Motivation

During the last years, tremendous progress has been achieved in the development of methods for a posteriori error control and grid adaptation related to numerical methods for solving Partial Differential Equations (PDEs). It turned out that sophisticated adaptivity techniques are a key ingredient for fast and reliable numerical simulations. Hereby, the emphasis has been put on methods utilising selective refinement of grid cells (h-adaptivity) according to estimated error distributions. However, this leads to highly unstructured grids which decrease the numerical efficiency of an FEM code as these grids require many unaligned and costly memory accesses during the program run. Because of this, only a small fraction of the theoretically available CPU-power can be used in practical computations. Our FEM library FEAST [1,2] potentially overcomes this difficulty by using grids consisting of multiple logical tensor product grids. Logical tensor product grids are grids which are topologically equivalent to a tensor product mesh, i.e. every inner node has exactly four neighbouring mesh nodes (cf. Figure 1). These meshes unfortunately prevent applying standard h-adaptivity techniques. As a remedy, we apply patchwise h-adaptivity techniques and, more important, grid deformation methods (r-adaptivity) which relocate the grid points preserving the logical tensor product structure of the underlying mesh [3, 4]. In this article, we give an overview over the adaptivity and error control techniques applied in FEAST and compare them regarding the quality of the simulation. This is done by considering several prototypical test examples.

2. Numerical concepts and practical examples

The examples given in [3] demonstrate that many of today's numerical simulation tools - based on the standard *sparse* MV techniques - are not able to achieve a significant percentage of the high performance on recent processors which is in the range of more than 1 GFLOP/s; in the case of *fully adaptive FEM codes* we observe in our experiments performance rates of about 10 - 50 MFLOP/s and less for matrix-vector multiplications which are already the fastest components in numerical codes, while complete multigrid solvers often run with even lower speed. One reason for this performance drop are the numerous indirect accesses via global vectors storing the mesh topology and the structure information of the sparse system matrix. As these vectors do not fit into the cache due to their size, effective caching is prevented and therefore the CPU is hampered by waiting for data from the relatively slow RAM. In contrast to this, logical tensor products allow band system matrices. Therefore, for these grids, direct addressing is possible. To gain flexibility, our grids consist of several logical tensor product meshes (cf. Figures 2, 4).

2.1. Adaptive FEM approaches

Most of the recent numerical approaches for adaptive grid refinement (or coarsening) are working in a hierarchical way: One starts with a 'coarse' mesh which is recursively refined. We call it in the following the *macro mesh*, and each triangle/quadrilateral/etc. is a *macro*. Depending on the *error indicator* or *error estimator*, certain global refinement steps are performed, or mesh refinement is applied only locally (we refer to [5] for such grid structures which can lead to 'optimal' locally refined meshes with hanging nodes).

Numerical experience tells that not all numerical simulations benefit from adaptive algorithms to the same extend. Adaptive algorithms usually lead to large improvements if local features have to be resolved in the simulation. This is the case if the solution to be computed features local phenomena like singularities (e.g. stress at re-entrant corners) or jumps (e.g. shock fronts in CFD simulation) which have to be resolved properly. On the other hand, local

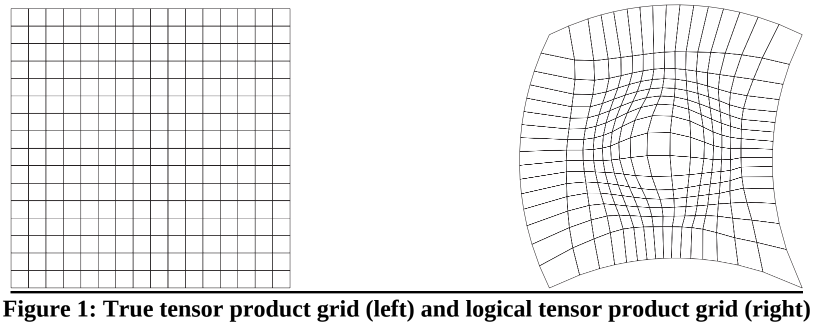

Figure 1: True tensor product grid (left) and logical tensor product grid (right)

quantities like e.g. point values or values for lift or drag can be computed far more accurate by adaptive algorithms. In both cases, it is important to concentrate the mesh in an appropriate way in the region of the local feature. This indicates that in most of the regions of the computational domain tensor product meshes remain feasible even in the situations described here.

First approach: Macro-oriented h-adaptivity

After the computation of our problem on an initial grid, we estimate the error of the obtained approximate solution. If the estimated error η exceeds a user-prescribed error tolerance, adaptive refinement takes place. Hereby, we assume that our error estimator gives insight into the spatial error distribution of the computed solution. Using this information, we compute the error contribution of each macro and sort the macros accordingly. Beginning with the worst one, all macros are refined until the sum of their error contributions exceeds $\alpha \cdot \eta$, where $\alpha \in (0,1)$ stands for a user-defined parameter. Then, the problem is recomputed on the adapted mesh. The adaptation process is repeated until the error falls below the prescribed error tolerance.

Second approach: r-adaptivity

Unlike h-adaptivity, r-adaptivity aims at relocating the grid points according to a prescribed cell size distribution. Hereby, the number of grid points remains constant in contrast to h-adaptivity. The desired cell size distribution is coded in a so-called *monitor function* which can be either defined analytically (a priori adaptivity) or can be created based upon the error distribution estimated by an error estimator (a posteriori adaptivity). The new coordinates of a grid point are calculated by solving a transport ODE which itself depends on a global deformation vector field. This is obtained by solving one global Poisson problem. Therefore, grid deformation requires the solution of a single Poisson problem and $2n$ decoupled explicit ODEs of first order for a grid with n grid points (For details, the reader is referred to [3] and [4]). The quality of the deformed grid in terms of cell size can be assessed by a certain quality measure and can be improved by iteration.

Afterwards, the problem is recomputed on the new deformed mesh and the whole cycle is repeated until the error falls below the given tolerance. However, grid deformation is a special method of grid optimisation with the side condition that the mesh topology and the number of nodes remain unchanged. If for such an optimal mesh the error still exceeds the tolerance, no further improvement is possible using grid deformation and the tolerance can never be reached in this case. To force the algorithm to terminate even in these cases, we stop after a prescribed number of tests or if the error has increased during the last step, as this indicates that grid deformation does not improve the quality of the calculations in the current phase of the simulation.

2.2. Numerical examples

In this section, we present the results of several **numerical tests** in order to compare the three aforementioned approaches with each other as well as with numerical simulations performed without adaptivity. All Test Problems are computed with our FEM package FEAST using bilinear conforming elements.

Test Problem 1: L-domain We consider the Laplace-equation on the domain $[-0.5, 0.5]^2/[0,0.5]^2$ (cf. Figure 2). The Dirichlet boundary conditions are chosen such that in polar coordinates

$$u(r,\varphi) = r^{2/3}\sin(\frac{2}{3}\varphi)$$

is a solution of the Laplace equation. This particular solution features a so-called corner-singularity in the re-entrant corner (0,0) as the gradient of this solution becomes arbitrary large in the region around this corner. Because of this, the solution is hard to approximate, and FEM exhibits a reduced order of convergence in terms of gradient error for this particular problem. The error of the approximating FEM solution is highly concentrated around the re-entrant corner. This can be compensated to some extend by concentrating the grid points around the re-entrant corner. This test example is one of the most widespread benchmark problems for adaptive codes in the literature.

Test Problem 2: donut domain We want to solve the Poisson equation on the domain $1.5 < r < 3$ (cf. Figure 4). The Dirichlet boundary conditions and the right hand side are chosen such that the equation is solved by

$$u(r,\varphi) = (r-1.5)(r-3)\,.$$

In this example we are interested in the error of the functional

$$J(\phi) = \oint_{\Gamma} \frac{\partial\phi}{\partial n} ds\,,$$

where Γ symbolises the inner boundary component and n stands for the outer normal unit vector. This functional is closely related to lift- and drag evaluation in CFD simulations.

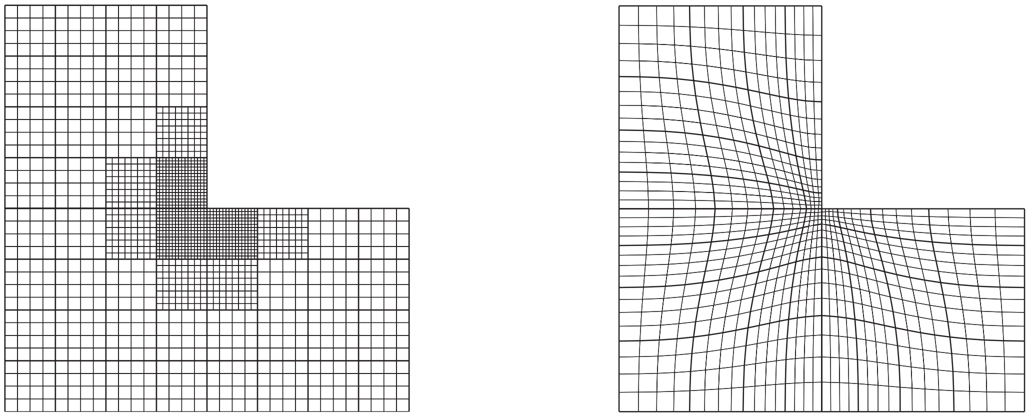

Figure 2: h-adapted (left) and r-adapted (right) computational grid for Test Problem 1, macro borders are indicated by thicker lines

The grid used for Test Problem 1 consists of 48 macros (Figure 2). As it is clear for the first Test Problem that the grid points have to be concentrated around the corner, the grid is predeformed according to the monitor function *f(r)* = max(*r*, 0.01*h*), where *h* denotes the mesh size on the undeformed grid. No further deformation was employed for this Test Problem. Moreover, we compute Test Problem 1 utilising h-adaptivity as described before, where α is set to 0.5. The norm of the gradient error vs. the number of unknowns is displayed in Figure 3. Both adaptivity techniques improve the accuracy of simulation significantly: For an accuracy of 1.0E-3 in the norm of the gradient error, it takes roughly 200,000 unknowns when using grid deformation. Employing h-adaptivity in the way described here, estimated four million unknowns are necessary, while an extrapolation of the data provided in Figure 5 leads to the conclusion that obtaining this accuracy without any adaptivity would require no less than 30 millions of unknowns! This underlines the statement made in the introduction that adaptivity and error control are mandatory for fast and accurate simulation. It can deduced from the data in Figure 3 that by grid deformation even the decrease of convergence order due to the singularity can be cured!

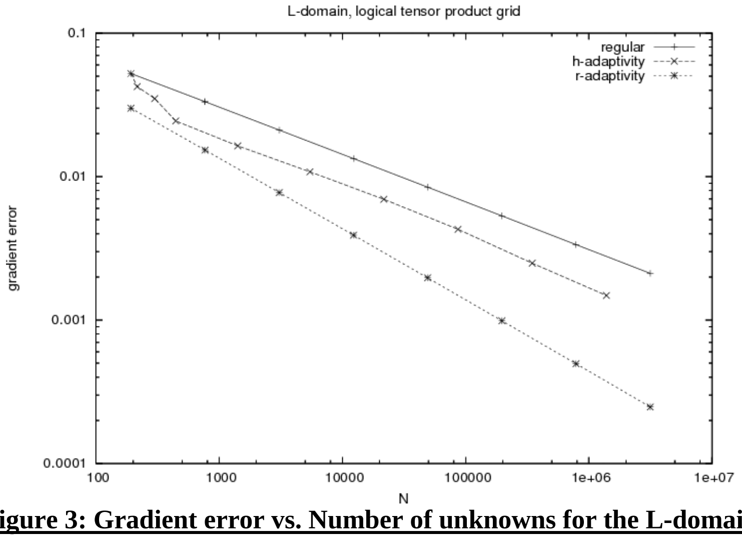

Figure 3: Gradient error vs. Number of unknowns for the L-domain

The grid employed for Test Problem 2 consists of 64 macros (cf. Figure 4). The error of the functional J is estimated using the DWR method of Becker and Rannacher (see [5]). The corresponding dual problem was computed on the same grid as the actual one employing conforming bilinear elements. According to the estimated error distribution, the monitor function is created. To do so, we take the negative logarithm of the error distribution, which is below one, and smooth this function several times using Laplacian smoothing. The smoothing is necessary to increase the quality of the deformed grid. It should be noted that apart from the one proposed here, there are many possibilities to filter and rescale the error distribution in order to get reasonable monitor functions. This, however, is topic of current research and will be addressed more in detail in [3]. The number of deformation cycles is bounded to eight.

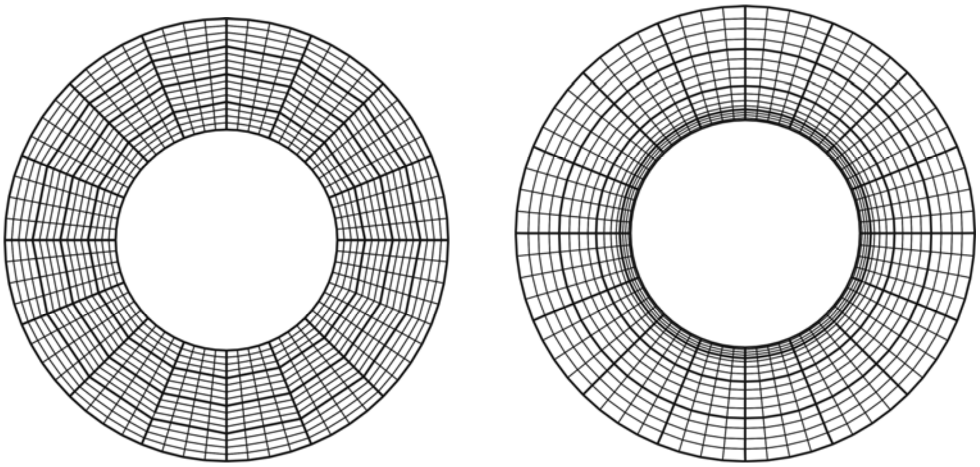

Figure 4: Unadapted (left) and r-adapted grid for Test Problem 2. The macro borders are indicated by thicker lines

The results of the computations (Figure 5) indicate that like in the first Test Problem mesh adaptation yields tremendous improvements in quality. In order to reach the accuracy of the deformed mesh with roughly 1000 unknowns, it needs a unadapted mesh with one million unknowns, i.e. applying appropriate error-driven deformation techniques, one can save 99,9% of the unknowns in the computation!

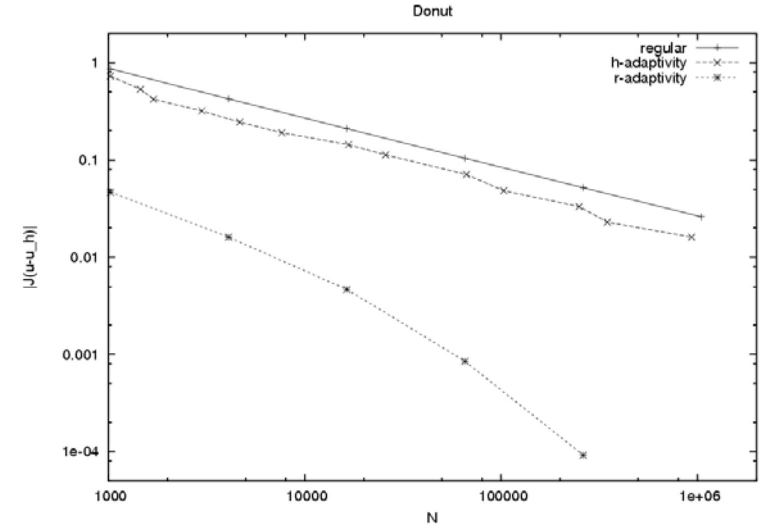

Figure 5: Resulting errors for the computations on the donut domain

References

[1] Chr. Becker. FEAST - The realization of Finite Element software for high-performance applications. PhD Thesis, University of Dortmund, 2006 (to be published)

[2] Chr. Becker, S. Kilian, S. Turek. Consequences of modern hardware design for numerical simulations and their realization in FEAST. Proceedings "Euro-Par'99 Parallel Processing" (P. Amesto, P. Berger, M. Dayde, I. Duff, V. Fraysse, L. Giraud, D. Ruiz eds.), Toulouse, France, August/September 1999, 643 - 650

[3] M. Grajewski. Goal-oriented Adaptivity for FEM Simulation Techniques in CFD and CSM. PhD Thesis, University of Dortmund, 2006 (to be published)

[4] M. Grajewski, M. Köster, S. Kilian, S. Turek. Numerical Analysis and Practical Aspects of a Robust and Efficient Grid Deformation Method in the Finite Element Context, Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 294, FB Mathematik, Universität Dortmund, 2005

[5] R. Rannacher. Error control in finite element computations. In H. Bulgak and C. Zenger, eds., Proc. Summer School Error Control and Adaptivity in Scientific Computing, Kluwer, Academic Publishers, 247 – 278, 1998