Numerical analysis and implementational aspects of a new multilevel grid deformation method

Matthias Grajewski and Michael Köster and Stefan Turek

Institute of Applied Mathematics, Dortmund University of Technology, Vogelpothsweg 87, D-44227 Dortmund, Germany,
matthias.grajewski@mathematik.tu-dortmund.de

Abstract

Recently, we introduced and mathematically analysed a new method for grid deformation [15] we call basic deformation method (BDM) here. It generalises the method proposed by Liao [4,6,19]. In this article, we employ the BDM as core of a new multilevel deformation method (MDM) which leads to vast improvements regarding robustness, accuracy and speed. We achieve this by splitting up the deformation process in a sequence of easier subproblems and by exploiting grid hierarchy. Being of optimal asymptotic complexity, we experience speed-ups up to a factor of 15 in our test cases compared to the BDM. This gives our MDM the potential for tackling large grids and time-dependent problems, where possibly the grid must be dynamically deformed once per time step according to the user's needs. Moreover, we elaborate on implementational aspects, in particular efficient grid searching, which is a key ingredient of the BDM.

Key words: grid generation, deformation method, grid adaptation
AMS classification: 65N15, 65N30, 76D05, 76D55

1 Introduction

Grid deformation, i.e. the redistribution of grid points while preserving its topology, has become an important alternative to elementwise refinement for grid adaptivity. Today, many methods for grid deformation or (in the case of time-depending problems) moving the grid are available and have been sucessfully applied to the computation of sophisticated problem like the Navier-Stokes equations [12], the Euler equations [10] and multi-phase flow [11], only to name a few. However, many popular methods for grid deformation necessitate expensive solution of complicated nonlinear PDEs [5,8,17] or even the solution of optimisation problems [11]. Moreover, for many methods it is not straightforward to predict which effect in detail the *monitor function* prescribed has, which is to control the deformation process [7]. Liao's method [4,6] is a prominent member of the group of *dynamic approaches* to grid deformation. These kind of methods involve time stepping or pseudo-time stepping. Liao's method aims at deforming the given equidistributed grid such that the spatial distribution of the element area corresponds to the monitor function. In contrast to the aforementioned approaches, its effect is thus clear. Recently, we developed a generalisation of this method [14,15] which we call basic deformation method (BDM) here for a large class of possibly non-equidistributed initial grids and provided a rigorous convergence analysis. Both methods require the solution of a single Poisson problem and of a decoupled system of initial value problems (IVPs) only. Furthermore, grid tangling cannot occur [20] in both Liao's and our method under certain mild conditions on the boundary vertices.

Adapting a given computational grid by grid deformation (*r-adaptivity*) offers some advantages over the widespread method of subdividing selected elements (*h-adaptivity*): In many applications, local and anisotropic phenomena occur. By anisotropic refinement and alignment according to such phenomena, the accuracy of the calculation can be vastly improved [1,13]. If the given grid however is not well-aligned, the h-adapted grid remains not well-aligned. In contrast, grid deformation in principle permits to adjust the orientation of the elements as well and thus offers additional flexibility.

It is well known that in simulations using the finite element method (FEM), grid adaptivity controlled by a posteriori error estimation is crucial for reliable and efficient computations. The widely used realisation of h-adaptivity by allowing hanging nodes on element level, however, has severe impact on the speed of computation. Recent research [18,23] shows that in typical adaptive FEM codes using this kind of h-adaptivity, only a small fraction of the available processor performance can be typically exploited. This is partially due to the extensive usage of indirect addressing in such codes which is necessary to handle the unstructured grids emerging from the adaptation procedure. On the other hand, by using local generalised tensor product grids and thereby avoiding indirect addressing, we were able to achieve a very significant speedup. This has been successfully implemented in the new FEM package FEAST [2]. In this context, grid deformation is an ideal tool to grant the geometric flexibility necessary for grid adaptation according to a posteriori error estimators while maintaining logical tensor product structures of the grid.

In the next section, we describe briefly the BDM and its convergence analysis. In section 3, we focus on implementational aspects, in particular on efficient grid search. Section 4 deals with constructing a more robust deformation method by splitting the deformation problem into several easier subproblems when necessary. In section 5, we introduce our multilevel deformation method which turns out to be superior with respect to accuracy and speed.

2 Description and numerical realisation of the BDM

A computational domain $\Omega \subset \mathbb{R}^2$ is triangulated by a conforming grid $\mathcal{T}$ consisting of NVT vertices and *NEL* quadrilateral elements T with maximum edge length h_T and area $m(T)$. We denote the set of edges in $\mathcal{T}$ by $\mathcal{E}$ and the set of grid points by $\mathcal{V}$. We symbolise the standard Lebesgue norm by $||\cdot||_0$. For a domain $D \subset \mathbb{R}^2$, the function space of k-fold continuously differentiable functions on D is referenced by $\mathcal{C}^k(D)$. For an interval I, $\mathcal{C}^{k,\alpha}(I)$, $0 < \alpha < 1$, denotes the space of functions with Hölder-continuous k-th derivatives. A domain has a $\mathcal{C}^{k,\alpha}$-smooth boundary $\partial\Omega$, iff $\partial\Omega$ can be parameterised by a function in $\mathcal{C}^{k,\alpha}(I)$. The Jacobian matrix of a smooth mapping $\Phi : \Omega \to \Omega$ is denoted by $J\Phi$, its determinant by $|J\Phi|$.

The theoretical background of our approach – like Liao's [4,6] – is based on Moser's work [9]. All numerical grid deformation algorithms described below aim at constructing a bijective mapping $\Phi : \Omega \to \Omega$ satisfying

$$g_0(x)|J\Phi(x)| = f(\Phi(x)), \quad x \in \Omega \quad \text{and} \quad \Phi : \partial\Omega \to \partial\Omega. \tag{1}$$

The monitor function f represents the desired spatial distribution of the element area on the grid to create from the given one, whose spatial distribution of the element area is represented by the *area function* g_0. The new coordinates X of a grid point x are computed by $X := \Phi(x)$. Similar to [4,6], the BDM obtains Φ in four steps. For more details regarding the derivation of this algorithm, we refer to earlier work [15].

Theorem 1 ([15]) *Let the boundary of Ω be $\mathcal{C}^{3,\alpha}$-smooth and let $f, g_0 \in \mathcal{C}^1(\Omega)$ be strictly positive in $\bar{\Omega}$. Then, if the mapping $\Phi : \Omega \to \Omega$ constructed in the BDM exists, it fulfills (1).*

Liao's methods appears as special case of the BDM for $g_0 \equiv 1$. Although Liao's method can applied in our more general case, the area distribution of the deformed grid matches f only for uniform initial grids.

Analogous to Liao's method, grid tangling cannot occur due to $|J\Phi(x)| > 0$, if $\Phi(x) = x \; \forall x \in \partial\Omega$ holds. This could be achieved by replacing ∇v by $\widehat{\nabla v} := \nabla v + \operatorname{curl} w$ in (4), where w is a vector field chosen such that $\widehat{\nabla v}|_{\partial\Omega} = 0$ holds. In this paper, we neglect $\operatorname{curl} w$ and thus allow the boundary points to move along the boundary.

Our numerical realisation of the BDM is as follows. The user-defined *monitor*

Algorithm 1: Basic grid deformation method (BDM)

Input: f, $g_0 : \Omega \to \mathbb{R}^+$
Output: Φ

1) Scale f or g_0 such that

$$\int_\Omega \frac{1}{f(x)} dx = \int_\Omega \frac{1}{g_0(x)} dx. \tag{2}$$

For convenience, we assume that (2) is fulfilled from now on. Let $\tilde{f}$ and $\tilde{g}_0$ denote the reciprocals of the scaled functions f and g_0.
2) Compute $v : \Omega \to \mathbb{R}$ by solving

$$(\nabla v, \nabla \varphi) = (\tilde{f} - \tilde{g}_0, \varphi) \quad \forall \varphi \in H^1 \tag{3}$$

subject to $\partial_n v = 0$.
3) For any $x \in \Omega$, solve the IVP

$$\frac{\partial \varphi(x,t)}{\partial t} = \eta(\varphi(x,t),t), \quad 0 \le t \le 1, \quad \varphi(x,0) = x \tag{4}$$

with t representing an artificial time variable and

$$\eta(y,s) := \frac{\nabla v(y)}{s\tilde{f}(y) + (1-s)\tilde{g}_0(y)}, \quad y \in \Omega, s \in [0,1].$$

4) Define $\Phi(x) := \varphi(x,1)$.

function f describes the desired distribution of the element area on the deformed grid up to a spatially fixed constant. In practical computations, we use the bilinear interpolant of the analytically given monitor function f instead of f itself. This does not significantly affect the BDM [14]. In what follows, we denote both the analytic monitor function and its interpolant by f. As g_0 reflects the distribution of element area on the given initial grid $\mathcal{T}$, it has to be constructed from $\mathcal{T}$. In a grid point x, we set $g_0(x)$ as the arithmetic mean of the area of the elements surrounding x and define g_0 as the bilinear interpolant of these node values. Both f and g_0 must be strictly positive on $\bar{\Omega}$. In the same manner, we compute the area function $g_1(x)$ on the obtained deformed grid. Mainly due to numerical errors, the desired aea distribution f and the obtained area distribution g_1 may not fully coincide in practical computations.

We compute an approximate solution v_h of (3) by applying conforming bilinear finite elements. The solve of the corresponding algebraic systems requires special care, as the solution of the Neumann problem (3) is unique up to an additive constant only. We thus use a modified multigrid method in which after every iteration the side condition $\int_\Omega v_h \, dx = 0$ is imposed by adding a

0			
1/2	1/2		
1	−1	2	
	1/6	2/3	1/6

Table 1
Butcher scheme of the third order Kutta method (RK3)

suitable constant to the iteration vector.

We replace ∇v in (4) by a recovered gradient $G_h(v_h)$ computed by standard averaging techniques [26,27]. The fully decoupled IVPs (4) are then solved with standard methods for all grid points $x \in \mathcal{V}$. However, numerical tests [15] reveal that *any* IVP solver can be of second order at most in this situation, as the right hand side, being continuous only, does not meet the smoothness requirements of high order methods. In all numerical tests in this article, we employ the classical third order Kutta method (RK3) which performed preferably well in our tests. Table 1 depicts its Butcher scheme.

In what follows, we identify the deformation algorithm 1 which is formulated on the continuous level with its numerical realisation described above and call both BDM. Moreover, in the discrete case, we do not distinguish between the approximation of Φ and the image of $\mathcal{T}$ under the approximate Φ.

Remark 1 *The BDM can be applied in arbitrary dimension without any modifications. In this article, we restrict ourselves to the two-dimensional case for the sake of implementational simplicity. For the three-dimensional case, we refer to Miemczyk's [21] and Panduranga's [22] work.*

For the BDM, a rigorous mathematical analysis is available [14,15]. This includes a proof of convergence with respect to the *quality measures*

$$Q_0 := \left\| \frac{f(x)}{g_1(x)} - 1 \right\|_0 \quad \text{and} \quad Q_\infty := \max_{x\in\Omega} \left| \frac{f(x)}{g_1(x)} - 1 \right|.$$

If we do not distinguish between Q_0 and Q_∞, we omit the subscript. These measures describe the deviation of the obtained area distribution g_1 on the deformed grid from the prescribed area distribution f. The smaller Q is the more accurate is the grid deformation.

When processing the BDM with numerical methods, there are three error sources.

(1) The deformation Poisson problem (3) is solved approximately.
(2) The IVPs (4) are solved approximately.
(3) The interpolation of the discontinuous element area distribution induces

a *consistency error.*

The consistency error arises as the actual element area distribution is discontinuous and has to be interpolated in order to obtain g_0 and g_1. Therefore, even without any numerical error, we cannot expect one of the quality measures to be exactly zero.

We need the following preparations in order to formulate the main convergence result. For proofs and details, we refer to [15]. For an arbitrary grid point x, we denote its deformed counterpart by X_h, iff we solve the deformation PDE (3) numerically and the IVP (4) exactly, and by $\tilde{X}$, iff both differential equations are solved with numerical methods.

Definition 1 *a) We call a sequence of triangulations $(\mathcal{T}_i)_{i\in\mathbb{N}}$* edge-length regular*, iff*

$$h_i := \max_{e\in\mathcal{E}_i} |e| = \mathcal{O}(NEL_i^{-1/2}) \quad \forall i \in \mathbb{N}.$$

b) The sequence of triangulations $(\mathcal{T}_i)_{i\in\mathbb{N}}$ is said to be area regular*, iff*

$$\exists c, C > 0 : ch_i^2 \le m(T) \le Ch_i^2 \quad \forall T \in \mathcal{T}_i \, \forall i \in \mathbb{N}. \tag{5}$$

c) An edge-length regular sequence of triangulations $(\mathcal{T}_i)_{i\in\mathbb{N}}$ fulfils the similarity condition*, iff there is a function g with $0 < g_{\min} \le g(x) \le g_{\max} < \infty$ and $c_s, C_s > 0$ with*

$$\frac{1}{h_i^2} c_i m(T) = g(x) + \mathcal{O}(h_i) \quad \forall\, x \in T \quad \forall\, T \in \mathcal{T}_i \, \forall i \in \mathbb{N}, \quad c_s \le c_i \le C_s. \tag{6}$$

Here, c_i is a spatially fixed constant.
d) A grid deformation method converges *with respect to a quality measure Q, iff $Q \to 0$ for $h_i \to 0$.*

The similarity condition ensures that up to a constant, all grids in the sequence feature a similar distribution of the element size and thus essentially differ in their number of elements only. Any sequence of grids emerging from regular refinement of an initial coarse grid fulfills the similarity condition. The following theorem states the convergence of the BDM in the sense that the difference between the desired and the actual distribution of element size measured by Q decreases with finer initial grids. Moreover, edge-length regularity is preserved by the BDM. This is in contrast to the classical shape-regularity, which is not necessarily preserved, as Liao's method and the BDM do not feature control over the angles in the elements.

Theorem 2 *Let $(\mathcal{T}_i)_{i\in\mathbb{N}}$ be a sequence of grids with grid size $h_i \to 0$ which fulfils condition (6) and let us denote the sequence of numerically deformed grids by $(\tilde{\mathcal{T}}_i)_{i\in\mathbb{N}}$. We require $0 < \varepsilon < f \in \mathcal{C}^1(\bar{\Omega})$ and assume $||v - v_h||_\infty = \mathcal{O}(h^{1+\delta})$, $\delta > 0$. Let $||X_h - \tilde{X}|| = \mathcal{O}(h^{1+\delta})$ be true for any vertex. Then,*

a) the sequence of numerically deformed grids $(\tilde{\mathcal{T}}_i)_{i\in\mathbb{N}}$ is edge-length regular,
b) $(\tilde{\mathcal{T}}_i)_{i\in\mathbb{N}}$ is area regular,
c) the numerical realisation of the BDM converges with respect to Q_0 and Q_∞, and $\exists c > 0$ independent of h with

$$Q_0 \leq ch^{\min\{1,\delta\}}, \quad Q_\infty \leq ch^{\min\{1,\delta\}}.$$

Corollary 1 *Assume that the numerical method for computing the IVPs (4) is of second order at least. Let $(\mathcal{T}_i)_{i\in\mathbb{N}}$ be area- and shape-regular and fulfil (6). For f and $(\mathcal{T}_i)_{i\in\mathbb{N}}$, the assumptions of theorem 2 may hold. Furthermore, we choose the step size Δt of the deformation IVPs as $\Delta t = \mathcal{O}(h)$. Then, if $||v - v_h||_\infty = \mathcal{O}(h^2)$, $Q_0 \leq ch$ and $Q_\infty \leq ch$.*

3 Search methods

From a theoretical point of view, the solution of the deformation IVPs (4) is standard. However, all numerical methods for IVPs require at least one evaluation of the right hand side per time step which implies the evaluation of FEM functions ($G_h(v_h)$, $\tilde{f}$ and $\tilde{g}$) in real coordinates. This requires searching the grid since one has to find the encompassing element for each evaluation point. Unfortunately, these points (except for the very first time step) may have moved to an entirely different region of the grid during the time steps already performed. Note that the evaluations of $G_h(v_h)$, $\tilde{f}$ and $\tilde{g}$ have to be carried out on the original grid, where these functions have been computed. Hence, the BDM needs to search the whole grid (at least once) *per time step* and *per grid point*. Thus efficient searching strategies are indespensable to solve the IVPs efficiently. For simple domains (e.g. the unit square) in combination with uniform grids, the encompassing element can easily be found by e.g. comparing real coordinates. However, such a simple approach fails for more general initial grids we aim at. We propose and test three different strategies.

Brute Force We loop for each point over all elements in the grid until the element containing the point is found. The search time is $\mathcal{O}(NVT \cdot NEL) = \mathcal{O}(NEL^2)$, as the whole grid has to be searched in the worst case. Note that NEL and NVT grow with the same order.

Raytracing Being adopted from computer graphics, this method avoids searching the entire grid. We test if the point we search is inside the element T_{old} where it was in the previous time step. If not, we connect the center of T_{old} with the point to find. Detecting the element edge e which intersects with this ray, we move to the element T_{new} given by $T_{\text{old}} \cap T_{\text{new}} = e$. Then, setting $T_{\text{old}} := T_{\text{new}}$, we proceed as before (figure 1, left).

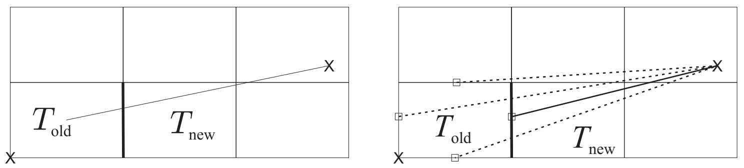


Fig. 1. The principles of raytracing and distance search

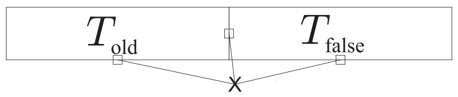


Fig. 2. In this situation, distance search fails

The ray must not intersect the element in one of its vertices as e is not unique then. If so, we modify the starting point of the ray heuristically by disturbing the coordinates of the element center. Searching in a non-convex domain Ω requires special care as well. Even though the grid point before and after the current time step is located in Ω, the ray between these points may leave Ω. Then, we find the second intersection of this ray with $\partial\Omega$ by looping over all elements containing at least one boundary edge. We continue the search in the element containing this second intersection.

Distance search We test if the point is still inside the element T_{old} where it was in the previous time step. If not, we compute the squares of the distances of the edge midpoints of T_{old} to the point we search for. The new element T_{new} is the one which is adjacent to the edge with the shortest distance (figure 1, right). However, in certain cases the distance information is misleading. Then, the search may trap into an infinite loop (figure 2): Here, the comparison of the distances leads to T_{false} as next candidate. But on T_{false}, the comparison of the distances leads back of T_{old} as successor. Therefore, if during the search the previous element is chosen as T_{new}, we declare the distance search failed and start a raytracing search on T_{old} as fallback. As one step in distance search step needs less arithmetic operations than one step of raytracing search, we expect the distance search to perform faster.

The maximum length of the search path, i.e. the number of element changes during the search for a single grid point, and therefore the maximum amount for both one raytracing and distance search is $\mathcal{O}(NEL)$. Thus, the total search time behaves like $\mathcal{O}(NEL^2)$ in the worst case like for brute force search. However, grids used in FEM calculations are commonly created by regular refinement of a given coarser grid. On these grids, the numbers of nodes in x- and y-direction grow with the same order leading to a maximal length of the search path of $\mathcal{O}(\sqrt{NEL})$. Then, the total search time grows like $\mathcal{O}(NEL^{3/2})$ in contrast to the brute force approach, where the search time still shows quadratic growth.

Test Problem 1 *We triangulate the unit square* $\Omega = [0,1]^2$ *with an equidis-*

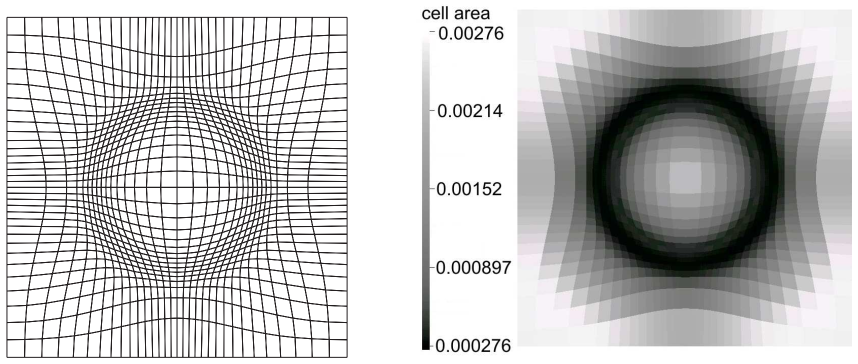


Fig. 3. Resulting grid (left) and area distribution (right) for test problem 1, $\varepsilon = 0.1$, 1,024 elements

tant tensor product grid. The monitor function f is defined by

$$f(x) = \min\left\{1, \max\left\{\frac{|d-0.25|}{0.25}, \varepsilon\right\}\right\}, \quad d := \sqrt{(x_1-0.5)^2 + (x_2-0.5)^2}. \tag{7}$$

Our choice $\varepsilon = 0.1$ implies that on the deformed grid the largest element has 10 times the area of the smallest one. In figure 3, we show a resulting grid consisting of 1,024 elements.

For this test problem, we evaluate the time for searching the grid during the solution of the IVPs (4). The code is compiled using the Intel Fortran Compiler v9.1 with full optimisation and runs on an AMD Opteron 250 server equipped with a 64-bit Linux operating system. In the IVP solve, we perform 10 equidistant time steps. The RK3 method requires three evaluations per component and IVP time step and thus yields 60 evaluations per grid point. We compare the search time needed for all evaluations in the BDM for the three search algorithms on various levels of regular refinement. Due to the natural inaccuracy in measuring times, all tests have been repeated until the relative error of the mean value of the measured times fell below 1%.

The search times are displayed in figure 4 (left). Tests skipped due to overly long computational times are omitted. Both distance and raytracing search are by far faster than the brute force approach, which by its quadratic search time (indicated by the slope of 2) leads to inappropriate search times (≈ 3 h in our example). In contrast, the search times of distance and raytracing search grow even smaller than $\mathcal{O}(NEL^{3/2})$. However, the slope obviously increases to 1.5 with grid refinement. On very fine grids, where the search paths are longer than on coarse ones, distance search is superior: For $NEL \approx 10^6$, distance search is roughly 30% faster than the raytracing approach.

Potentially, hierarchical search methods based on spatial partitions feature even higher speed due to their total search time of $\mathcal{O}(NEL \log NEL)$. However, these methods require sufficiently long search paths to perform superior

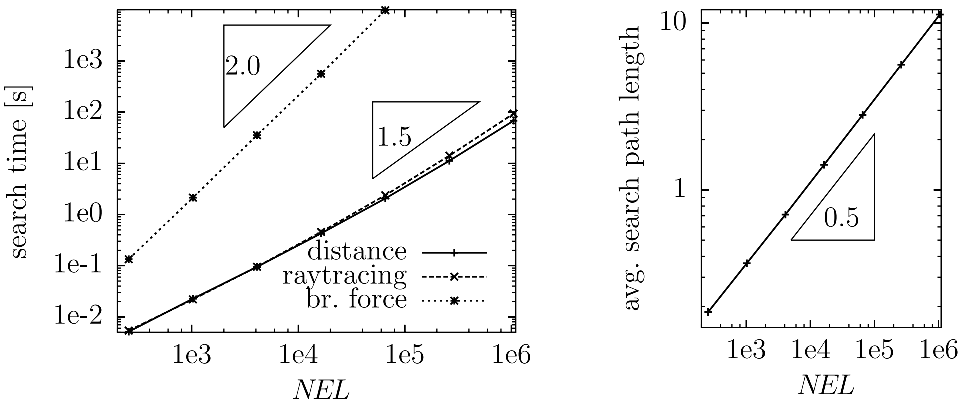


Fig. 4. Search times (left) and average search path lengths (right) vs. number of elements *NEL* for test problem 1, 10 time steps

because of their inherent overhead of constructing the search hierarchies. For test problem 1, we present in figure 4 (right) the average length of the search paths during deformation. We ignore the searches of the intermediate points occurring inside RK3 as well as searches of boundary points. The average search path length doubles in every refinement step. That arises from the fact that a grid point on a fine grid is moved (almost) to the same cartesian coordinates as on a coarse grid, but the element size is halved in each refinement. Even on very fine grids, the search path length is rather small. Being smaller than 10 for test problem 1, *the search paths are too short to expect significant speed-ups using hierarchical searching even on very fine grids.*

4 Enhancing robustness

The grid resulting from test problem 1 (figure 3) demonstrates that this test problem can be easily solved with the BDM. However, in the case of monitor functions with very steep gradients or monitor functions implying extreme variations in element size, it may happen that the corresponding solution of the Poisson problem (3) cannot be resolved properly on the initial given grid. In this case, non-convex elements can appear. This is not due to theoretical shortcomings of the method, but due to the numerical error induced by the approximate solution of (3). In this context, the error produced by the numerical solution of the IVP (4) seems to be far less critical. Computing test problem 1 on tensor product grids of various levels of refinement, we now investigate the robustness of the BDM with respect to desired sharp concentrations of elements in one part of Ω. We solve the IVPs using a constant time step size of 0.02. Decreasing the shape parameter ε in the monitor function (7) leads to an increasing concentration of the grid around the circle. In table 2, we show the lowest shape parameter $\varepsilon_{\min}$ for which the deformation process leads to valid grids with respect to *NEL* and to the method for gradient recovery.

NEL	INT	SPR	PPR
4,096	$1.9 \cdot 10^{-2}$	$2.0 \cdot 10^{-2}$	$1.9 \cdot 10^{-2}$
16,384	$2.0 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$	$2.0 \cdot 10^{-2}$
65,536	$1.5 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$
262,144	$1.2 \cdot 10^{-2}$	$1.2 \cdot 10^{-2}$	$1.2 \cdot 10^{-2}$
1,048,576	$9.0 \cdot 10^{-3}$	$9.0 \cdot 10^{-3}$	$9.0 \cdot 10^{-3}$

Table 2
Minimal shape parameters $\varepsilon_{\min}$ for which test problem 1 can be computed

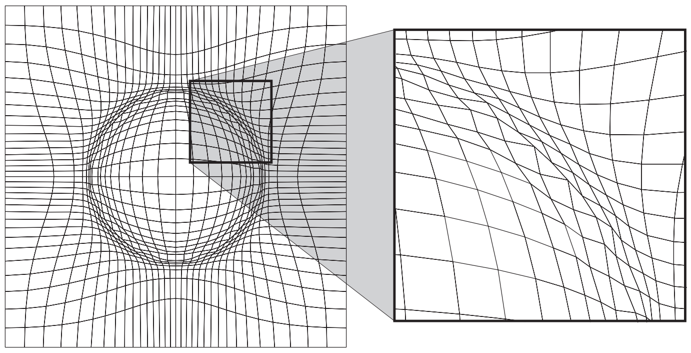


Fig. 5. Resulting grid for test problem 1, $\varepsilon = 0.018$, 1,024 elements.

We compare standard averaging (INT), the superconvergent patch recovery technique by Zienkiewicz and Zhu (SPR) [27] and the polynomial preserving patch recovery technique (PPR) by Zhang [26]. All these techniques lead to recovered gradients of second order accuracy on regular grids. To determine $\varepsilon_{\min}$, we compute test problem 1 repeatedly and decrease ε by 0.001 in every run. Regardless of the level of refinement, the BDM fails if ε falls below ≈ 0.01. For coarser grids with 1024 elements or even less, the monitor function cannot be properly resolved on the initial grid which disturbs the measurements. Therefore we ignore this particular case. The results (table 2) do not show any strong dependency of $\varepsilon_{\min}$ on the grid level although on increasing high levels, $\varepsilon_{\min}$ seems to decrease. It is almost independent of the recovery method for this test problem. For $\varepsilon = 1.8 \cdot 10^{-2}$ and *NEL* = 1,024, the elements around the circle are nearly non-convex, the maximal angle measured in this grid is 179.89° (figure 5).

Repeating this numerical test with a smaller constant time step size of 0.002 yields exactly the same results. Because of this, the main numerical error in the BDM must be introduced by the approximate solution of (3).

Enhancing the robustness by solving (3) more accurately, e.g. by using a highly refined grid would require demanding calculations. Instead, we split the deformation up into several less demanding *adaptation steps*. To do so, we consider a *blended monitor function* f_s, a linear combination of f and g_0:

$$f_s(x) := sf(x) + (1-s)g_0(x), \quad s \in [0,1] \tag{8}$$

The key idea in our robust deformation method (RDM, algorithm 2) is to iterate the BDM employing f_s instead of f itself with increasing *blending parameter* s such that the monitor function in the BDM steps resembles f more and more with every new adaption step. The composition of all BDM steps is to yield the desired mapping Φ. The number of BDM steps and the blending parameters are chosen such any BDM step can be computed by our numerical realisation of the BDM (see section 2). For compatibility reasons, we require $\int_\Omega f\,dx = \int_\Omega g_0\,dx$ here.

Algorithm 2: Robust Deformation Method (RDM)

```
Input: $f$, $g_0 : \Omega \to \mathbb{R}^+$, $1 < \gamma_0$
Output: $\Phi$

$\Phi_0 := Id$;
$n_a :=$ CompNa($f$, $g_0$, $\gamma_0$);
$S :=$ CompBlendpars($f$, $g_0$, $\gamma_0$);
for $i = 1$ to $n_a$ do
    $\Phi_i :=$ BDM($f_{S(i)}, g_{i-1}$) $\circ\ \Phi_{i-1}$ ;
    $g_i := f_{S(i)}$;
end
$\Phi := \Phi_{n_a}$
```

We now investigate how to choose the *blending parameter* s in (8) and how many adaptation steps n_a to perform. In the RDM, this corresponds to defining the functions `CompNa` and `CompBlendPars` which determines for all adaptation steps the blending parameter $S(i), 1 \leq i \leq n_a$. For the monitor function f defined in (7), ε, which describes the ratio of the area of the smallest and largest elements on the deformed grid, is a certain measure for the “numerical difficulty” of the deformation problem: For $\varepsilon = 1$, the deformation is trivial as $\Phi(x) = x\,\forall x \in \Omega$, the choice $\varepsilon = 0.1$ served as first test example; for $\varepsilon = 0.01$, the BDM fails as demonstrated above. Deforming a grid according to f with $\varepsilon = 0$ is impossible at all because of requiring elements with vanishing area. We set $f_{\min} := \min_{x\in\Omega} f(x)$ and $f_{\max} := \max_{x\in\Omega} f(x)$. With $\varepsilon \approx f_{\min}/f_{\max}$ for the monitor function (7), we interpret the ratio $f_{\max}/f_{\min}$ as indicator for the “numerical difficulty” of the deformation problem. These considerations apply to the special case of an equidistributed initial grid. On a suitably pre-deformed grid, we expect the BDM to work properly for test problem 1 with small ε as well, as the actual deformation is almost trivial due to the pre-adjustment. Then, regardless of the properties of both f and g_0, these functions almost coincide and thus we have $f(x)/g_0(x) \approx 1$. This and rearranging (1) to $|J\Phi(x)| = f(\Phi(x))/g_0(x)$ suggests

$$\gamma' := \frac{\max_{x\in\Omega} \frac{f(\Phi(x))}{g_0(x)}}{\min_{x\in\Omega} \frac{f(\Phi(x))}{g_0(x)}} \geq 1$$

as heuristical difficulty measure for non-equidistributed grids. Unfortunately, γ' is unsuitable for practical computations as it relies on the unknown transformation Φ. Therefore, we neglect the influence of Φ and define

$$\gamma := \gamma(f, g_0) := \frac{(f(x)/g_0(x))_{\max}}{(f(x)/g_0(x))_{\min}} \geq 1$$

as such indicator. For the trivial deformation problem $f = g_0$, it holds $\gamma = \gamma' = 1$. In the special case of test problem 1, we have $g_0 = 1$ after scaling. Hence, it then holds $\gamma = f_{\max}/f_{\min} \approx \varepsilon^{-1}$, and the initial "difficulty measure" is recovered.

We use the blending in formula (8) in order to *reduce γ such that every adaptation step according to f_s in the RDM can be treated by the BDM.* Let us assume that the BDM leads to valid grids for all settings with $1 < \gamma < \gamma_0$, where $1 < \gamma_0$ is a user-defined parameter, and take for granted that for the given monitor function $\bar{f}$ and the given current area distribution $\bar{g}$, it holds $\gamma > \gamma_0$. In this situation, n_a is set to

$$n_a = \left\lceil \frac{\ln(\gamma)}{\ln(\gamma_0)} \right\rceil, \tag{9}$$

$\lceil x \rceil := \min_{k\in\mathbb{Z}}\{k \geq x\}$ denoting the ceiling function. The blending parameter $S(i)$ (compare formula (8)) in the i-th grid adaptation step is chosen such that

$$\gamma_i := \gamma(f_{S(i)}, g_0) = \left(\sqrt[n_a]{\gamma}\right)^i \leq (\gamma_0)^i \tag{10}$$

holds. Here, γ_i is computed prior to any deformation. Starting on the undeformed grid, we perform a sequence of n_a BDM steps where each of these steps satisfies

$$g_{i-1}(x)|\Phi_i(x)| = f_{S(i)}(\Phi_i(x)), \quad i = 1, \ldots n_a. \tag{11}$$

We conclude that for the composition $\Phi := \Phi_{n_a} \circ \ldots \circ \Phi_1$ which implicitly has been applied, it holds

$$\begin{aligned}
|J\Phi(x)| &= |J(\Phi_{n_a} \circ \ldots \circ \Phi_1(x))| \\
&= |J(\Phi_{n_a})(\Phi_{n_a-1} \circ \ldots \circ \Phi_1(x)) \cdot J(\Phi_{n_a-1} \circ \ldots \circ \Phi_1(x))| \\
&= |J(\Phi_{n_a})(\Phi_{n_a-1} \circ \ldots \circ \Phi_1(x))| \ldots |J\Phi_1(x)| \\
&\overset{(11)}{=} \frac{f_{S(n_a)}(\Phi_{n_a} \circ \ldots \circ \Phi_1(x))}{g_{n_a-1}(\Phi_{n_a-1} \circ \ldots \Phi_1(x))} \frac{f_{S(n_a-1)}(\Phi_{n_a-1} \circ \ldots \circ \Phi_1(x))}{g_{n_a-2}(\Phi_{n_a-2} \circ \ldots \Phi_1(x))} \cdots \frac{f_{S(1)}(\Phi_1(x))}{g_0(x)}.
\end{aligned}$$

Because of

$$g_i(x) = f_{S(i)}(\Phi_i(x)), \tag{12}$$

this leads, if $S(n_a) = 1$, to the deformation equation (1). By virtue of equations (10), (11) and (12), every single BDM step can be computed with our numerical realisation (section 2).

Lemma 1 *The blending parameter $S(i)$ in the i-th adaptation step has to be chosen as*

$$S(i) = \frac{\left(\sqrt[n_a]{\gamma}\right)^i - 1}{\left(\frac{f}{g_0} - 1\right)_{\max} - \gamma_i \left(\frac{f}{g_0} - 1\right)_{\min}}. \tag{13}$$

In the case of an equidistributed initial grid, the computation of $S(i)$ simplifies to

$$S(i) = \frac{\left(\sqrt[n_a]{\gamma}\right)^i - 1}{f_{\max} + \left(\sqrt[n_a]{\gamma}\right)^i (1 - f_{\min}) - 1}. \tag{14}$$

Proof: Statement (14) follows directly from equation (13), which we obtain by a straightforward calculation starting from

$$\frac{(f_{S(i)}/g_0)_{\max}}{(f_{S(i)}/g_0)_{\min}} \overset{!}{=} \gamma_i.$$

$\square$

In practical computations with the RDM, all BDM steps are subject to numerical inaccuracy. In the i-th step, the obtained area distribution g_i only approximates $f_{S(i)}$, the desired one. This is in contrast to the continuous case, where these functions coincide. Thus, we compute g_i in the same way as g_1 in the BDM instead of just defining $g_i := f_{S(i)}$ as in the continuous case (algorithm 2). With the RDM, we can easily compute test problem 1 for $\varepsilon = 0.018$ and even $\varepsilon = 0.005$ (figure 6). We prescribe $\gamma_0 = 10$ resulting in 2 adaptation steps and employ a tensor product grid with 1,024 elements. In contrast to this, the BDM leads to invalid grids in this case regardless of the IVP solver, the number of time steps and the method for gradient recovery.

The choice of γ_0 is of largely heuristic nature. The numerical tests presented lead to the practical guess of $\gamma_0 = 10$ which is chosen for all subsequent tests. For practical computations, the break-up into several less harsh deformation problems guided by γ and the choice of adaptation steps described above leads to a significant gain of robustness and thus enlarges the class of solvable deformation problems considerably.

Remark 2 *Computing test problem 1 with $\varepsilon = 0.001$ still leads to tangled grids in spite of the improvements of the deformation algorithm. This is due to the fact that our grid deformation method is able to control the area of the elements, but not their shape. With decreasing ε, the minimal angle in*

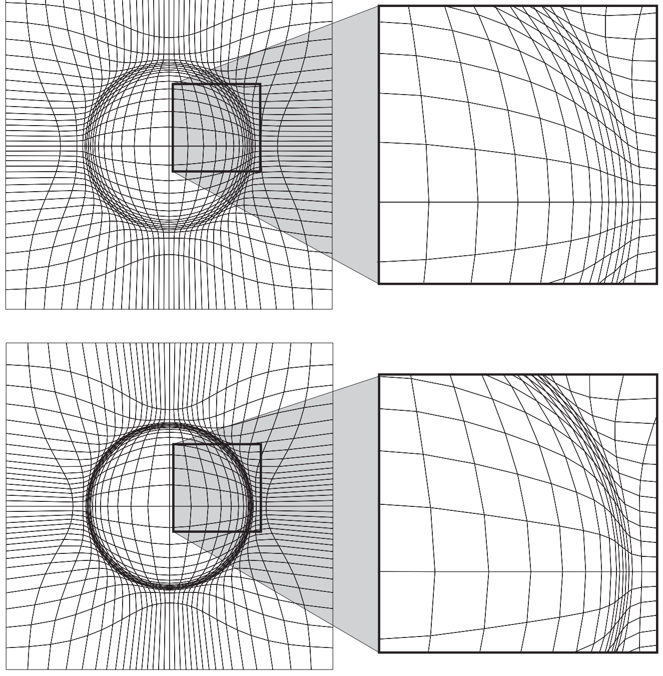


Fig. 6. Resulting grid for test problem 1, $\varepsilon = 0.018$ (top) and $\varepsilon = 0.005$ (bottom), 1,024 elements. The comparison with figure 5 demonstrates the improved grid quality.

the resulting grid tends to 0 and the maximal angle to π. Then, even very small numerical errors during deformation may lead to non-convex elements. As a remedy, one can apply grid smoothing after every adaptation step. Even though numerical tests provide encouraging results, a detailed investigation on combining grid smoothing and the RDM is beyond the scope of this paper. However, for practical computations, where the monitor function may stem from an estimated error distribution and where grid deformation is possibly iterated many times, smoothing and filtering techniques will play a much more prominent role than for the test cases we consider in this article.

5 The Multilevel Deformation Method

For practical computations, besides accuracy and robustness, the runtime behaviour of a deformation method is important, as the deformation time shall only be a small fraction of the overall runtime. We now test every main component of the RDM with respect to numerical complexity and time consumption.

To solve the Poisson problem (3) in the RDM, we employ multigrid solvers which grow linearly with the number of unknowns and therefore are of optimal complexity, at least for reasonable grids. The same applies obviously for the actual IVP solve, given a constant number of IVP time steps. In con-

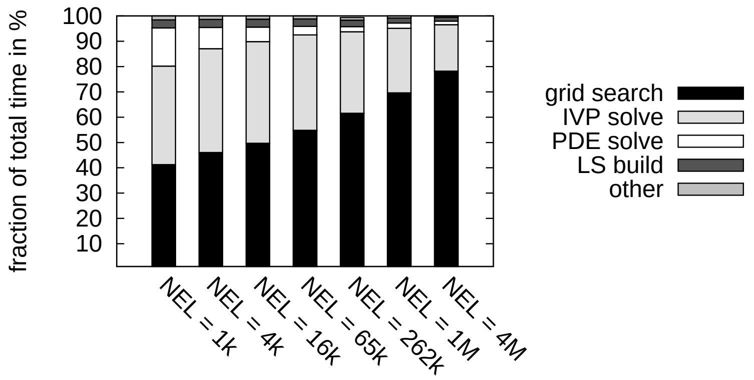


Fig. 7. Relative timings for selected components of the RDM

trast to this, the amount for grid search during the IVP solve behaves like $NEL^{3/2}$ (section 3). Thus the time for searching the grid dominates the overall deformation time on sufficiently fine grids.

Remark 3 *In order to obtain convergence, we need to double the number of time steps per refinement step of the initial grid (see section 2, corollary 1). Because of this, the search path length remains bounded and the asymptotic complexity is $\mathcal{O}(NEL^{3/2})$ like in the case of constant time step size. However, the former choice of the time step size implies by far longer computational times than the latter one.*

We now aim at accelerating the RDM in the case of constant time step size and neglect in a first step the convergence behaviour. We compute test problem 1 with $\varepsilon = 0.1$ on an equidistant tensor product grid with the RDM and $n_a = 1$ according to formula (9). The IVPs are computed employing 10 equidistant time steps combined with raytracing search. We solve (3) with multigrid which performs four line ADI steps [24] for pre- and postsmoothing, respectively. The corresponding coarse grid problems are solved with the method of conjugate gradients (CG). The multigrid iteration stops when the norm of the residuum falls below 10^{-9}. For the tests, we utilise the same computer and infrastructure as before. The relative timings (figure 7) confirm the anticipated dominance of the time for raytracing search caused by its superlinear growth. Searching the grid ("grid search") takes approx. 60% of the overall computational time on the finest grid, whereas the actual IVP solve ("IVP solve") and the assembly of the linear system for the PDE (3) ("LS build") form a small part only. The same holds for its solve time ("PDE solve"). All other operations occurring in the RDM like scaling the monitor function, computing the current area distribution or computing quality measures are embraced in "other".

Accelerating the RDM therefore implies decreasing the overall search time. From now on, we assume and exploit that our sequence of initial grids is constructed by repeated regular refinement of one coarse grid. This is a much stronger condition than the similarity condition (6) considered so far. We assume that at least on fine grids, regular refinement and applying the RDM almost commute, i.e. that the grids $\mathcal{T}_1$ resulting from applying the RDM after one step of regular refinement and $\mathcal{T}_2$ coming from one step of regular refinement after RDM are very similar. Compared to the deformation algorithms presented so far, the average search path length is halved for $\mathcal{T}_2$. Applying the RDM to the pre-deformed grid $\mathcal{T}_2$ will not change the positions of the vertices significantly. Therefore we can expect short average search paths for this deformation process resulting in short search times on the finest grid. We define the search path length as in section 3. Iterating this two-level approach leads to the *multilevel deformation method* (MDM, alg. 3). In contrast to the BDM and RDM, we formulate the MDM on a discrete level, as its key idea, exploiting grid hierarchies, does not make sense on continuous level. Consequently, its output is now a grid $\mathcal{T}$ instead of a mapping Φ.

Algorithm 3: Multilevel Deformation Method (MDM)

```
    **Input**: $f$, $g$, $\gamma_0$, $i_{\min}$, $i_{\max}$, $i_{\text{incr}}$, $N_{\text{pre}}(i)$, $\mathcal{T}$
    **Output**: $\mathcal{T}$

    $\mathcal{T}_{i_{\min}} :=$ Coarsen($\mathcal{T}$, $i_{\min}$);
    for $i = i_{\min}$ to $i_{\max}$ do
        $\mathcal{T}_i :=$ GridSmooth($\mathcal{T}_i, N_{\text{pre}}(i)$);
        if $i - i_{\min} \mod i_{\text{incr}} = 0$ then $\mathcal{T}_i :=$ RDM($f, g, \gamma_0, \mathcal{T}_i$);
        if $i < i_{\max}$ then $\mathcal{T}_{i+1} :=$ Refine($\mathcal{T}_i$)
    end
    $\mathcal{T} := \mathcal{T}_{i_{\max}}$;
```

By `Refine`, we denote one step of regular refinement of $\mathcal{T}$. The coarsening operation `Coarsen`$(\mathcal{T}, i)$ is performed by omitting all vertices of $\mathcal{T}$ not belonging to the coarse grid on level i. The grid smoothing operation `GridSmooth` is realised by applying $N_{\text{pre}}(i)$ steps of Laplacian smoothing. It must not be confused with the similarly named building block of a multigrid solver for linear systems.

We again consider test problem 1 with the same settings as above, but using the MDM instead. We set $i_{\text{incr}} = 1$ and $i_{\min} = 3$ leading to an initial grid consisting of 256 elements. On level $i_{\min}$, the number of BDM steps inside the RDM is chosen according to (9). For higher levels, we only allow one BDM step. We set $N_{\text{pre}} \equiv 2$. As desired, the average search path length remains bounded for MDM (figure 8, right) even for a step size constant with respect to the level of refinement. Based upon this numerical result, we now prove *optimal complexity* of the MDM.

Lemma 2 *We assume that for the MDM, the average search path length is bounded independently of the level of refinement. Then, the total number of IVP time steps N_{solve} as well as the total number N_{search} of point-in-element-tests during search grow at most like NEL.*

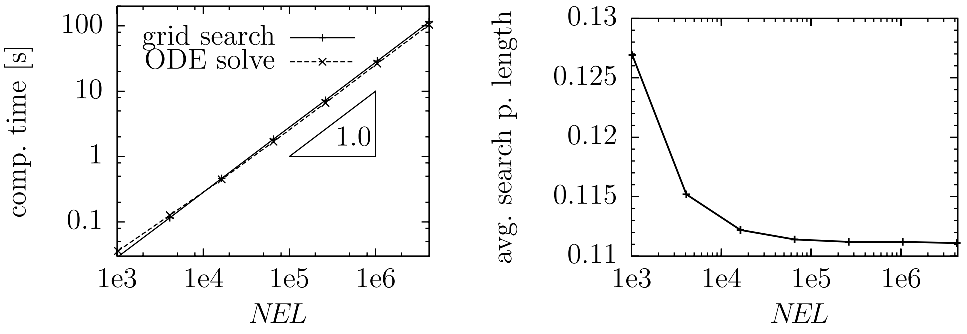


Fig. 8. Computational times for grid search and IVP solve for the MDM (left) and average search path length (right) vs. *NEL*

Proof: Let us denote the number of nodes on the grid on level $i_{\min}$ by N_0. Further, we assume $i_{\text{incr}} = 1$ without loss of generality. Then, the grid on the finest level consists of less than $2N_0 \cdot 4^{(i_{\max}-i_{\min})}$ grid points. As all grids on the levels in between $i_{\min}$ and $i_{\max}$ are deformed, totally less than $2\sum_{i=i_{\min}}^{i_{\max}} N_0\, 4^{i-i_{\min}}$ grid points are moved, and thus

$$\begin{aligned} N_{solve} \leq 2C \sum_{i=i_{\min}}^{i_{\max}} N_0 4^{i-i_{\min}} &\leq 2CN_0N \sum_{i=i_{\min}}^{i_{\max}} \frac{1}{N_0 4^{i_{\max}-i_{\min}}} 4^{i-i_{\min}} \\ = 2CN \sum_{i=i_{\min}}^{i_{\max}} \left(\frac{1}{4}\right)^{i-i_{\max}} &= 2CN \sum_{k=0}^{i_{\max}-i_{\min}} \left(\frac{1}{4}\right)^{k} \leq \frac{8}{3}CN \end{aligned}$$

where C stands for the maximum over the number of time steps. This proves the first part of the statement. The second one follows from the boundedness of the search path length. □

For test problem 1, the computational time for grid search and IVP solve grows linearly (cf. figure 8) when applying the MDM in contrast to $\mathcal{O}(NEL^{3/2})$ for the RDM. Regarding the computational time as measure for their numerical complexity, these findings confirm the statement of lemma 2.

The MDM performs much faster than the RDM method on fine grids (figure 9). Moreover, it produces more accurate results indicated by smaller quality measures (table 5). This confirms our assumption, that on fine grids, regular refinement and grid deformation almost commute. For coarse grids up to 65,000 elements, the quality measures of the MDM are inferior to the ones obtained from the RDM. This likely stems from the poor resolution of the desired area distribution on coarse grids such that the RDM step on the fine level can only partially be regarded as correction step. However, the MDM is intended for very fine grids only.

The MDM requires the user to choose the parameters $i_{\min}$, i_{incr} and $N_{\text{pre}}(i)$. We now investigate numerically how the accuracy of the MDM and the quality of the resulting grid depend from $i_{\min}$ and N_{pre}. We again compute test

NEL	Q_0	Q_∞	$h_{\min}$	$\alpha_{\min}$	$\alpha_{\max}$	runtime [s]
1,024	$5.817 \cdot 10^{-2}$	$2.630 \cdot 10^{-1}$	$1.10 \cdot 10^{-2}$	41.13	140.41	$1.02 \cdot 10^{-1}$
16,384	$7.833 \cdot 10^{-3}$	$6.622 \cdot 10^{-2}$	$2.56 \cdot 10^{-3}$	36.80	143.62	$8.46 \cdot 10^{-1}$
262,144	$1.422 \cdot 10^{-3}$	$1.668 \cdot 10^{-2}$	$6.32 \cdot 10^{-4}$	35.81	144.32	$1.07 \cdot 10^{1}$
1,048,576	$6.626 \cdot 10^{-4}$	$8.372 \cdot 10^{-3}$	$3.15 \cdot 10^{-4}$	35.68	144.37	$3.97 \cdot 10^{1}$
4,194,304	$3.190 \cdot 10^{-4}$	$4.208 \cdot 10^{-3}$	$1.57 \cdot 10^{-4}$	35.63	144.41	$1.60 \cdot 10^{2}$
1,024	$6.449 \cdot 10^{-2}$	$3.004 \cdot 10^{-1}$	$1.13 \cdot 10^{-2}$	44.66	136.92	$9.09 \cdot 10^{-2}$
16,384	$8.380 \cdot 10^{-3}$	$7.295 \cdot 10^{-2}$	$2.48 \cdot 10^{-3}$	38.21	142.21	$8.26 \cdot 10^{-1}$
262,144	$1.476 \cdot 10^{-3}$	$1.836 \cdot 10^{-2}$	$6.10 \cdot 10^{-4}$	37.20	142.98	$1.06 \cdot 10^{1}$
1,048,576	$6.808 \cdot 10^{-4}$	$9.203 \cdot 10^{-3}$	$3.05 \cdot 10^{-4}$	37.10	142.99	$4.12 \cdot 10^{1}$
4,194,304	$3.266 \cdot 10^{-4}$	$4.625 \cdot 10^{-3}$	$1.52 \cdot 10^{-4}$	37.05	142.98	$1.62 \cdot 10^{2}$
1,024	$8.108 \cdot 10^{-2}$	$3.045 \cdot 10^{-1}$	$1.23 \cdot 10^{-2}$	46.33	135.65	$5.97 \cdot 10^{-2}$
16,384	$9.091 \cdot 10^{-3}$	$8.277 \cdot 10^{-2}$	$2.53 \cdot 10^{-3}$	37.84	142.52	$8.15 \cdot 10^{-1}$
262,144	$1.535 \cdot 10^{-3}$	$2.000 \cdot 10^{-2}$	$6.22 \cdot 10^{-4}$	36.71	143.46	$1.10 \cdot 10^{1}$
1,048,576	$7.050 \cdot 10^{-4}$	$1.001 \cdot 10^{-2}$	$3.10 \cdot 10^{-4}$	36.61	143.47	$4.16 \cdot 10^{1}$
4,194,304	$3.373 \cdot 10^{-4}$	$5.007 \cdot 10^{-3}$	$1.55 \cdot 10^{-4}$	36.56	143.48	$1.62 \cdot 10^{2}$

NEL	Q_0	Q_∞	$h_{\min}$	$\alpha_{\min}$	$\alpha_{\max}$	runtime [s]
1,024	$1.380 \cdot 10^{-1}$	$6.899 \cdot 10^{-1}$	$3.67 \cdot 10^{-3}$	15.77	168.34	$1.47 \cdot 10^{-1}$
16,384	$2.047 \cdot 10^{-2}$	$1.557 \cdot 10^{-1}$	$4.95 \cdot 10^{-4}$	8.39	172.52	$1.10 \cdot 10^{0}$
262,144	$3.117 \cdot 10^{-3}$	$3.735 \cdot 10^{-2}$	$1.19 \cdot 10^{-4}$	7.92	172.31	$1.35 \cdot 10^{1}$
1,048,576	$1.370 \cdot 10^{-3}$	$1.872 \cdot 10^{-2}$	$5.94 \cdot 10^{-5}$	7.93	172.18	$4.88 \cdot 10^{1}$
4,194,304	$6.371 \cdot 10^{-4}$	$9.348 \cdot 10^{-3}$	$2.96 \cdot 10^{-5}$	7.92	172.13	$2.01 \cdot 10^{2}$

Table 3
Quality measures, minimal element size $h_{\min}$ as well as minimal and maximal angles $\alpha_{\min}$, $\alpha_{\max}$ and runtime (AMD Opteron 250), $i_{\min} = 2$ (upper part), $i_{\min} = 3$ (medium part) and $i_{\min} = 4$ (lower part) for selected levels of refinement, test problem 1 with $\varepsilon = 0.1$ and $i_{min} = 4, \varepsilon = 0.01$ (lower table)

problem 1 with the MDM using the same settings as above for $i_{\min} = 2, \ldots 4$ yielding initial grids consisting of 64, 256 and 1,024 elements, respectively. The influence of $i_{\min}$ is small and decreases with growing grid level (table 3). This holds true both for the quality measures and the quantities measuring the grid quality. The difference of the runtimes turned out to be insignificant. These experiments indicate that the MDM is robust with respect to the choice of $i_{\min}$ given that the deformation problem is solvable on the initial grid.

We set $i_{\min}$ to 3, but now vary N_{pre}. We compute test problem 1 with the same parameter settings as above and choose the N_{pre} for all grid refinement levels to 2 and 4. As the computational time for grid smoothing contributes only insignificantly to the overall runtime, we omit the runtime for these tests. The MDM turns out to be robust with respect to the choice of smoothing steps as

NEL	Q_0	Q_∞	$h_{\min}$	$\alpha_{\min}$	$\alpha_{\max}$
1,024	$6.449 \cdot 10^{-2}$	$3.004 \cdot 10^{-1}$	$1.13 \cdot 10^{-2}$	44.66	136.92
16,384	$8.380 \cdot 10^{-3}$	$7.295 \cdot 10^{-2}$	$2.48 \cdot 10^{-3}$	38.21	142.21
262,144	$1.476 \cdot 10^{-3}$	$1.836 \cdot 10^{-2}$	$6.10 \cdot 10^{-4}$	37.20	142.98
1,048,576	$6.808 \cdot 10^{-4}$	$9.203 \cdot 10^{-3}$	$3.05 \cdot 10^{-4}$	37.10	142.99
4,194,304	$3.266 \cdot 10^{-4}$	$4.625 \cdot 10^{-3}$	$1.52 \cdot 10^{-4}$	37.05	142.98
1,024	$6.112 \cdot 10^{-2}$	$2.842 \cdot 10^{-1}$	$1.12 \cdot 10^{-2}$	44.24	137.43
16,384	$6.921 \cdot 10^{-3}$	$7.020 \cdot 10^{-2}$	$2.50 \cdot 10^{-3}$	38.29	142.20
262,144	$8.221 \cdot 10^{-4}$	$1.746 \cdot 10^{-2}$	$6.14 \cdot 10^{-4}$	37.44	142.72
1,048,576	$2.878 \cdot 10^{-4}$	$8,731 \cdot 10^{-3}$	$3.06 \cdot 10^{-4}$	37.32	142.76
4,194,304	$1.040 \cdot 10^{-4}$	$5.231 \cdot 10^{-3}$	$1.53 \cdot 10^{-4}$	37.27	142.77

Table 4
Quality measures, minimal element size $h_{\min}$ as well as minimal and maximal angles $\alpha_{\min}$ and $\alpha_{\max}$, $N_{\text{pre}} = 2$ (upper part) and $N_{\text{pre}} = 4$ (lower part) for selected levels of refinement, test problem 1 with $\varepsilon = 0.1$

	Q_0			Q_∞		
NEL	one-lev	$i_{\text{incr}} = 1$	$i_{\text{incr}} = 2$	one-lev	$i_{\text{incr}} = 1$	$i_{\text{incr}} = 2$
1,024	$8.11 \cdot 10^{-2}$	$6.45 \cdot 10^{-2}$	$8.11 \cdot 10^{-2}$	$3.05 \cdot 10^{-1}$	$3.01 \cdot 10^{-1}$	$3.05 \cdot 10^{-1}$
4,096	$3.08 \cdot 10^{-2}$	$2.27 \cdot 10^{-2}$	$2.40 \cdot 10^{-2}$	$1.75 \cdot 10^{-1}$	$1.45 \cdot 10^{-1}$	$1.62 \cdot 10^{-1}$
16,384	$1.08 \cdot 10^{-2}$	$8.38 \cdot 10^{-3}$	$1.10 \cdot 10^{-2}$	$8.26 \cdot 10^{-2}$	$7.30 \cdot 10^{-2}$	$9.33 \cdot 10^{-2}$
65,536	$3.74 \cdot 10^{-3}$	$3.39 \cdot 10^{-3}$	$3.38 \cdot 10^{-3}$	$4.67 \cdot 10^{-2}$	$3.68 \cdot 10^{-2}$	$4.08 \cdot 10^{-2}$
262,144	$3.61 \cdot 10^{-3}$	$1.48 \cdot 10^{-3}$	$1.80 \cdot 10^{-3}$	$3.69 \cdot 10^{-2}$	$1.84 \cdot 10^{-2}$	$2.33 \cdot 10^{-2}$
1,048,576	$5.71 \cdot 10^{-3}$	$6.81 \cdot 10^{-4}$	$7.03 \cdot 10^{-4}$	$5.18 \cdot 10^{-2}$	$9.20 \cdot 10^{-3}$	$9.94 \cdot 10^{-3}$
4,194,304	$7.02 \cdot 10^{-3}$	$3.27 \cdot 10^{-4}$	$3.99 \cdot 10^{-4}$	$6.27 \cdot 10^{-2}$	$4.63 \cdot 10^{-3}$	$5.80 \cdot 10^{-3}$

Table 5
Quality measures for test problem 1 computed with one-level deformation (left part) and MDM with $i_{\text{incr}} = 1$ (medium part) and $i_{\text{incr}} = 2$ (right part)

well (table 4). Increasing N_{pre} from 2 to 4 leads to a significant improvement of Q_0 which is particularly notable on very fine levels. However, Q_∞ changes much less, and the difference in the other grid-related quantities displayed are minor to insignificant. It is possible to set grid smoothing aside at all, but this lowers grid quality and deformation accuracy. Overall, the MDM benefits from grid smoothing, but is robust with respect to the number of smoothing steps, i.e. changing their number will not lead to significantly different resulting grids.

As obviously the RDM steps on the finer grids change the grid only minimally, we may omit some “intermediate” levels for further acceleration. We set $i_{\text{incr}} = 2$ in the MDM and repeat the tests from above. For even levels of refinement, the starting level $i_{\min}$ is set to 4, for odd ones to 3 resulting in initial grids with 256 and 1024 elements, respectively. This is to guarantee that the actual level increment is always two. The corresponding quality measures (table 5) show a

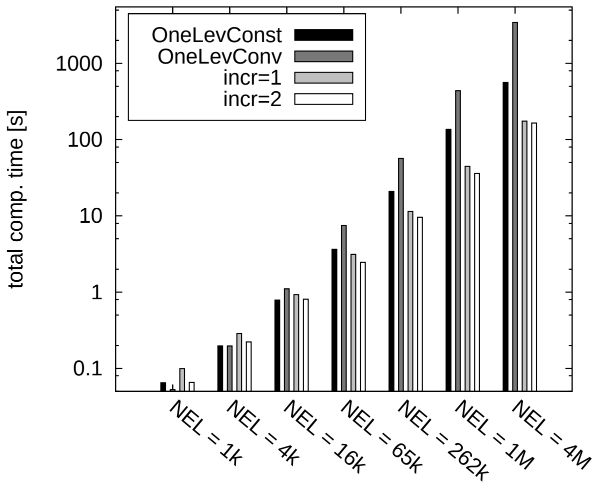


Fig. 9. Runtime comparisons for the RDM with constant and increasing step size and the MDM with level increment 1 and 2

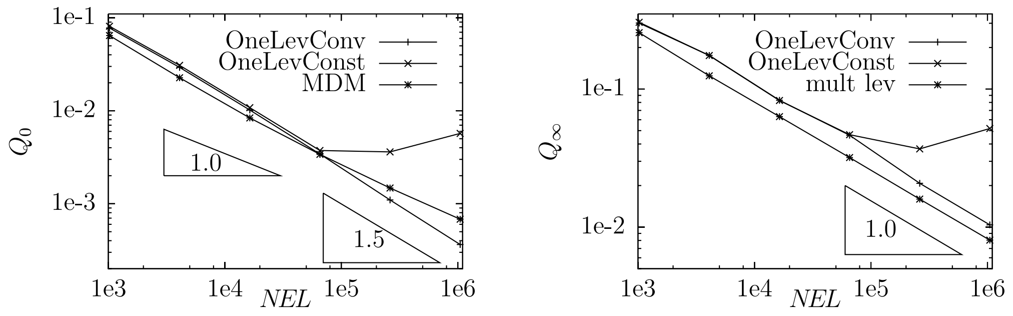


Fig. 10. Quality measures vs. *NEL* for the RDM and the MDM

slight decline compared to the case of $i_{\text{incr}} = 1$, but they are still much better than the ones of the RDM on grids with 10^6 elements and more (cf. table 5).

The comparison of the overall runtimes for the different deformation algorithms in figure 9 demonstrates that by the MDM, a significant speed-up can be achieved. Here, we denote by OneLevConst the variant of the RDM with constant size of IVP time steps independent of the refinement level. OneLevConv stands for the variant where the number of time steps doubles per refinement in order to achieve convergence. The runtimes are measured on the same computer and with the same settings as above. The tests were repeated until the standard deviations dropped below 1%. On high levels of refinement, OneLevConv is up to 3 times slower than OneLevConst confirming the statements of remark 3. The speed-up of the MDM increases with the level of refinement due to the different orders of complexity.

Our numerical tests revealed the superior accuracy of the MDM. We now revisit the convergence aspects of section 2 and compare the MDM with the different variants of the RDM. The convergence observed for OneLevConv relies on doubling the number of time steps per regular refinement. OneLevConst does not converge at all, as the IVP-induced error does not decrease

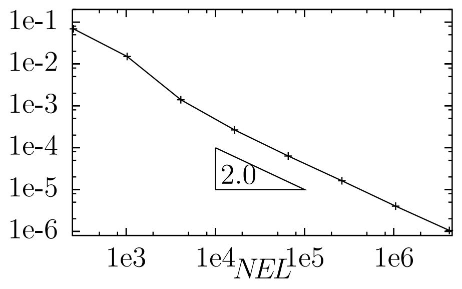


Fig. 11. d_k vs. *NEL*, test problem 1

because of its fixed time step size. We now compare these variants of the RDM with our MDM with $i_{\min} = 3$, $i_{\text{incr}} = 1$ and $N_{\text{pre}} = 2$. The results (figure 10) indicate that *the MDM features first order convergence*, i.e. $Q_0 = \mathcal{O}(h)$ and $Q_\infty = \mathcal{O}(h)$. Moreover, *the quality measures are comparable with OneLevConv* for all investigated levels of refinement. Runtime comparisons (figure 9) however demonstrate that the MDM is up to 15 times faster.

The accuracy of any of our deformation algorithms is limited by three error sources: the consistency error, the error induced by solving (3) numerically and the error resulting from the approximate solution of all IVPs. All three errors must thus decrease with grid refinement due to the convergence observed. For the consistency error being $\mathcal{O}(h)$, this is obvious. The same holds true for the error of the solution of (3). Apparently even the IVP-induced error decreases as well despite the constant time step size. This requires further investigations.

The MDM works as obviously regular refinement and applying the RDM almost commutes on sufficiently fine grids. This makes us interpret the RDM step on an intermediate level as correction step on a pre-deformed grid. We expect that these corrections become smaller the finer the grid is. For refinement level k, we measure for any grid point x the euclidean distance to its counterpart X after the RDM step and define its maximum $d_k := \max_{x\in\mathcal{V}} ||x - X||$. Denoting the associated mapping by Φ_k, we obtain $||x - \Phi_k(x)|| \leq d_k \,\forall\, x \in \mathcal{V}$. Let us furthermore assume that $d_k = \mathcal{O}(h^\tau), \tau > 1$. We reuse the notions X_h and $\tilde{X}$ from section 2 and assume that the *relative error induced by the approximate IVP solve* is bounded, i.e.

$$\frac{||X_h - \tilde{X}||}{||x - X||} \leq C \quad \forall x \in \mathcal{V}.$$

This is a by far weaker condition on the IVP solve than convergence which we required so far. It follows $||X_h - \tilde{X}|| = \mathcal{O}(h^\tau)$ and the IVP-induced error decreases as h^τ *without increasing the number of IVP time steps.* Together with the decay of the other two error sources, this explains the convergence observed. For test problem 1, numerical tests show that $d_k = \mathcal{O}(h^2)$ (figure 11). We summarise our results in the following conjecture.

Conjecture 1 *Let the sequence of initial grids $(\mathcal{T}_i)_{i\in I}$ emerge from regular refinement of one coarse grid and assume $0 < \varepsilon < f \in \mathcal{C}^1(\bar{\Omega})$ and $0 < g_{\min} < g < g_{\max}$. Then, the MDM*

a) is of optimal asymptotic complexity $\mathcal{O}(NEL)$
b) converges with first order: $Q_0 = \mathcal{O}(h)$, $Q_\infty = \mathcal{O}(h)$.

6 Applications

In their investigation on rigid particaluate flows, Turek and Wan [25] considered among other examples the numerical simulation of the sedimentation of many balls with same size using the MDM combined with the fictitious boundary method (FBM). The flow field was computed by a special ALE formulation of the Navier-Stokes equations to compensate the grid deformation in time. The equations were discretised by nonconforming Finite Elements, the resulting linear subproblems were solved with multigrid techniques in the framework of the software package FeatFlow [3]. Solid particles could move freely through the computational grid which dynamically concentrated around the falling balls by grid deformation using the distance to the particles as monitor function. Numerical results showed the benefit of grid deformation in particulate flows with many moving rigid particles. This work inspired the following test problem.

Test Problem 2 *On $\Omega = [0, 1.5] \times [0.6]$, we denote for a ball with index i, center $(x_{c,i}, y_{c,i})$ and radius r_i the euclidean distance of (x, y) to $(x_{c,i}, y_{c,i})$ by $d_i(x, y)$. With $f_i(x, y) := \max\{(d_i(x, y) - r_i)/r_i, \varepsilon\}$, we define the global monitor function:*

$$f_{\text{mon}}(x, y) := \left(\Pi_{i=1}^4 f_i(x, y)\right)^{1/2}.$$

We consider four balls with $r_i = 0.25$ and centers $(1, 1)$, $(1.55, 1.1)$, $(4, 0.3)$ and $(3.9, 1.1)$, respectively and set $\varepsilon = 0.01$. A resulting grid is displayed in figure 12.

We compute this test problem with the MDM on an equidistant tensor product grid. We set $i_{\text{incr}} = 1$ and $i_{\min} = 3$ leading to a coarse grid with 256 elements. We solve the IVPs with an equidistant step size of 0.2 and the Poisson problem (3) with the multigrid method described above. We now stop the iteration when the residuum is decreased by a factor of 10^6. Before every refinenement step and after every RDM, we employ two steps of Laplacian smoothing. The quality measures (figure 13, left) and runtime results (figure 13, right) confirm conjecture 1.

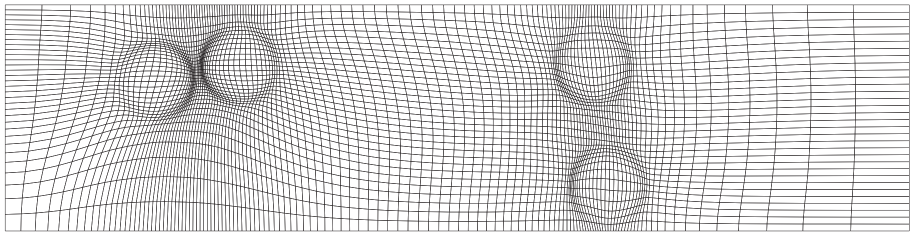
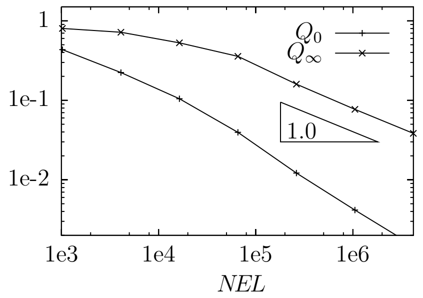


Fig. 12. Resulting grid for test problem 2



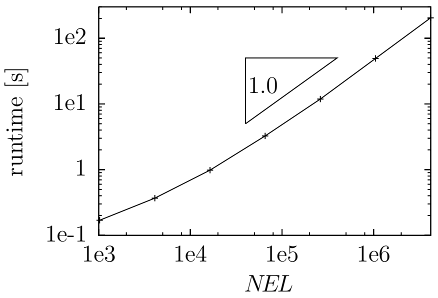


Fig. 13. Quality measures (left) and total deformation runtime (right) for test problem 2 computed with the MDM

7 Conclusions and Outlook

We presented a novel method for grid deformation based on the basic grid deformation method BDM we recently introduced [14,15]. By splitting up the deformation problem in a sequence of easier subproblems, we significantly improved the robustness of the BDM. Exploiting grid hierarchy leads to a *multilevel grid deformation method* MDM converging with optimal order. Its runtime grows linearly with the problem size. This makes the MDM suitable for applications in r- and rh-adaptive algorithms for simulating challenging real world problems. First numerical results for both static and time-dependent problems show encouraging improvements of the computational accuracy. For details, we refer to a forthcoming paper [16] devoted to r- and rh-adaptive methods based on the MDM presented here.

Acknowledgements

This work is partially supported by the German Research Association (DFG) through the collaborative research center SFB 708 and through the grant TU 102/24-1 (SPP 1253). We thank M. Möller for his valuable suggestions regarding this article. Thanks to the FEAST group (C. Becker, S. Buijssen, D. Göddeke, M. Grajewski, H. Wobker) for developing FEAST and the corresponding infrastructure.

References

[1] T. Apel, S. Grosman, P. Jimack, A. Meyer, A new methodology for anisotropic mesh refinement based upon error gradients., Appl. Numer. Math. 50 (3-4) (2004) 329–341.

[2] C. Becker, S. Kilian, S. Turek, Hardware-oriented numerics and concepts for PDE software, in: FUTURE 1095, Elsevier, 2003, pp. 1–23, international Conference on Computational Science ICCS2002, Amsterdam.

[3] C. Becker, S. Turek, Featflow - finite element software for the incompressible Navier-Stokes equations, User manual, Universität Dortmund, `http://www.featflow.de` (1999).

[4] P. B. Bochev, G. Liao, G. C. de la Pena, Analysis and computation of adaptive moving grids by deformation, Numerical Methods for Partial Differential Equations 12 (1996) 489ff.

[5] J. U. Brackbill, J. S. Saltzman, Adaptive zoning for singular problems in two dimensions, J. Comput. Phys. 46 (1982) 342–368.

[6] X.-X. Cai, D. Fleitas, B. Jiang, G. Liao, Adaptive grid generation based on least-squares finite-element method, Computers and Mathematics with Applications 48 (7-8) (2004) 1077–1086.

[7] W. Cao, W. Huang, R. D. Russell, A study of monitor functions for two-dimensional adaptive mesh generation, SIAM Journal on Scientific Computing 20 (6) (1999) 1978–1994.

[8] G. F. Carey, Computational Grids: Generation, Adaptation, and Solution Strategies, Taylor und Francis, 1997.

[9] B. Dacorogna, J. Moser, On a partial differential equation involving Jacobian determinant, Annales de le Institut Henri Poincaré 7 (1990) 1–26.

[10] W. D. de Boer, H. Z. Tang, P. A. Zegeling, Robust and efficient adaptive moving mesh solution of th 2-d euler equations, in: Recent Advances in Adaptive Computation, Contemporary Mathematics, AMS, 2005, pp. 419–430, recent Advances in Adaptive Computation 2004, Hangzhou, China.

[11] Y. Di, R. Li, T. Tang, A general moving mesh framework in 3D and its application for simulating the mixture of multi-phase flows, Communications in Computational Physics 3 (3) (2008) 582–602.

[12] Y. Di, R. Li, T. Tang, P. Zhang, Moving mesh finite element methods for the incompressible Navier-Stokes equations, SIAM Journal on Scientific Computing 26 (3) (2005) 1036–1056.

[13] L. Formaggia, S. Perotto, Anisotropic error estimates for elliptic problems, Numerische Mathematik 94 (2003) 67–92.

[14] M. Grajewski, A new fast and accurate grid deformation method for r-adaptivity in the context of high performance computing, Ph.D. thesis, Dortmund University of Technology, Logos Verlag, Berlin (2008).

[15] M. Grajewski, M. Köster, S. Turek, Mathematical and numerical analysis of a robust and efficient grid deformation method in the finite element context, SIAM J. Sci. Comput. 31 (2) (2009) 1539–1557.

[16] M. Grajewski, S. Turek, Establishing a new grid deformation method as tool in r- and rh-adaptivity, in preparation, Dortmund University of Technology, Vogelpothsweg 87, 44227 Dortmund (2010).

[17] W. Huang, Variational mesh adaptation: Isotropy and equidistribution, J. of Comput. Phys. 174 (2001) 903–924.

[18] M. Köster, D. Göddeke, H. Wobker, S. Turek, How to gain speedups of 1000 on single processors with fast FEM solvers – benchmarking numerical and computational efficiency, Ergebnisberichte des Instituts für Angewandte Mathematik, Nr. 382, Fakultät für Mathematik, TU Dortmund (Oct. 2008).

[19] G. Liao, D. Anderson, A new approach to grid generation, Applicable Analysis 44 (1992) 285–298.

[20] F. Liu, S. Ji, G. Liao, An adaptive grid method and its application to steady Euler flow calculations, SIAM Journal on Scientific Computing 20 (3) (1998) 811–825.

[21] M. Miemczyk, Hexaeder-Gittergenerierung durch Kombination von Gitterdeformations-, Randadaptions- und "Fictitious-Boundary"-Techniken zur Strömungssimulation um komplexe dreidimensionale Objekte, `http://www.mathematik.tu-dortmund.de/lsiii/static/schriften_ger.html`, diploma thesis, Dortmund University of Technology, 2007.

[22] R. Panduranga, Numerical analysis of new grid deformation method in three-dimensional finite element applications, `http://www.mathematik.tu-dortmund.de/lsiii/static/schriften_ger.html`, master thesis, Dortmund University of Technology, 2006.

[23] S. Turek, D. Göddeke, C. Becker, S. H. M. Buijssen, H. Wobker, UCHPC - UnConventional High Performance Computing for Finite Element Simulations, Tech. rep., FB Mathematik, Universität Dortmund, ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 360, submitted to CCPE (Feb. 2008).

[24] S. Turek, A. Runge, C. Becker, The FEAST indices - realistic evaluation of modern software components and processor technologies, Computers and Mathematics with Applications 41 (2001) 1431–1464.

[25] D. Wan, S. Turek, Fictitious boundary and moving mesh methods for the numerical simulation of rigid particulate flows, J. Comput. Phys. (22) (2006) 28–56.

[26] Z. Zhang, A. Naga, Validation of the a posteriori error estimator based on polynomial preserving recovery for linear elements., Int. J. Numer. Methods Eng. 61 (11) (2004) 1860–1893.

[27] O. C. Zienkiewicz, J. Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique, Int. J. Numer. Methods Eng. 33 (1992) 1331–1364.