Hamburger Beiträge
zur Angewandten Mathematik

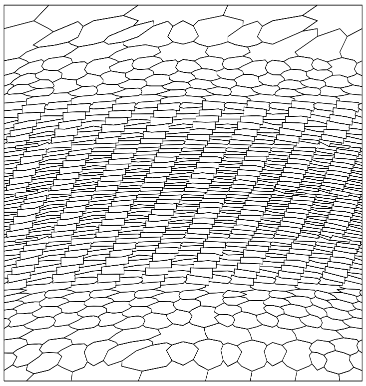
Anisotropic Adaption and Structure Detection

Ingo Thomas

Reihe F
Computational Fluid Dynamics and Data Analysis 11
August 1999

Hamburger Beiträge zur Angewandten Mathematik

Reihe A	Preprints
Reihe B	Berichte
Reihe C	Mathematische Modelle und Simulation
Reihe D	Elektrische Netzwerke und Bauelemente
Reihe E	Scientific Computing
Reihe F	Computational Fluid Dynamics and Data Analysis



Anisotropic Adaption and Structure Detection

Abstract

In this paper, we suggest a general approach of an anisotropic adaption procedure in the context of numerical schemes for hyperbolic conservation laws. This approach consists of coupling a structure detection tool from image processing with a new, non-iterative anisotropic adaption algorithm.

Ingo Thomas
Inst. f. Angew. Math.
Bundesstr. 55
20146 Hamburg
thomas@math.uni-hamburg.de

August 20, 1999

Contents

Introduction

Solutions to Euler's equations of gas dynamics or to the Navier-Stokes equations governing compressible flow of a viscous fluid are known to exhibit anisotropic structures. Typical examples can be found in shock waves and contact discontinuities as well as in boundary and interior viscous layers.

To capture these structures numerically by means of isotropic grids neither does respect the physical nature of the underlying phenomena nor does it lead to economical numerical methods. To resolve a boundary layer satisfactorily in a flow characterized by a Reynolds number of order 10^6 would require a grid spacing perpendicular to the boundary so small that an isotropic grid would very well exceed the size of computer memories of our and the following generations.

Hence, in order to design efficient discretization methods as well as to treat physically anisotropic phenomena properly a need is felt for anisotropic adaptation algorithms. By this we mean the detection of anisotropic phenomena as well as the technical algorithms ensuring that edges of the grid will be aligned with the anisotropic structures. In this context a truely anisotropic error indicator would be highly desirable.

The problems of anisotropic adaptation are well known for quite a long period of time. If a certain grid quality is violated solutions on anisotropic meshes may look weired and the anisotropic phenomena one is interested in seem to be nailed to the kinks and other topological changes in the grid, see [1]. Isotropic refinement techniques automatically ensure smooth topological changes and are thus much simpler to apply and more robust, see [9], [11]. In this report we exploit the possibilities given by the so-called structure tensor of image processing following [16] to smoothly capture the locus of anisotropy and to control a novel anisotropic grid adaptation algorithm which allows the exact capturing of anisotropic phenomena while retaining the smoothness features of isotropic algorithms.

The sections 1 to 5 are concerned with the technicalities of anisotropic grid adaptation. Given a *grid resolution function* a given mesh has to be worked upon so that the resulting grid will exhibit the anisotropies in the numerical solution. The philosophy behind this strategy is the interpretation of anisotropy as the existence of a local coordinate system, where one axis points in the direction tangential to the anisotropiv feature and the other in perpendicular direction.

In section 6 a possible anisotropic error indicator is discussed which can be used to derive the grid resolution function. In classical approaches in which the residual $r^h := \mathcal{L}u^h$ of a numerical solution u^h to a problem $\mathcal{L}u = 0$ is employed to get estimates of the error $e^h := u^h - u$ the only available information is a number $\|r^h\|$ on each cell. Based on this number a decision is made to refine the cell, to coarsen it, or to leave it in its shape, see [9], [11], [12]. A source of anisotropic information certainly can be found in the finite difference based refinement indicators of Sonar and Warnecke, see [14], [13] and it is one of our future goals to exploit this source in detail. Here, the structure tensor used in edge detection algorithms of image processing is employed as a refinement indicator. The local *coherence* is introduced as a local measure of anisotropy and error control is interpreted as fitting locally the edge lengths in anisotropic regions so that the error is equidistributed.

1 Anisotropic Adaption

1.1 Representation of an anisotropic grid resolution by symmetric positive definite matrices

Using an isotropic refinement strategy, we usually have an error indicator which decides whether a cell has to be refined (or re-coarsened) depending on the local data u. This technique may be interpreted as the existence of a scalar *grid resolution function*

$$h : \Omega \times \mathcal{U} \to (0, \infty)$$

where $\mathcal{U}$ is the space of the state variables. The structure of the actual grid is coupled to the grid resolution function by a set of discrete conditions. This condition may be simply

$$\exists c, C > 0 : \; c \leq \frac{1}{h(d)} \|d\|_2 \leq C \text{ for all edges } d. \tag{1}$$

We are interested in a generalization of the grid resolution function to the anisotropic case.
In two space dimensions, a local anisotropic structure is completely described by a local data-fitted coordinate system. This coordinate system is given by two basis vectors e_1, e_2 and the associated reference lengths h_1, h_2. A mathematically elegant way of bundling this information is a symmetric positive definite matrix. By an inverse centerline transformation we construct a 2×2 - matrix $\mathbf{H}$ with eigenvectors e_1, e_2 and eigenvalues h_1, h_2. This construction is unique, if we demand additionally $h_1 \geq h_2$. Therefore, we describe the anisotropic structure of the data by a matrix valued function [1]

$$\mathbf{H} : \Omega \times \mathcal{U} \to SPD(2) \tag{2}$$

Now we have to develop a discrete interpretation of the anisotropic refinement information. In other words: we are looking for an anisotropic generalization of (1).

1.2 Discrete interpretation of the anisotropic refinement information

A very natural generalization of condition (1) can be derived by using the elementary fact, that

$$\|\cdot\|_{\mathbf{H}} := \left\|\mathbf{H}^{-1} \cdot\right\|_2$$

is a norm on $\mathbb{R}^2$. It is in fact a rather interesting one, since it measures in units of the inverse eigenvalues of $\mathbf{H}$ in directions of the associated eigenvectors. Therefore we generalize (1) to

$$\boxed{\exists c, C > 0 : \; c < \|d\|_{\mathbf{H}} < C \text{ for all edges } d} \tag{3}$$

[1] $SPD(n)$ is the space of symmetric positive definite matrices with $n \times n$ entries

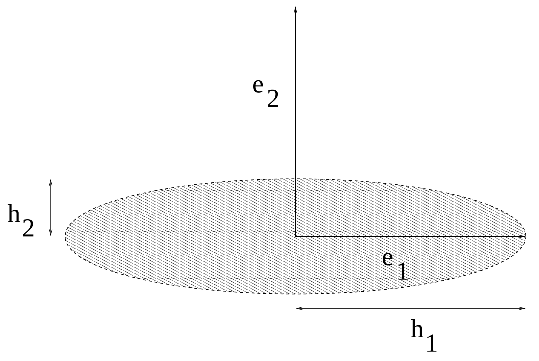


Figure 1: Local coordinate system. The ellipse is the unit disc with respect to the norm $\|\cdot\|_{\mathbf{H}}$

One can see easily that the approximation error of a finite volume scheme on an anisotropic cell is minimal if each edge is parallel to a local basis vector. Therefore we demand additionally:

$$\boxed{\max_d \min \left\{ \left| \left\langle \frac{d}{\|d\|_2}, e_1 \right\rangle \right|, \left| \left\langle \frac{d}{\|d\|_2}, e_2 \right\rangle \right| \right\} \to 0 \quad \text{for } \frac{h_2}{h_1} \to 0} \tag{4}$$

1.3 Strategy

Our object is an algorithm which fulfills the conditions (3) and (4). We first decouple this task into several steps:

1. Perform an (standard) isotropic adaption step to fulfill the *isotropic condition*

   $$\boxed{c \leq \frac{1}{h_1} \|d\|_2 \leq C} \tag{3a}$$

   (See Section 2)

2. Perform an anisotropic adaption step to fulfill the *anisotropic condition*

   $$\boxed{c \leq \frac{1}{h_2} |\langle d, e_2 \rangle| \leq C} \tag{3b}$$

   Note that (3a) and (3b) are sufficient for (3) (See Section 3).

3. We define a modified dual grid which fulfills (4) automatically on triangular grids which satisfy (3) (See Section 4).

1.4 Strategy

Our object is an algorithm which fulfills the conditions (3) and (4). We first decouple this task into several steps:

1. perform an (standard) isotropic adaption step to fulfill the *isotropic condition*

$$\boxed{c \leq \frac{1}{h_1} \|d\|_2 \leq C} \tag{3a}$$

See section 2

2. perform an anisotropic adaption step to fulfill the *anisotropic condition*

$$\boxed{c \leq \frac{1}{h_2} |\langle d, e_2 \rangle| \leq C} \tag{3b}$$

Note that (3a) and (3b) are sufficient for (3).
This is done in section 3

3. We define a modified dual grid which fulfills (4) automatically on triangular grids which fulfill (3). See section 4.

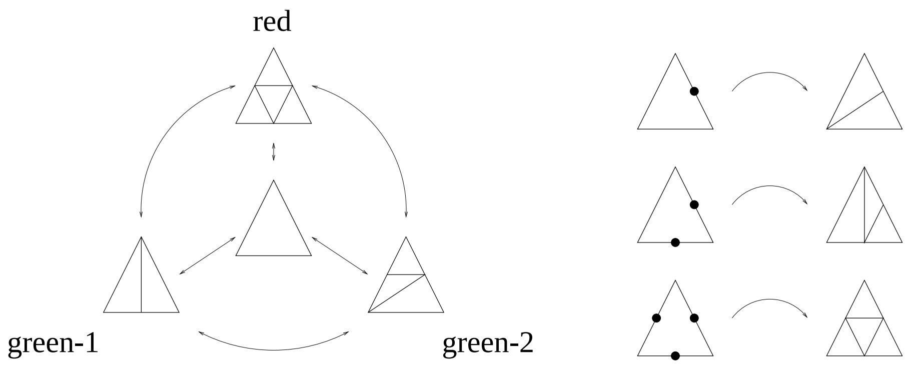


Figure 2: Left: Red and Green Refinements of a triangle, Right: Elimination of hanging nodes

2 Isotropic Adaption

An efficient and elegant way to perform the isotropic adaption is described in [5]. For the sake of completeness we give a short sketch of this technique.
The algorithm is strictly local in the sense, that it consists of subdividing single triangles of a given input triangulation. There are only three classes of subdivisions of a single triangle; these are sketched in Figure 2.
Before describing the adaption algorithm itself, we note an important invariant of our algorithm, which can be understood as some sort of smoothness of the triangle size:

$$\boxed{\Delta \cap \Delta^* \neq \emptyset \;\Rightarrow\; |\mathrm{level}\,(\Delta) - \mathrm{level}\,(\Delta^*)| \leq 1} \tag{5}$$

where

$$\mathrm{level}\,(\Delta) = \begin{cases} 0 & \text{if } \Delta \text{ is part of the input triangulation} \\ \mathrm{level}\,(\mathrm{parent}\,(\Delta)) + 1 & \text{if } \Delta \text{ is child of a red-refined cell} \end{cases}.$$

The refinement algorithm consists of three steps:

1. Eliminate all green refinements
2. Ensure condition (3):
   for each triangle Δ **do**
     if second inequality of (3a) is violated **then**
       Perform a red-refinement
     else if first inequality of (3a) is violated **then**
       if parent(Δ) is red-refined and each child of parent(Δ) violates the first inequality of (3a) **then**
         re-coarsen parent(Δ)
       end if
     end if
     Force condition (5).
   end for
3. Eliminate hanging nodes by green-refining (see Figure 2).

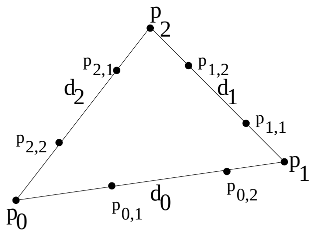


Figure 3: Notations

3 Anisotropic Adaption

3.1 Notations

A triangle Δ is always mathematically positively oriented. The vertices are $p_i, i = 0, 1, 2$ and the edges $d_i, i = 0, 1, 2$. Newly generated points on the edge d_i are denoted $p_{i,j}$. We often identify the *vector* $p_{i+1} - p_i$ with the *edge* d_i. The numbering of the vertices is cyclic, i.e. $d_3 = d_0$.

3.2 First Step: Splitting of edges

A very natural way of fulfilling (3a) is to simply split each edge which violates this condition. We have to split each edge in

$$n = \lfloor \|d\|_{\mathbf{H}} \rfloor + 1$$

"sub-edges" ($e_{1,d}, h_{1,d}$ are the grid refinement function values assigned to the edge). The vertices of the sub edges are convex combinations of the points of the triangle:

$$p_{i,j} := p_i + \frac{j}{n} p_{i+1} \tag{6}$$

It has proven useful to satisfy the prescribed point density in the corners asymptotically (for $h_2/h_1 \to 0$) exact. To do this, we modify the trivial point distribution by introducing a *point distribution function* φ:

$$p_{i,j} := p_i + \varphi\left(\frac{j}{n}\right) p_{i+1}. \tag{7}$$

We derive some conditions for the choice of φ. Trivially we need

$$\varphi(0) = 0 \quad \text{and} \quad \varphi(1) = 1.$$

Our "density condition" has the form

$$h_{2,i} \overset{!}{\xleftarrow{n\to\infty}} \varphi\left(\frac{1}{n}\right) \underbrace{|\langle p_1 - p_0, e_2\rangle|}_{=:l}.$$

Combining this with $\varphi(0) = 0$ we obtain at p_0

$$\frac{nh_{2,0}}{l} = \frac{1}{1/n}\left(\varphi\left(\frac{1}{n}\right) - \varphi(0)\right) \xrightarrow{n\to\infty} \varphi'(0).$$

Doing the same at p_1, we obtain four linear conditions:

$$\begin{array}{lcl} \varphi(0) = 0 & \text{and} & \varphi(1) = 1, \\ \varphi'(0) = \frac{nh_{1,0}}{l} & \text{and} & \varphi'(1) = \frac{nh_{1,1}}{l}. \end{array}$$

Since we need four degrees of freedom to satisfy this conditions, a cubic polynomial will do, i.e.

$$\varphi(t) := \sum_{i=0}^{3} a_i t^i.$$

The first condition yields $a_0 = 0$ and the third $a_1 = h_{2,0}$. The remaining system leads to

$$a_3 = h_{2,1} - 2 + h_{2,0} \quad \text{and} \quad a_2 = 3 - h_{2,1} - 2h_{2,0}.$$

3.3 Second step: Building new triangles

The remaining task is to construct a new triangulation which contains the newly generated points without violating condition (3b). Obviously, it is sufficient to solve this problem locally on each triangle.
We first split the newly generated points in two families according to

$$P_{\pm 1} = \{p \in p_{i,j} \, : \, \text{sgn} \, \langle \mathrm{d_i}, \mathrm{e_2} \rangle = \pm 1\}$$

We can assign the old triangle points to the families $P_{\pm 1}$ in the following way:

$$p_i \in P_{\pm 1} :\Leftrightarrow \text{sgn} \, \langle \mathrm{d_{i-1}}, \mathrm{e_2} \rangle \, \text{sgn} \, \langle \mathrm{d_i}, \mathrm{e_2} \rangle \geq 0 \wedge \text{sgn} \, (\text{sgn} \, \langle \mathrm{d_{i-1}}, \mathrm{e_2} \rangle + \text{sgn} \, \langle \mathrm{d_i}, \mathrm{e_2} \rangle) = \pm 1$$

The remaining p_i can be assigned to either of the families. We assume that the points of each family are sorted by their e_2-component (i.e. the quantity $\langle p, e_2 \rangle$).
To write down our algorithm, we assume that there are two elementary functions given, namely

- *Collinear*(x_0, x_1, x_2) tests whether the points x_i are collinear, [2]
- *Triangle*(p_0, p_1, p_2) inserts a new triangle spanned by the points p_i in the triangulation [3].

Now our algorithm reads as

while $i < |P_{+1}| - 1 \wedge j < |P_{-1}| - 1$ **do**
  if $(\langle P_{+1,i+1}, e_2 \rangle < \langle P_{-1,j+1}, e_2 \rangle \wedge \neg\text{Collinear}\,(P_{+1,i+1}, P_{+1,i}, P_{-1,j}))$
  $\vee\text{Collinear}\,(P_{+1,i+1}, P_{+1,i}, P_{-1,j})$ **then**
    $\text{Triangle}(P_{+1,i}, P_{+1,i+1}, P_{-1,j})$
    $i \leftarrow i + 1$
  else
    $\text{Triangle}(P_{-1,j}, P_{+1,i+1}, P_{-1,j})$
    $j \leftarrow j + 1$

[2] This function should be implemented as a bit operation instead of a - bad conditioned - calculation
[3] See section 5

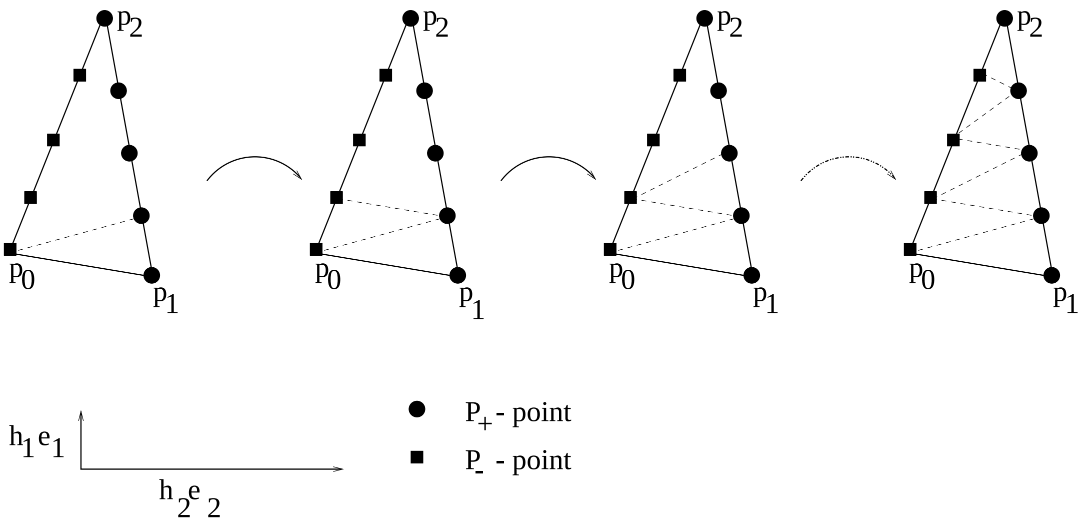


Figure 4: Algorithmus

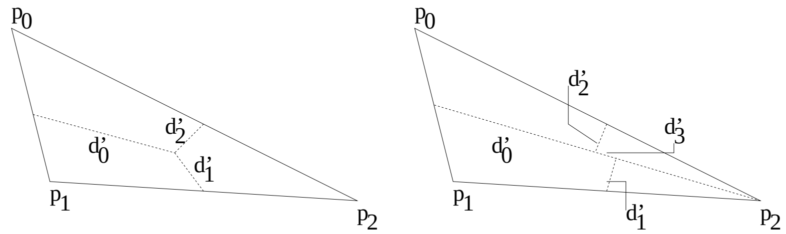


Figure 5: Left: Canonical dual mesh, Right: Modified dual mesh

    end if
  end while
  if $i + j < |P_{+1}| + |P_{-1}| - 2$ **then**
    if $i = |P_{+1}| - 1$ **then**
      for $j < |P_{-1}| - 1$ **do**
        Triangle$(P_{-1,j}, P_{-1,j+1}, P_{+1,|P|_{+1}-1})$
      end for
    else
      for $i < |P_{+1}| - 1$ **do**
        Triangle$(P_{+1,i}, P_{+1,i+1}, P_{-1,|P|_{-1}-1})$
      end for
    end if
  end if

An example for the steps of the algorithm is shown in figure 4.

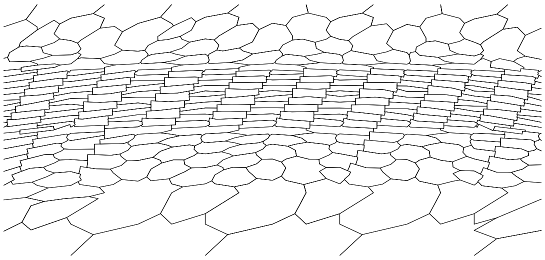


Figure 6: Modified dual mesh

4 Modified Dual grid

Up to now, we ignored condition (3b). One can see easily, that this condition is violated for the "canonical" dual mesh as described in [10]. A simple analysis shows that this problem cannot be fixed by simply moving the barycenter as suggested in [4, 6]. Their approach of an edge length-weighted movement of the barycenter does work only for right-angled triangles. Since the construction of right-angled triangles can be interpreted as the generation of a structured grid this can inherently not be done locally.

Therefore we suggest a slightly more complicated local modification of the dual mesh.

Each triangle that is constructed by the algorithm 3.3 has exactly one "short edge"; namely the edge which lies on an edge of the parent triangle. We denote this edge by d_0. Now we construct the line $m = \overline{\frac{1}{2}(p_0 + p_1)p_2}$. With the help of this line we construct all edges of the modified dual mesh. We set

$$d_1' := \overline{\frac{1}{2}(p_1 + p_2)p_{12}'} \quad \text{and} \quad d_2' := \overline{\frac{1}{2}(p_2 + p_0)p_{20}'}$$

where p_{12}' (p_{20}') is the orthogonal projection of $\frac{1}{2}(p_1 + p_2)$ ($\frac{1}{2}(p_2 + p_0)$) onto m, if this projection lies in Δ. Otherwise let p_{12}' (p_{20}') be the best-approximation of this projection. Let $p^* \in \{p_{12}', p_{20}'\}$ be the point closer to $\frac{1}{2}(p_0 + p_1)$; the other one is denoted p^{**}. We define

$$d_2' := \overline{\frac{1}{2}(p_0 + p_1)p^*}$$

and finally

$$d_3' := \overline{p^* p^{**}}.$$

The modified dual mesh of the example of the last section is shown in Figure 6.

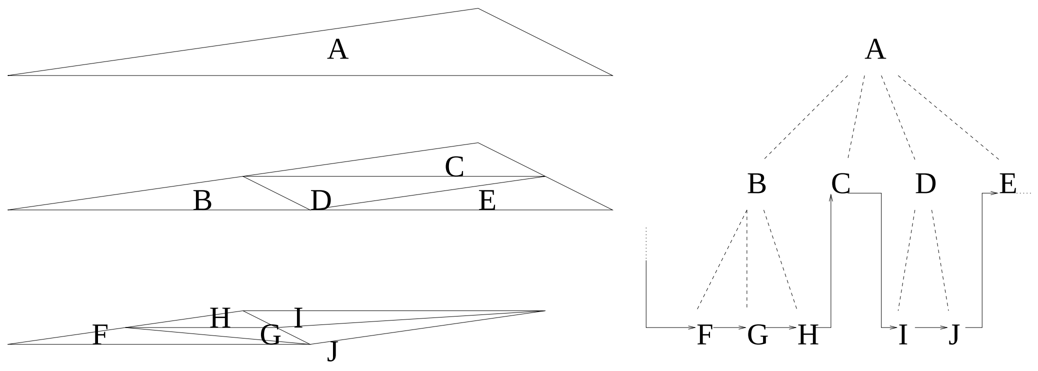


Figure 7: Left: refinement of a triangle, right: data structure

5 Data structures

5.1 Required complexity

Our objective was the implementation of an efficient explicit finite volume ENO - scheme. A very natural requirement on the design of our data strucure is that the overall complexity of that scheme is not larger than the complexity of the same ENO scheme on a fixed triangulation - which is $\mathcal{O}(n)$ for a single time step. Let us first compile the operations we do have to provide for an ENO scheme (see [2]):

1. An enumeration mechanism for all triangles,
2. Neighbourhood calculation for each triangle.

The enumeration algorithm itself has to have the complexity $\mathcal{O}(n)$. Since we need to calculate the neighbourhood of each triangle in a single time step, the neighbourhood calculation has to have the complexity $\mathcal{O}(1)$.

5.2 A threaded tree structure

From an algorithmical point of view the isotropic and the anisotropic adaption step have one important property in common: both introduce a hierarchy of triangles in a natural way. Therefore it is natural to use a tree structure with the following properties:

- Each node represents a triangle,
- The childs of a triangle build a triangulation of the parent node ,
- The leaves of the tree build a triangulation of the physical domain

The enumeration of all triangles of the actual triangulation requires the enumeration of all leaves of the tree. In a usual tree structure, this operation has the same complexity as the enumeration of all nodes - which is $\mathcal{O}(n \log n)$. Therefore we decided to use a *threaded tree* instead, i.e. we linked all leaves to a doubly linked list (see Figure 7).

In the isotropic case the position of a node in the tree alone describes all geometrical and topological information we need (in the anisotropic case we do need very few additional information). Unfortunately, the algorithms to calculate these informations have the complexity $\mathcal{O}(\log n)$, which is too large. Therefore we need to store the geometrical information (i.e. the points) as well as the neighbourhood information of each triangle explicitly.

6 Structure Detection

The most important open question in an anisotropic adaptive scheme apart from the adaption algorithm itself is the construction of an anisotropic adaption indicator (or, in the terms of section 1.1, the *grid resolution function*). We developd a nicely working approach for an anisotropic adaption indicator, which is in fact rather a residual estimator than an error estimator following the ideas of [13]; this idea will be described in a forthcoming paper.
The anisotropic error estimators we think of assume that there is some sort of local coordinate system given. Therefore, we have to think of how this coordinate systems can be computed. A close look at different kinds of error indicators [13, 3, 15] shows that the error of numerical schemes are generally tightly related to the approximation error of the unknown exact solution [8]. Therefore, we expect that the anisotropic structure of the grid should reflect the anisotropic structures of the data.
The detection of the structure of given data may be interpreted as an image processing problem. Luckily, there are efficient and powerful tools for this purpose, which allow us to introduce a structure detection step into an adaptive numerical scheme without boosting the overall complexity.

6.1 Structure Detection

Efficient Techniques for the detection of anisotropic structures in spatial scalar data are well known in the image processing community [16]. An elementary, but for our purposes sufficient tool for the detection of anisotropic structures is the *structure tensor*

$$J_\sigma(\nabla u_\delta) := \nabla u_\delta \otimes \nabla u_\delta = \begin{bmatrix} u_{\delta,x}^2 & u_{\delta,x} u_{\delta,y} \\ u_{\delta,x} u_{\delta,y} & u_{\delta,y}^2 \end{bmatrix}$$

The structure tensor consists of the tensor product of the gradient of the data with itself. To avoid the influence of local noise we do some presmoothing on the data by solving the heat equation over a short time proportional to δ. Since the structure information itself may contain sharp edges, which might lead to oddities in the grid generation, it is wise to smooth the tensor J_σ componentwise. The solution of the heat equation can be done numerically by a standard finite-element discretization (see, for instance [7]).

7 Examples

7.1 Co-operation of structure detection and anisotropic adaption

Figure 8 illustrates the co-operation of the structure detection mechanism and the anisotropic adaption algorithm. We have some model data grey-level-coded on the left hand side and the resulting grid on the right hand side.

7.2 The Euler equations

Consider the two dimensional system of Euler equations studied in [2, 10]. A standard testcase is the NACA0012 Profile with $Ma = 0.8, \alpha = 1.25$. The steady solution is known to exhibit some characteristical features: A strong shock at the upper surface, a weaker shock at the lower surface and a stagnation point at the nose of the profile.

We coupled our anisotropic framework with a standard second-order WENO solver. We used an anisotropic version of the finite difference error indicator introduced in [13], which uses the structure information of the structure tensor. This anisotropic error indicator will be described in detail in a forthcoming paper. In figure 9 we displayed the grid we obtained at steady state in some details.

8 Conclusions

The technique presented seems to be a promising approach to stable anisotropic adaption. But, from my experience, there is a still major problem in this field: The complexity of the data projection step is significantly higher than in the isotropic case. Another problem is a certain loss of stability in the numerical scheme. We had to choose a significantly smaller CFL number (0.3) than in the isotropic adaptive scheme (CFL 0.7).

9 Acknowledgments

This work was carried out as a part of the project SO/363-5-1 in the DFG Schwerpunktprogramm *Analysis und Numerik von Erhaltungsgleichungen*. The author would like to thank my university teacher Th. Sonar for all the support and encouragement I have received. It is also a pleasure to thank my colleagues Lars Hoffmann and Thorsten Grahs for many interesting discussions about the fascinating subject of structure detection.

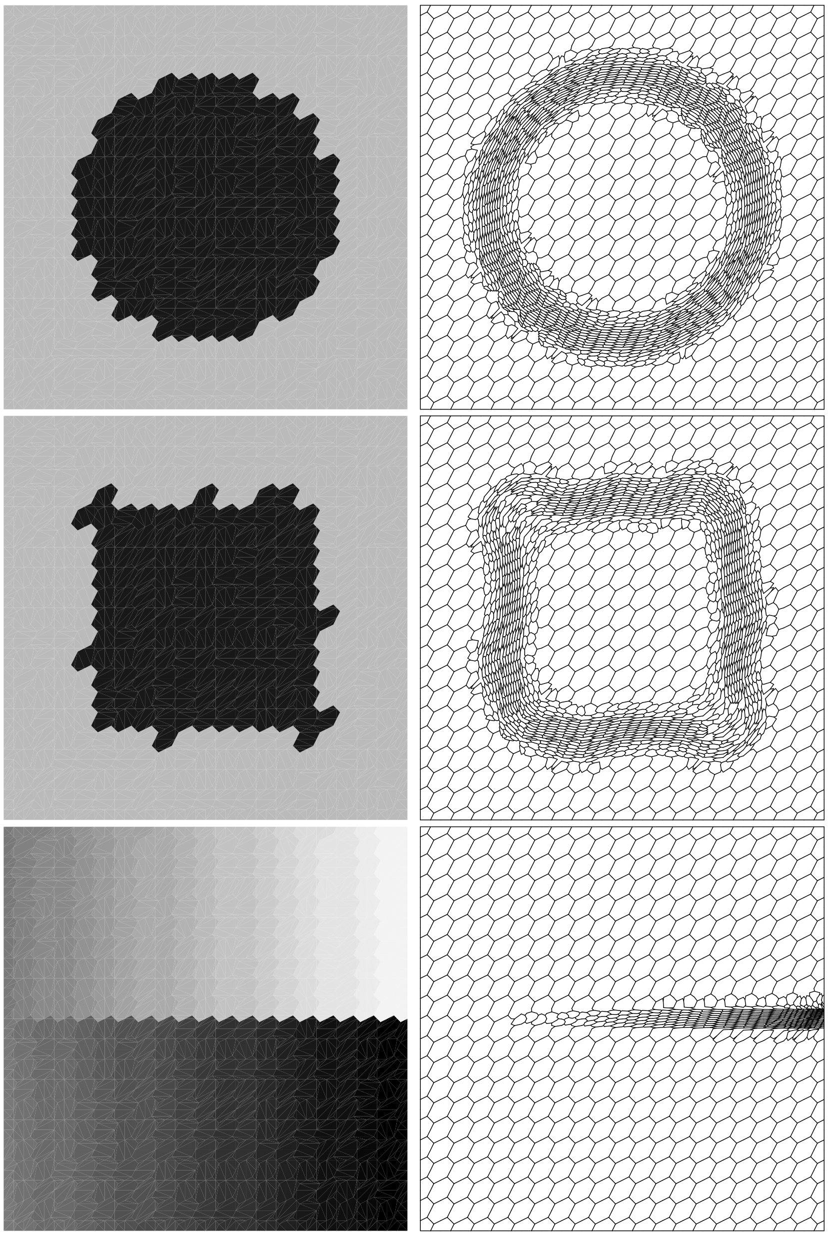


Figure 8: Left: input data, Right: data - fitted grids

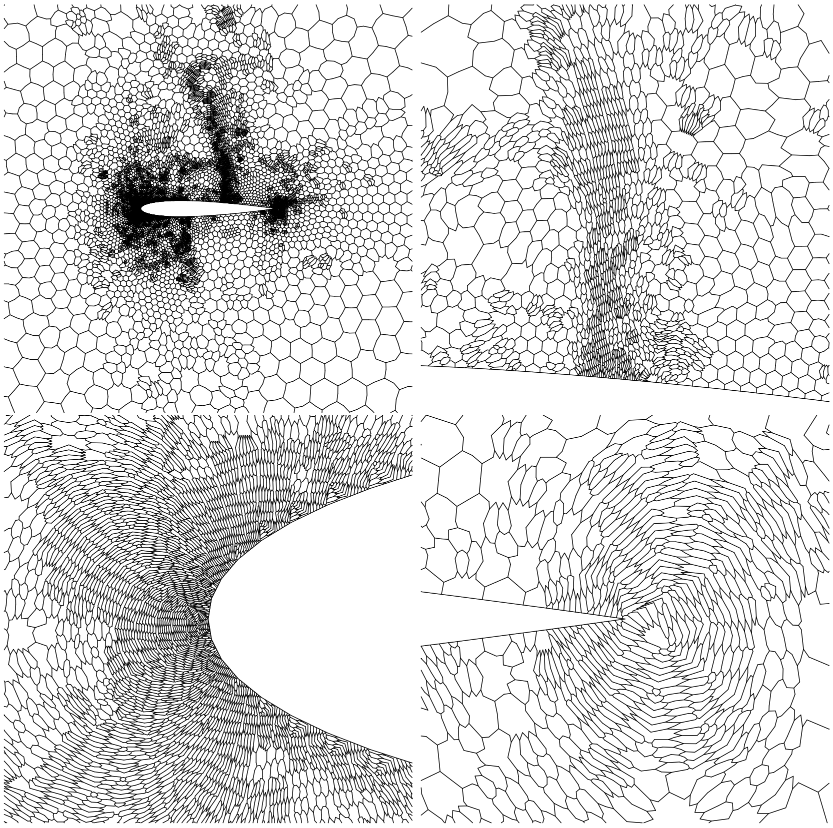


Figure 9: Grid for NACA0012, $Ma = 0.8, \alpha = 1.25$

References

[1] R. Beinert and D. Kröner, *Finite volume methods with local mesh alignment in 2d*, in Adaptive Methods-Algorithms, Theory and Applications.

[2] O. Friedrich, *Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructered grids*, Journal of Computational Physics, 144 (1997), pp. 194–212.

[3] U. Göhner and G. Warnecke, *A second order finite difference error indicator for adaptive transsonic flow problems*, Numerische Mathematik, 70 (1995), pp. 129–161.

[4] V. Hannemann, *Numerische Simulation von Stoß-Stoß-Wechselwirkungen unter Berücksichtigung von chemischen und thermischen Nichtgleichgewichtseffekten*, Dissertation, Universität Göttingen, 1996.

[5] D. Hempel, *Isotropic refinement and recoarsening in two dimensions*, Numerical Algorithms, 13 (1996), pp. 33–43.

[6] A. Meister, *Zur zeitgenauen numerischen Simulation reibungsbahafteter, kompressibler, turbulenter Strömungsfelder mit einer impliziten Finite-Volumen-Methode vom Box-Typ*, Dissertation, T.H. Darmstadt, FB Mathematik, 1996.

[7] H. R. Schwarz, *Die Methode der Finiten Elemente*, B.G. Teubner, 1980.

[8] T. Skalicky and H.-G. Roos, *Anisotropic mesh refinement for problems with internal and boundary layers*, tech. rep., 1997.

[9] T. Sonar, *Strong and weak norm refinement indicators based on the finite element residual for compressible flow computation*, Impact of Computing in Science and Engineering, 5 (1993), pp. 111–127.

[10] T. Sonar, *ENO - Verfahren*, B.G. Teubner, 1997.

[11] T. Sonar, V. Hannemann, and D. Hempel, *Dynamic adaptivity and residual control in unsteady compressible flow computation*, Mathematical and Computer Modelling, 20 (1994), pp. 201–213.

[12] T. Sonar and E. Süli, *A dual graph-norm refinement indicator for finite volume approximations of the euler equations*, Numerische Mathematik, 78 (1998), pp. 619–658.

[13] T. Sonar and G. Warnecke, *On finite difference error indication for adaptive approximations of conservation laws (revised 2nd printing)*, Hamburger Beiträge zur Angewandten Mathematik, (1997).

[14] T. Sonar and G. Warnecke, *On a posteriori error indication based on finite differences in triangular grids*, ZAMM, Z. angew. Math. Mech. 78 (1998), pp. 47–48.

[15] E. Süli, *A posteriori error analysis and adaptivity for finite element approximations of hyperbolic problems.*

[16] J. Weickert, *Anisotropic Diffusion in Image Processing*, B.G. Teubner, 1998.