Bericht des Instituts für Aerodynamik und Strömungstechnik

Report of the Institute of Aerodynamics and Flow Technology

IB 124-2007/1

Using the Adjoint Formulation for Target Functional-Based Mesh Adaption

Daniel B. Vollmer

Herausgeber:
Deutsches Zentrum für Luft- und Raumfahrt e.V.
Institut für Aerodynamik und Strömungstechnik
Lilienthalplatz 7, 38108 Braunschweig

ISSN 1614-7790

Stufe der Zugänglichkeit: 1
Braunschweig, im März 2007

Institutsdirektor:
Prof. Dr.-Ing. habil. C.-C. Rossow

Verfasser:
Daniel B. Vollmer

Abteilung: Numerische Verfahren
Abteilungsleiter:
Prof. Dr. N. Kroll

Der Bericht enthält:
22 Seiten
9 Bilder
2 Tabellen
23 Literaturstellen

Abstract

In this paper we will present a goal-oriented mesh adaption for the DLR TAU code, together with first applications of the procedure to simple inviscid, 2D flow around airfoils. This improvement to the existing mesh adaption already available in the TAU code allows us to produce improved computational meshes that provide more accurate predictions for selected functionals of the flow solution such as drag or lift — in fact, for any for which the necessary adjoint solution can be computed.
After a short introduction to the problem at hand, we show the mathematical foundation of the method and its derivation. Next is a section on the implementation of the procedure into the DLR TAU code, followed by results of mesh refinement studies done with the new method and comparisons to global and the default feature (gradient) based adaption strategies. Lastly, we present our conclusions and discuss further work to improve the method and its implementation.

Contents

1 Introduction

Today's computational fluid dynamics (CFD) solvers are used to solve problems of ever increasing size and complexity, with many industrial partners relying more and more on the predictions of their computations and less on actual testing until later development stages are reached.

A big improvement for complex configurations — although not the "panacea" it may once have been touted as — was the introduction of unstructured methods, which allow the cumbersome and time-consuming process of mesh generation for such computations to be largely automated. This was thought to reduce the reliance on experienced users for setting up such complicated computations. Unfortunately, the quality of the computational mesh is still directly related to the accuracy of the result[1]. The quality of the mesh is in turn often dependent on the experience of the user with similar flow configurations as it is usually not clear *a priori* where dominant flow features will occur, and even whether they will have an impact on the outcome of the computation.

To remedy this and produce meshes that resolve each of the flow phenomena of interest, local mesh adaption was introduced (for examples in the DLR Tau code see [1]). This procedure takes an existing computational mesh and a flow solution thereon[2] to produce a new mesh that will in some sense yield a better result for the observed flow topology, usually by inserting additional points in "regions of interest" and sometimes removing points where they are not needed or moving existing points to improve element quality.

In order to isolate these regions, a so-called *adaption indicator* or *sensor* is computed for each point in the mesh to obtain a measure of the local mesh quality[3]. This is a non-trivial process as mesh quality is not a purely local phenomenon because it depends on how the flow solver itself operates — for example the size of the stencil of the scheme, the exact nature of its gradient computations, or the flux functions used all influence how accurate a solver can be on a particular mesh. For this reason, almost all mesh adaption indicators resort to computing gradients of the solution variables with the notion being that mesh regions where large changes in the solution occur (e.g. shocks / discontinuities) have a strong impact on the quality of the result. In a sense, this is certainly true as by Godunov's theorem such regions can only exhibit 1st order

[1] Or in some non-trivial cases, whether a result could be obtained at all as a low quality mesh has a high impact on the robustness of the solver.

[2] As opposed to more dynamic methods that modify the mesh *during* the solver iteration itself and not as a separate post-process, for example [23].

[3] The quality of a mesh is not only dependent on the density of points, but also on the elements that are constructed out of them, but we presuppose that a good mesh adaption will produce adequate elements for the solver.

accuracy and as such, increased spatial resolution will improve the resolution of those discontinuities.

Unfortunately, the existence of such strong shocks can lead to a problem where most — if not all — of the added points during mesh adaption are spent over-resolving shocks all the while neglecting either weaker shocks or under-resolving smooth solution areas which nevertheless can have a larger impact on the accuracy of the computation, depending on how the "accuracy" is measured. This can result in solutions whose inaccuracy is *amplified* by the repeated application of local mesh refinement based on such a gradient-based indicator — a clear contradiction to the common opinion that using more points will always give a better solution.

In fact, most of the time it is not necessary for a solution to contain equally accurate results over the whole domain (which is also one of the reasons why constructing an accurate *and* efficient mesh is hard) as the main interest in many computations is the prediction of key performance functionals, such as the drag (C_D) or lift (C_L) coefficient of transport aircraft configurations for example. But not each point of the grid contributes equally to the evaluation of these functionals, and that can be used to our advantage. By using the so-called adjoint equation to obtain information about where discretization errors in the solution have an impact on the actual functional of interest, one can reliably identify regions of the mesh for adaption for a given target functional. Obtaining an estimate of the discretization error itself is a challenge of its own because if it *were* known we would know the exact solution to the problem.

Another advantage of the described approach to mesh adaption based on a target functional is that it — by its very construction — allows for an estimate of the remaining error in the functional for the current solution. This is very useful for measuring the reliability or conversely the uncertainty of a computation (as investigated among others in the NODESIM-CFD [17] or MUNA [15] projects), and thus can used to terminate a solver / mesh adaption cycle once the desired accuracy in the target functional is reached.

In the remainder of this paper, we show the mathematical foundation of the method and its derivation, followed by a section on the implementation of the procedure into the DLR TAU code. Then we show results of mesh refinement studies done with the new method and comparisons to global and the default feature (gradient) based adaption strategies. Lastly, we present our conclusions and discuss further work to improve the method and its implementation.

2 Mathematical Background

The approach to goal-based adaption for finite volume methods described herein stems from the error *correction* approach to integral functionals with the dual (or adjoint) problem (which is already very popular in the context of gradient-based optimization, see for example [6]) as shown by Pierce and Giles [8] [19]. While they mention uses of the correction term as an adaptation indicator in passing, their focus lies in achieving higher-order convergence as measured by the error in the functional of choice.

Venditti and Darmofal [21] [22] then develop a correction *and* mesh refinement strategy for unstructured finite volume methods using a discrete adjoint solution. Their correction term is analogous to Pierce and Giles' adjoint correction term [20], but introduced by a slightly different argument. It is derived via a Taylor series expansion on a coarse mesh to an embedded fine mesh where then terms are approximated to avoid computations on the fine mesh altogether — except for a residual evaluation.

We will give a compact derivation of the procedure as described by Venditti and Darmofal [22] and as such will mostly adhere to their naming conventions, these being

- U – (Converged) flow solution of the Euler or Reynolds-averaged Navier-Stokes equations.
- R – Residual operator, i.e. to which degree the argument fails to solve the discretized equations.
- I – Target functional (of the solution U), such as aerodynamic drag C_D or lift C_L.
- Ψ – (Converged) solution of the dual (or adjoint) problem, with boundary conditions as imposed by the choice of target functional I.
- H, h – Used as super- and subscripts and refer to meshes on which particular operations have been carried out. H is a "coarse" mesh and h is an embedded "fine" mesh.
- L, J – Interpolation or prolongation operators for moving values between meshes, often used implicitly when both mesh-subscripts are used (e.g. $X_h^H = L_h^H X_H$).

We depart from a Taylor-series expansion of the functional I on the fine grid based on the coarse-grid solution:

$$I_h(U_h) = I_h(U_h^H) + \left.\frac{\partial I_h}{\partial U_h}\right|_{U_h^H} (U_h - U_h^H) + \cdots \tag{2.1}$$

Similarly, we can expand the residual on the fine mesh

$$R_h(U_h) = R_h(U_h^H) + \left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H} (U_h - U_h^H) + \cdots$$

If we now truncate the above series and isolate terms, we obtain

$$\left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right)^{-1} (R_h(U_h) - R_h(U_h^H)) \approx (U_h - U_h^H).$$

By its very definition and the postulate the we deal with converged solutions $R_h(U_h) = 0$ for a given U_h[1]. If we use this fact, then the previous expansion of the residual reduces to

$$-\left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right)^{-1} R_h(U_h^H) \approx (U_h - U_h^H). \tag{2.2}$$

If we now substitute Equation 2.2 into Equation 2.1 one obtains

$$I_h(U_h) \approx I_h(U_h^H) - \left.\frac{\partial I_h}{\partial U_h}\right|_{U_h^H} \left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right)^{-1} R_h(U_h^H).$$

At this point, we introduce the adjoint variable Ψ with a definition as follows

$$\left(\Psi_h|_{U_h^H}\right)^T = \left.\frac{\partial I_h}{\partial U_h}\right|_{U_h^H} \left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right)^{-1} \tag{2.3}$$

$$\Leftrightarrow \left(\Psi_h|_{U_h^H}\right)^T \left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right) = \left.\frac{\partial I_h}{\partial U_h}\right|_{U_h^H} \tag{2.4}$$

$$\Leftrightarrow \left(\left.\frac{\partial R_h}{\partial U_h}\right|_{U_h^H}\right)^T \Psi_h|_{U_h^H} = \left(\left.\frac{\partial I_h}{\partial U_h}\right|_{U_h^H}\right)^T \tag{2.5}$$

based U_h^H, i.e. the primal coarse grid solution interpolated to the fine mesh, with the adjoint computation taking place on the *fine* level. This definition of the dual problem is analogous to the one used in the optimization context (see for example Gauger [6]) — except for the particular form of the primal solution (usually U instead of the U_h^H used here) — and can be solved with exactly the same solver and boundary conditions that have been developed in the optimization context.

So as long as Equation 2.5 holds true, we have

$$I_h(U_h) \approx I_h(U_h^H) - \left(\Psi_h|_{U_h^H}\right)^T R_h(U_h^H), \tag{2.6}$$

which is an approximation of the sought-after functional evaluation from a fine-mesh solution based on a coarse-mesh solution *interpolated to the fine grid* minus a correction

[1] As an aside, for a given converged coarse-grid solution U_H we have $R_H(U_H) = 0$ but $R_h(L_h^H U_H) \neq 0$. Instead of using a finer mesh to create a residual of higher accuracy, one could also use a consistent residual operator with a higher order of accuracy than the one used to solve for U to gage this discretization error.

term. As written above, this correction term incorporates a solution for Ψ on the fine mesh, which we would like to avoid as the solution of the adjoint problem is very roughly about half as expensive as the solution U of the original problem so we could just have solved U on the fine mesh to start with. Thus we used following *ansatz* of replacing the fine-mesh adjoint with an interpolated adjoint from the coarse grid:

$$\Psi_h|_{U_h^H} \approx L_h^H \, \Psi_H|_{U_H} = \Psi_h^H, \tag{2.7}$$

where L_h^H is a prolongation operator from the coarse grid H to the fine grid h. This is a sensible approximation because the adjoint solution is smooth where the primal solution exhibits discontinuities (Giles [7]). The smooth regions in the adjoint solution are well-predicted by the interpolation operator, and these regions are also the ones that coincide with large residuals (e.g. across discontinuities in the primal solution), which in turn make a significant contribution to the correction term. When we now apply this approximation to Equation 2.6 and write U_h^H as $J_h^H U_H$ (with J_h^H being a similar prolongation operator), we arrive at

$$I_h(U_h) \approx I_h^H(J_h^H U_H) - (L_h^H \Psi_H)^T R_h(I_h^H U_h) \tag{2.8}$$

for which Ψ_H has to satisfy the traditional adjoint equation of

$$\left(\frac{\partial R_H}{\partial U_H}\right)^T \Psi_H = \left(\frac{\partial I_H}{\partial U_H}\right)^T. \tag{2.9}$$

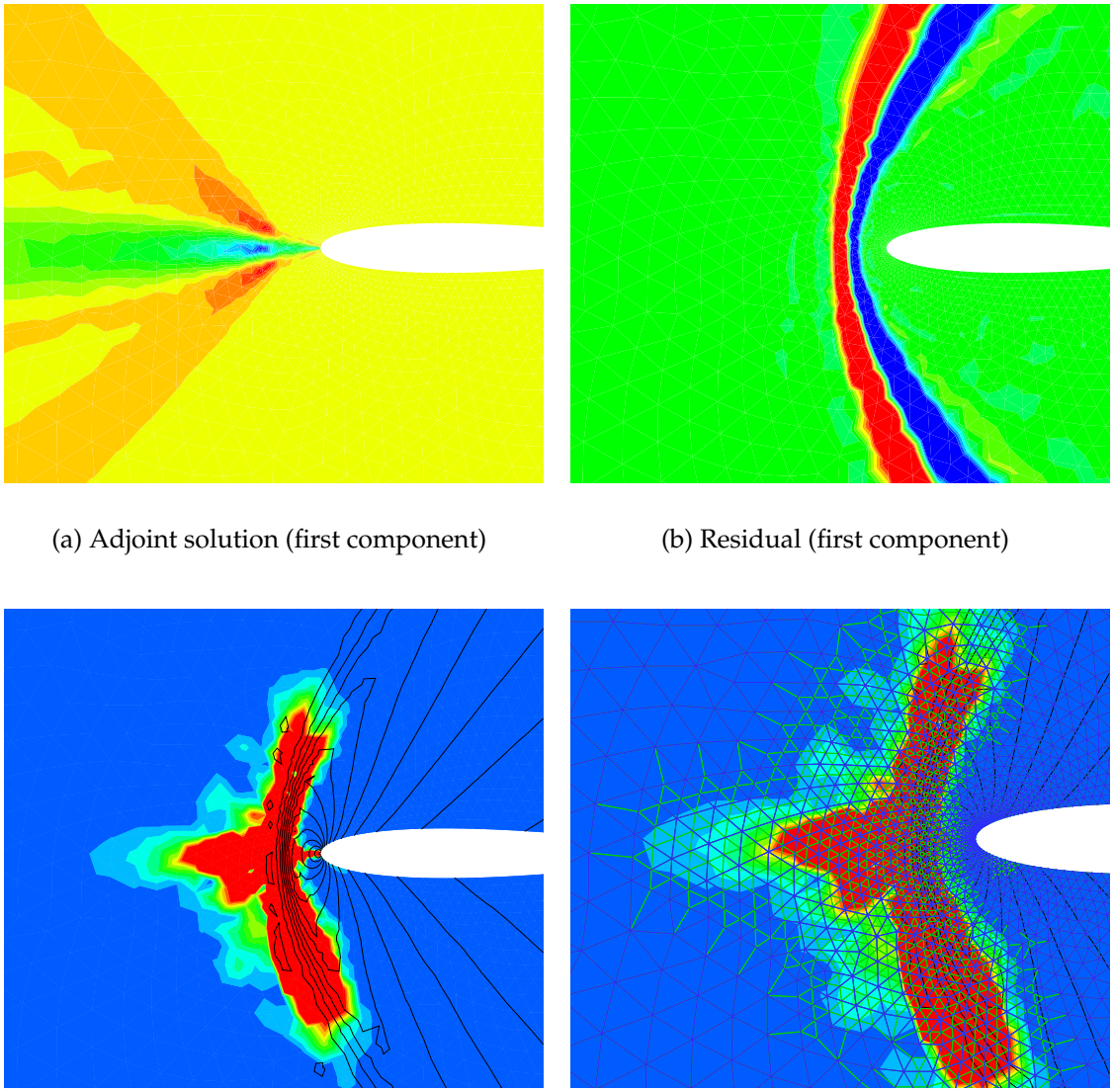
The correction term in Equation 2.8 forms the basis of our mesh adaption indicator. It is made up of many individual contributions, one from each of the points in the computational mesh (of the form shown in Equation 2.10) and as such we can use these point-wise contributions to identify regions in the grid where refinement would have a non-negligible impact on the target functional, while the sum over all these individual terms for all mesh points yields the correction term itself.

Another way of looking at the correction term in a more colloquial manner concentrates on the two components that make up each point-wise contribution. For the solution at a given mesh point i (expressed in primitive variables) this contribution boils down to

$$\Psi|_i \cdot R|_i = \Psi_\rho|_i \, R_\rho|_i + \Psi_u|_i \, R_u|_i + \Psi_v|_i \, R_v|_i + \cdots \tag{2.10}$$

Individually, the residual R obtained from the fine mesh can be seen as a measure of the discretization error on the original mesh (i.e. where a fine mesh would result in a large update to the solution) whereas the adjoint solution Ψ relates that to the sensitivity of the target functional I to changes in the solution. This viewpoint led to the name of "dual-weighted residual (DWR)" method in the Finite Element (e.g. Becker and Rannacher [2]) and Discontinuous Galerkin (e.g. Hartmann [9] and Houston [10] [11]) community, where constructing residual operators of varying order is comparatively easy.

Figure 2.1 serves as a demonstrative example illustrating the individual components that make up the adaption indicator (respective to Equation 2.10) for a supersonic flow case. The functional of interest for this case is the pressure at the tip of the profile. The adjoint solution clearly shows which parts of the flow field can have an influence



(a) Adjoint solution (first component)

(b) Residual (first component)

(c) Adaption indicator and pressure contours

(d) Adapted mesh overlaid over the sensor

Figure 2.1: An example illustrating the individual components of the adaption sensor (NACA0012 with $M_\infty = 1.5$ and $\alpha = 1.0°$).

on the pressure at the tip; and as expected they are nearly all upstream of it, fanning out as the distance increases. The residual is very large at the strong shock — across which it is also changes sign — and approaches zero everywhere else. Their point-wise inner product is then the adaption indicator, which for this case only refines those parts across the shock (and the region leading up to it) that are in front of the tip, as those are the only points that have an influence on the pressure there.

3 Implementation in the DLR TAU Code

The DLR TAU code [5] is a second order space and time[1] accurate cell-vertex finite-volume code working on hybrid unstructured meshes. The overall process-chain that has been implemented works as follows:

1. We compute a (fully converged) solution of the original configuration. For the purposes of this paper, the central scheme with an implicit LUSGS solver [3] was used, but the procedure is completely independent of the solver itself so any of the existing options in the TAU code can be used with no modification to the adaption (as long as the same solver is used for the residual evaluation).

2. Based on this solution, we obtain the matching discrete adjoint solution as described in [4]. Again, the procedure is not tied a particular adjoint implementation and could equally well use the continuous adjoint implementation that is available in the code for Euler computations.

3. In order to obtain the fine-mesh residual needed, the original mesh is globally refined and the solution from the first step is linearly interpolated onto the new mesh.

4. The (non-zero) residual of the coarse solution interpolated onto the fine mesh is calculated. This is essentially a single iteration of the solver-loop to sum all the fluxes across each of the faces and yields information on where a finer mesh would modify the original flow solution.

5. The volume-weighted residuals are summed from the fine mesh back to the original mesh, as that is where the adaption indicator is needed. This means that the residual contributions from the dual cells that were added for the embedded fine mesh are distributed back to their coarse parent cell (see Figure 3.2). A positive side effect of evaluating the indicator on the coarse mesh is that the adjoint solution does not need to be interpolated to the fine mesh. Also, no residual smoothing as proposed by Müller and Giles [14] was done as the residuals did not exhibit a checkerboard pattern, most likely due the different nature of the residual evaluation.

6. The actual adaption indicator ϵ_i whose magnitude is used to select elements for refinement is calculated point-wise as $\Psi^T \cdot R$, where Ψ is the adjoint solution

[1] Although the time accuracy can also be third order if requested.

vector and R is the residual vector at the point. This is currently performed in an external Python script with a thin layer called `netcdfdic` on top of the NetCDF-interface [16] used by the TAU code. The script performs the sensor evaluation given the primal, adjoint and residual `pval`-files as arguments.

7. Finally, we use TAU's mesh adaption facility [1] to refine the edges where either endpoints' $|\epsilon_i| \geq \sigma \frac{\epsilon_t}{|\epsilon_g|}$ where σ is the standard deviation of the local indicators / errors ϵ_i, ϵ_t is the prescribed uncertainty one wants to achieve in the functional and ϵ_g is the "global error estimate" which is simply $\sum \epsilon_i$; similar to the limits used by Kim and Nakahashi [12] with $a = 1$.

The relevant processes and their inputs, outputs and dependencies are shown in Figure 3.1. A feature-based adaption only needs the left-most components from the diagram, i.e. an initial grid with a flow solution is used with the adaption to produce an adapted grid. While the overhead in terms of added complexity is large, most of it uses existing — and thus well-tested — functionality.

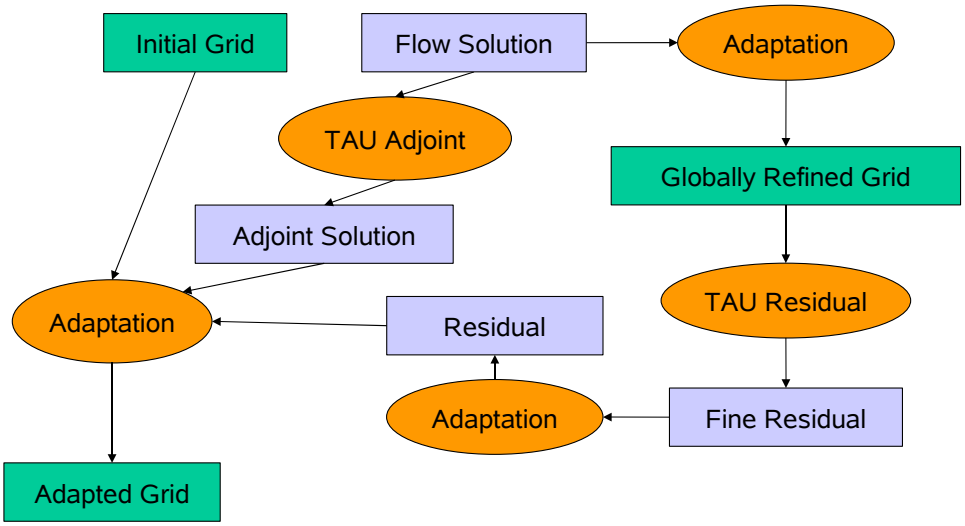
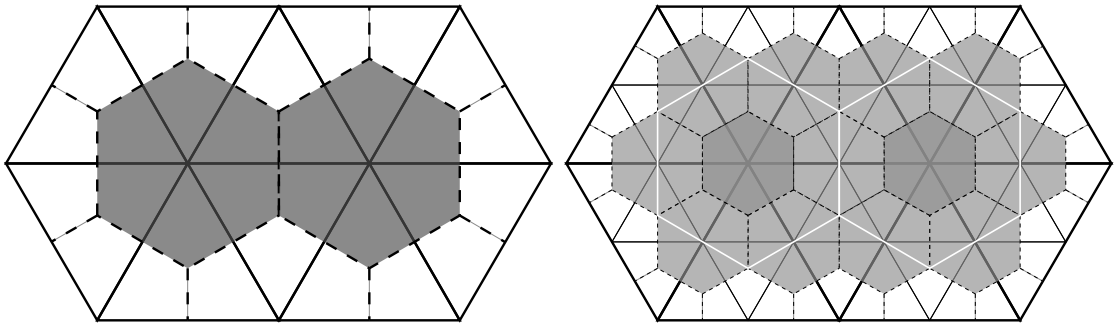


Figure 3.1: Flow chart of steps currently involved in the proposed adaption mechanism. Note that most of the steps are necessary to produce a fine-mesh (and thus non-zero) residual of the coarse grid solution for the estimation of the discretization error.

The global refinement of the mesh that is used to produce the residual (i.e. how much a finer mesh would change each point in the solution — a measure of the discretization error) uses a slightly modified version of the normal TAU adaption module, where all edges are forcibly bisected. Unfortunately, the TAU adaption can not currently refine hexahedral sublayers (which are usually used for resolving the boundary layer in Navier-Stokes computations). This stems from the fact that the TAU adaption employs automatic y^+ adaption to redistribute the points there. Nevertheless, work is under way to remove this restriction. Simple trials on such hybrid Navier-Stokes grids seem to suggest that the sensor computation procedure itself works when one ignores the structured parts of the mesh — although this of course makes the correction term incorrect...

For facilitating the evaluation of the sensor a new data-structure was introduced that specifically tracks edge bisections with parent-child relationships through the refinement and derefinement process. This allows us to use the normal derefinement procedure of the TAU adaption tool to restrict the sensor from the globally refined mesh h to the original mesh H in a mode where for each point that exists in h but not in H we distribute $1/2$ of its value to each of the parent edge's endpoints. That this holds true for volume-weighted values can be seen in Figure 3.2 as all of the dual cell's volume is accounted for and redistributed correctly back to the original mesh points.



(a) Original Mesh (b) Globally Refined Mesh

Figure 3.2: The left-hand side shows two dual cells cells and the accompanying primal mesh. On the right-hand side, the primal mesh has been globally refined, and one can see that by distributing $1/2$ of each newly added vertex' dual volume to the endpoints of the parent edge that it bisects, we account completely for the original volume as shown in white.

4 Results

In this section we will present and discuss results obtained with the aforementioned methodology. First, the block-structured DLR FLOWer code — part of the MEGAFLOW initiative [13] — was used to verify the integrity of the correction term and the various approximations contained therein, as its continuous adjoint implementation was already stabilized at the time this work was begun.

After the adjoint mode became available in the TAU solver, the implementation for the target functional-based sensor was migrated and modified to work in an unstructured mesh context, which is much more suited to automatic mesh adaption by construction.

4.1 FLOWer – Structured Meshes

We used the FLOWer code to conduct global refinement studies to verify the correctness and the applicability of the functional correction in our infrastructure. As a first test case the inviscid subsonic flow around the NACA0012 airfoil at $M_\infty = 0.63$ and $\alpha = 2.0°$ was considered. In this case the theoretical drag coefficient C_D is zero since the flow is inviscid and has no shocks. For numerical solutions this exact value cannot be obtained in practice due to the discretization error and the numerical dissipation introduced by the numerical method. However, the finer the mesh is, the closer the approximate value of C_D should be to zero.

Table 4.1 shows the number of cells for 4 different mesh levels for this NACA0012 case. These were obtained by global refinement. Here, one increase in level implies the application of one global mesh refinement. With this set of refined meshes a mesh consistency study for the adjoint based error estimator was performed. Results of this investigation are shown in Table 4.2 and Figure 4.1.

Level 1	768	x	256	=	196,608	cells
Level 2	384	x	128	=	49,152	cells
Level 3	192	x	64	=	12,288	cells
Level 4	96	x	32	=	3,072	cells

Table 4.1: Cell numbers for different mesh levels (NACA 0012, $M_\infty = 0.63$ and $\alpha = 2.0°$).

It can be seen in Table 4.2 (and correspondingly Figure 4.1) that the finer the mesh level H is, the better the correction term $\Psi_h^H \cdot R_h(U_h^H)$ agrees with the difference $C_D(U_h^H) - C_D(U_h)$ that it approximates. This lets us conclude that this is valid approximation

Level	4 – 3	3 – 2	2 – 1
$C_D(U_h^H)$	0.0036637	0.0009747	0.0004667
$C_D(U_h)$	0.0009919	0.0004889	0.0004019
$C_D(U_h^H) - C_D(U_h)$	0.0026718	0.0004858	0.0000648
$\Psi_h\|_{U_h^H} \cdot R_h(U_h^H)$	0.0018453	0.0004051	0.0000788
$\Psi_h^H \cdot R_h(U_h^H)$	0.0009619	0.0002830	0.0000505

Table 4.2: Drag coefficients and adjoint-based error estimates / correction terms with varying degrees of approximation for different mesh levels. Level 4 – 3 means that H refers to mesh level 4 and h to mesh level 3 and so on (NACA 0012, $M_\infty = 0.63$ and $\alpha = 2.0°$).

because it converges towards the actual difference between $C_D(U_h^H)$ and $C_D(U_h)$ as the meshes become progressively finer.

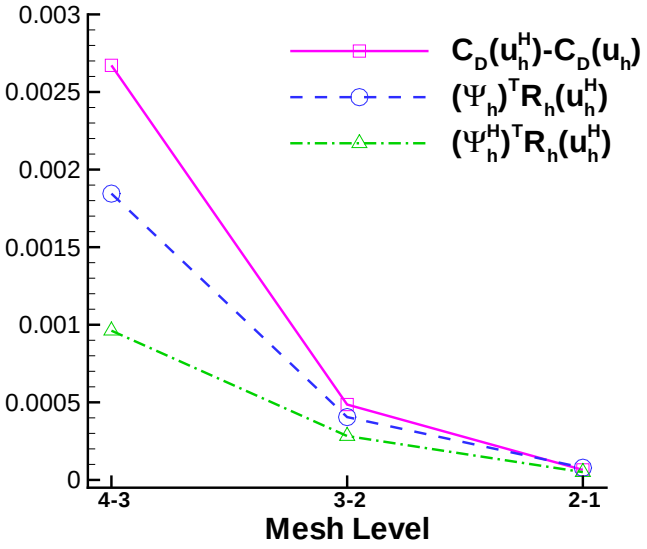
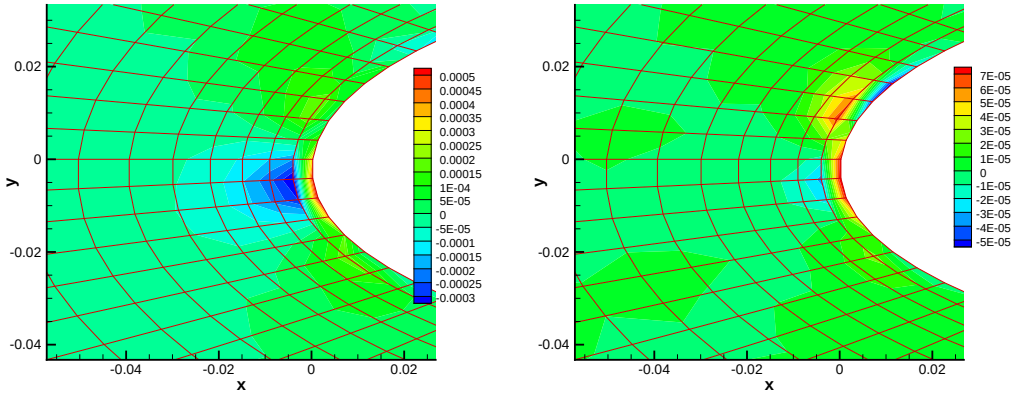


Figure 4.1: Plot of two variations of the functional correction term, one using an adjoint solution computed on the coarse mesh (based on U_H) with the other computed on the fine mesh (based on U_h^H); compared to the value both terms are approximating, namely $C_D(U_h^H) - C_D(U_h)$.

Furthermore, when looking at the individual contributions to the correction term from each point (for usage as an adaption indicator) in Figure 4.2 and the difference between computing the respective adjoint solution on the coarse grid (i.e. $L_h^H \; \Psi_H|_{U_H}$) and the fine grid (i.e. $\Psi_h|_{U_h^H}$), it is evident that the error is an order of magnitude smaller than the contributions to the correction term themselves, which again validates the simplification of using the coarse mesh adjoint solution, although this of course makes the correction less accurate — but not invalid — as seen in Figure 4.1.



(a) Pointwise correction term (coarse adjoint)

(b) Error in approximation

Figure 4.2: Magnified view of the nose-region of the NACA 0012 profile showing the point-wise values making up the correction term (left) as well as the error incurred by computing the adjoint solution on the coarse mesh (right). Note the difference in magnitude.

4.2 TAU – Unstructured Meshes

We have used the above-mentioned process for grid refinement studies on an Euler NACA0012 airfoil in 2D under differing flow conditions. Comparisons are made with global mesh refinement, which is also used to estimate the "exact" value of the functional via Richardson extrapolation, as well as the default feature-based adaption already available in TAU. The data-points for global refinement and the default feature-based adaptation were taken from an earlier mesh refinement study conducted by R. Dwight.

The feature-based adaptation was set up to introduce approximately 30% more points into the mesh at each adaptation step, whereas the goal-oriented procedure flagged as many elements of the mesh for refinement as it deemed necessary to achieve the prescribed error tolerance. This amount was usually well in excess of that produced by the feature-based adaptation, but as the goal-based adaption is more costly in terms of computational effort (as the adjoint solution has to be computed as well), this need not be a disadvantage as long as the mesh that is produced yields more accurate results.

4.2.1 Subsonic Flow

The first test-case was the symmetric flow around the NACA0012 profile at $M_\infty = 0.5$ and an angle of attack of $\alpha = 0.0°$ and it used drag (C_D) as the target functional for the adjoint computation (and thus the functional correction / mesh adaption procedure). Similarly to the FLOWer test-case, the exact value of the drag is 0, due to completely subsonic inviscid flow. The result of the mesh adaption study can be seen in Figure 4.3,

where the goal-based approach was used with three different values of ϵ_t.

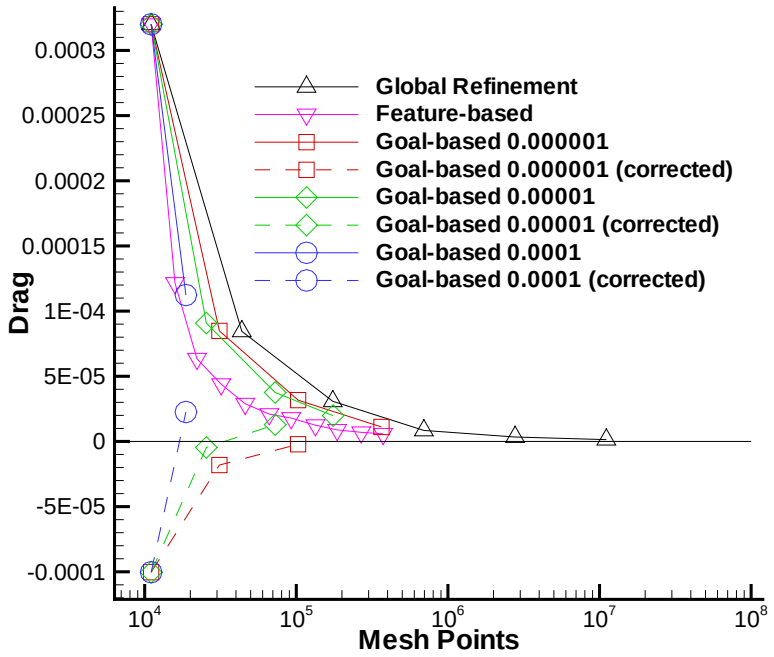


Figure 4.3: Convergence history for repeated solver / adaption cycles for different refinement strategies, including the correction term as dashed line (NACA 0012, $M_\infty = 0.5$ and $\alpha = 0.0°$).

One can see that for this case essentially any refinement will reduce the error in the functional compared to the exact value. As expected, the different error thresholds for the goal-oriented refinement strategy modify the number of elements (or more accurately edges — as the TAU adaption tool is based on edge splitting) introduced during each adaption cycle, where a lower error threshold (i.e. a higher required accuracy) results in more refinement during a single adaption step.

The goal-oriented strategy does not improve on the already good results the feature-based adaption provides, in fact it even performs very slightly worse; although one has to remember that all of this take place in the space of a single drag count. This state of affairs can be improved significantly when we use the correction term as introduced beforehand, which comes at no extra cost provided we are already computing the goal-oriented adaption indicator. Using this correction factor, the new adaption strategy provides consistently better results than the default gradient sensor. For the largest given error threshold of $\epsilon_t = 0.0001$, a single adaption cycle is enough to achieve a result that is accurate enough — which the procedure realizes after evaluating and applying the correction term on the second mesh — and self-terminates.

The grid in Figure 4.4 is the result after three mesh adaptions given the very low error tolerance of $\epsilon_t = 0.000001$. One can see that the whole region around the profile has been almost uniformly refined, suggesting that the error is very evenly distributed over the whole computational domain.

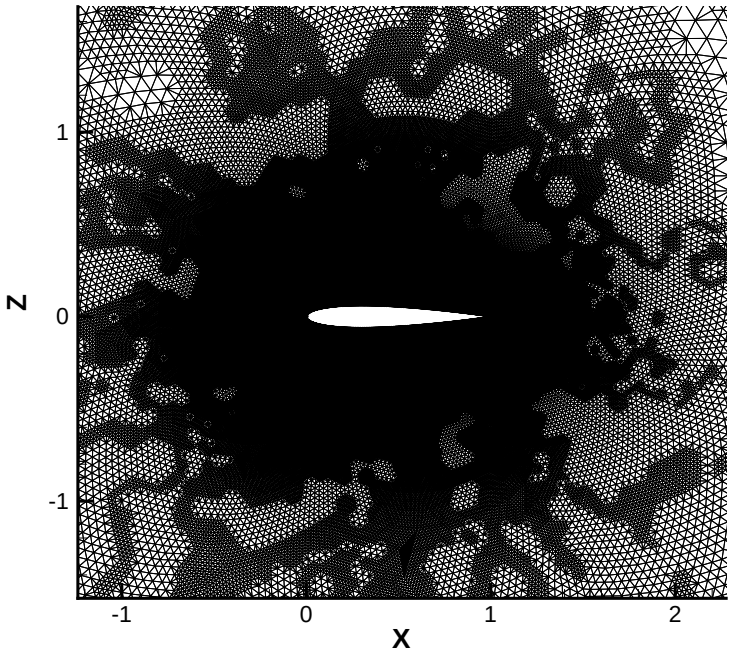


Figure 4.4: Part of the grid near the profile after the third adaption cycle for $\epsilon_t = 0.000001$ (NACA 0012, $M_\infty = 0.5$ and $\alpha = 0.0°$).

4.2.2 Transsonic Flow

Figure 4.5 shows a series of mesh refinements for the NACA0012 airfoil at $M_\infty = 0.85$ and $\alpha = 2.0°$. The horizontal black line represents the "correct" value of the lift coefficient as obtained from the globally refined meshes by Richardson extrapolation. This transonic case exhibits two shocks, a stronger one on the upper side and a weaker one on the lower side. The goal-oriented adaptation is performed to produce a mesh that increases accuracy of the lift coefficient C_L of the profile for different error tolerances ϵ_t. This tolerance essentially increases or decreases the sensitivity of the adaption process and thus somewhat dictates the amount of the new points introduced. If one wants to achieve a very accurate end-result, even regions with a relatively small adaption indicator magnitude will be refined in order to meet the required tolerance whereas as these regions will be left as they are when only lower accuracy is desired.

In this particular case, the default feature-based adaptation produces rather erratic results and even seems to diverge from the "exact" value of C_L. We suspect this is the case due to the large gradients across the upper shock, which make the grid adaption spend every last bit of its allocated point budget there and thus neglecting other flow features such as the weaker shock and the trailing edge that also have an impact on the lift.

The goal-oriented adaptation gives very good results for all tolerances (although none of them has achieved a mesh on which $\epsilon_g < \epsilon_t$ yet, but that may be due to the rather small tolerances chosen). After three adaptation steps all meshes provide a lift coeffi-

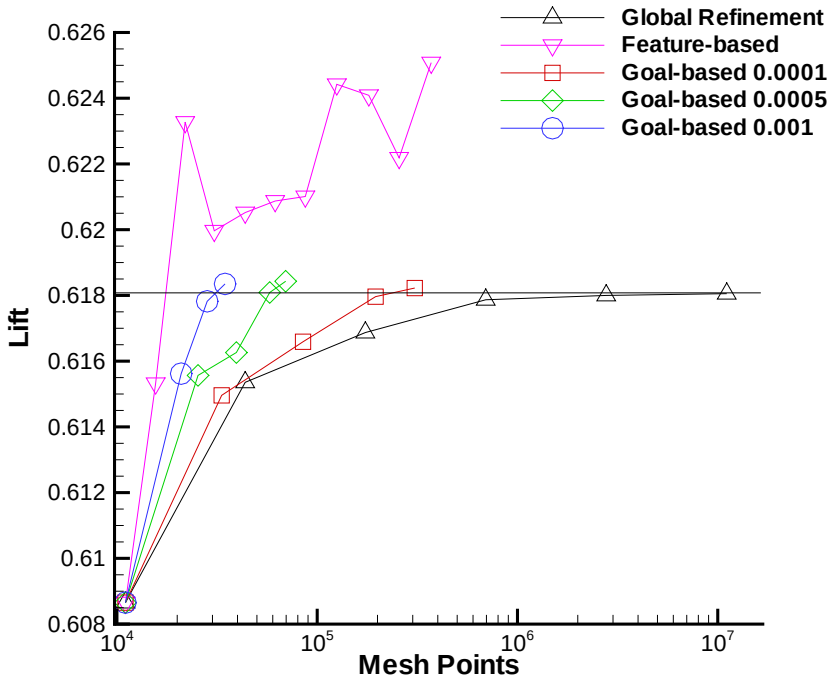


Figure 4.5: Lift convergence for different adaption indicators (NACA0012 with $M_\infty = 0.85$ and $\alpha = 2.0°$).

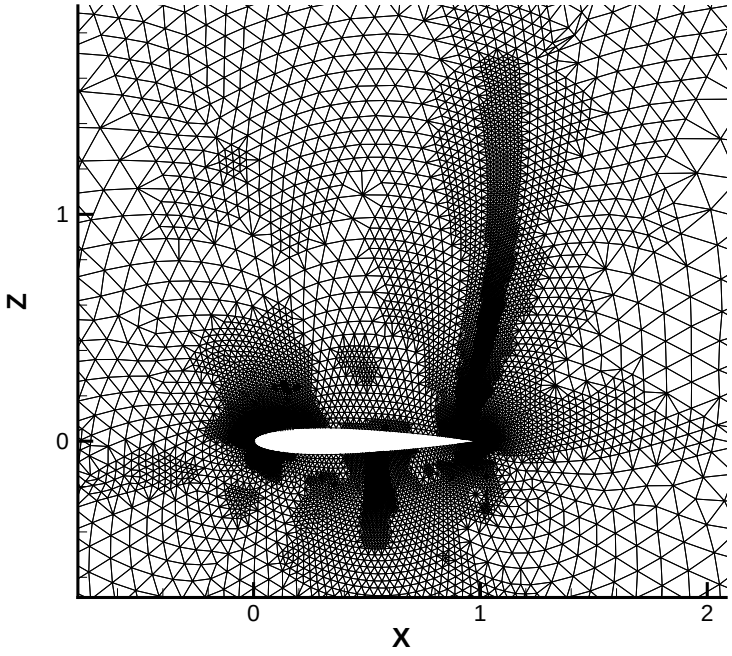


Figure 4.6: Grid after the three goal-oriented adaptions with $\epsilon_t = 0.001$ (NACA0012 with $M_\infty = 0.85$ and $\alpha = 2.0°$).

cient that is within 0.0003 of the exact value, whereas as the overall best result[1] from 10 feature-based adaptations is off by 0.002 and getting worse. Somewhat surprisingly, the mesh with the lowest requested accuracy yields the best results — at least for the first two adaptation cycles. This is probably due to the fact that the adaption with a higher requested accuracy also refines many areas which only become relevant in the later adaption cycles from a functional accuracy standpoint, while the adaption for the lower requested accuracy is able to ignore these completely because it never needs to resolve these smaller error contributions to reach its target accuracy.

The final mesh after the third adaptation for $\epsilon_t = 0.001$ can be seen in Figure 4.6. Easily visible are the two well-resolved shocks but the whole nose region as well as the trailing edge are also refined to provide a more accurate lift coefficient.

[1] Which is usually not known as the exact value is not available...

5 Conclusion

We have shown the reasoning behind and implementation details of the goal-oriented mesh adaption approach in the unstructured DLR TAU solver, as well as a consistency study using the structured FLOWer code. Encouraging results have been presented for inviscid 2D flows on triangular meshes around the NACA 0012 profile for a variety of flow conditions.

For the very simple subsonic case the additional expenditure for evaluating the adjoint solution and computing the fine-mesh residual does offer enough of an in improvement (taking the more accurate functional prediction afforded by the correction term into account) compared to the normal gradient-based approach. That said, this is a case where literally any refinement will improve the result as measured by the drag's proximity to 0.

The advantages of the approach become evident for more complex flows such as the transonic test-case presented, where the feature-based adaption process diverges from the correct lift value. This is a very real problem for users, who expect mesh adaption to *improve* their results and not make them worse, especially as the real or correct value is not known beforehand so that any prediction is as good or as bad as any other — but with more weight given to any result produced on a mesh with more points. This fallacy is clearly shown here as well as how the adaption based on a target functional alleviates the problem.

The interpretation of the magnitude of the correction term as a measure of the remaining uncertainty in a CFD solution gives users an idea of the reliability of the results they have produced and whether further adaption cycles are warranted or whether the obtained solution is good enough. This becomes increasingly important as the faster development cycles in the aerospace industry require reliable results from simulations while deferring actual testing to the nearly finished design.

5.1 Further Work

Nevertheless, further improvements and validation of the method are necessary. During analysis of the results for the transonic test-case it became evident that the correction term did not actually improve the estimate of the lift. Although the corrected values also seem converge to the assumed exact lift obtained via Richardson extrapolation from the globally refined meshes, on each individual grid the uncorrected aerodynamic coefficient was slightly closer to the "exact" value than the "improved" estimate.

We suspect this may be due to the way we currently evaluate the correction term. In

order to carry out most operations on the coarse grid and to reduce the number of interpolations carried out, we compute the term as $\Psi_H(L_H^h R_h(U_h^H))$ instead of the more involving $L_H^h((L_h^H \Psi_H) R_h(U_h^H))$. Work on implementing the alternative is currently ongoing to eliminate this potential source of inaccuracy.

Then we will verify the goal-oriented approach on larger 3D computations, as well as Navier-Stokes problems — as soon as the TAU adaption module is able to refine hexahedral layers in hybrid meshes so that we can obtain the higher order residual $R_H(U_h^H)$. Progress in this regard is being made [18] but a fully functioning version is not yet available.

Another focus is the selection of proper limits for which elements to refine in each step to ensure a prescribed accuracy. Multiple adaptations based on the same evaluation of the indicator (where areas with large ϵ_i are refined twice for example) can also help to improve the method's cost per accuracy metric.

A further area for improvement is the complex process needed for evaluating the sensor itself. We intend to look at ways of simplifying this overhead; the best alternative would probably be a way of obtaining a higher order residual as a measure of discretization error *on the coarse mesh* and thus eliminating the need for a global refinement step if methods can be found to accurately evaluate the correction term and mesh adaption indicator solely on the original grid.

Acknowledgements

I would like to express thanks to Richard Dwight for providing the data-points for the globally refined meshes and the results for the feature-based TAU adaption. He also provided assistance in using the discrete adjoint mode in TAU. Markus Widhalm explained the usage of the residual computation and the continuous adjoint solver for which I am grateful. During the preliminary phases of this work, Nicolas Gauger gave helpful suggestions on the inner workings of the FLOWer adjoint mode.

More thanks are due to Airbus Germany for supporting this research financially.

Bibliography

[1] T. Alrutz. Erzeugung von unstrukturierten Netzen und deren Verfeinerung anhand des Adaptionsmoduls des DLR-Tau-Codes. *Diplomarbeit* (2002).

[2] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Institut für Angewandte Mathematik* Universität Heidelberg (2001).

[3] R. P. Dwight. Time-Accurate Navier-Stokes Calculations with Approximately Factored Implicit Schemes. *Proceedings of the ICCFD3* Conference (Springer) Toronto (2004).

[4] R. P. Dwight and J. Brezillon Effect of Approximations of the Discrete Adjoint on Gradient-Based Optimization. *American Institute of Aeronautics and Astronautics* Journal **44(12)**, *pp.* 3022-3071 (December 2006).

[5] M. Galle, T. Gerhold and J. Evans. Technical Documentation of the DLR TAU-Code. *DLR* IB **233-97/A43** (1997).

[6] N. R. Gauger. Das Adjungiertenverfahren in der aerodynamischen Formoptimierung. *DLR* Report **2003-05** ISSN 1434-8454 (2003).

[7] M. B. Giles and N. A. Pierce. On the properties of solutions of the adjoint Euler equations. *Numerical Methods for Fluid Dynamics VI* ICFD, June 1998.

[8] M. B. Giles and N. A. Pierce. Improved lift and drag estimates using adjoint Euler equations. *American Institute of Aeronautics and Astronautics* Paper **99-3293** (1999).

[9] R. Hartmann. Adaptive Finite Element Methods for the Compressible Euler Equations. *PhD Thesis*, Universität Heidelberg (July 2002).

[10] R. Hartmann and P. Houston. Adaptive Discontinuous Galerkin Finite Element Methods for Nonlinear Hyperbolic Conservation Laws. *SIAM Journal on Scientific Computing* **24-3**, *pp.* 979-1004 (2001).

[11] R. Hartmann and P. Houston. Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations. *Journal of Computational Physics* **183-2**, *pp.* 508-532 (2002).

[12] H.-J. Kim and K. Nakahashi. Output-Based Error Estimation and Adaptive Mesh Refinement Using Viscous Adjoint Method. *American Institute of Aeronautics and*

Astronautics 44th Aerospace Sciences Meeting and Exhibit, Reno, NV **2006-1395** (2006).

[13] N. Kroll, C.-C. Rossow, K. Becker and F. Thiele. The MEGAFLOW project. *Aerospace Science and Technology* **4**, *pp.* 223-237 (2000).

[14] J.-D. Müller and M. B. Giles. Solution Adaptive Mesh Refinement Using Adjoint Error Analysis. *American Institute of Aeronautics and Astronautics* Paper **2001-2550** (2001).

[15] The MUNA project — Management und Minimierung von Unsicherheiten in der Numerischen Aerodynamik. `http://www.dlr.de/as/muna`

[16] NetCDF software library — network Common Data Form. `http://www.unidata.ucar.edu/software/netcdf`

[17] The NODESIM-CFD project — Non-Deterministic Simulation for CFD-based Design Methodologies. `http://www.nodesim.eu`

[18] M. Orlt. Hexahedra refinement and derefinement of TAU-Adaption — status and outlook. *TAU Grid Adaption Workshop* Presentation, June 2006.

[19] N. A. Pierce and M. B. Giles. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review* **42-2**, *pp.* 247-264 (2000).

[20] N. A. Pierce and M. B. Giles. Adjoint and defect error bounding and correction for functional estimates. *Journal of Computational Physics* **200-2**, *pp.* 769–794 (2004).

[21] D. A. Venditti and D. L. Darmofal. Adjoint Error Estimation and Grid Adaption for Funtional Outputs: Application to Quasi-One-Dimensional Flow. *Journal of Computational Physics* **164**, *pp.* 204–227 (2000).

[22] D. A. Venditti and D. L. Darmofal. Grid Adaption for Funtional Outputs: Application to Two-Dimensional Inviscid Flows. *Journal of Computational Physics* **176**, *pp.* 40–69 (2002).

[23] D. B. Vollmer. Adaptive Mesh Refinement using Subdivision of Unstructured Elements for Conservation Laws. *Master's Thesis*, University of Reading, UK, September 2003.

IB 124-2007/1

Using the Adjoint Formulation for Target Functional-Based Mesh Adaption

Daniel B. Vollmer

Verteiler:

Institut für Aerodynamik und Strömungstechnik, BS	1	Exemplar
Institut für Aerodynamik und Strömungstechnik, GÖ	1	Exemplar
Verfasser ..... je	1	Exemplar
Deutsche Bibliothek Frankfurt am Main	2	Exemplare
Zentralbibliothek	2	Exemplare
Reserve	21	Exemplare
	28	Exemplare