The role of the Jacobian in the adaptive discontinuous Galerkin method for the compressible Euler equations

Ralf Hartmann

Institute of Aerodynamics and Flow Technology, German Aerospace Center (DLR), Lilienthalplatz 7, 38108 Braunschweig, Germany Ralf.Hartmann@dlr.de

Abstract. We provide a full description of the Jacobian to the discontinuous Galerkin discretisation of the compressible Euler equations, one of the key ingredients of the adaptive discontinuous Galerkin methods recently developed in [7, 8]. We demonstrate the use of this Jacobian within an implicit solver for the approximation of the (primal) stationary flow problems as well as in the adjoint (dual) problems that occur in the context of *a posteriori* error estimation and adaptive mesh refinement. In particular, we show that the (stationary) compressible Euler equations can efficiently be solved by the Newton method. Full quadratic Newton convergence is achieved on higher order elements as well as on locally refined meshes.

1 Introduction

In this paper we consider the adaptive discontinuous Galerkin method for the numerical approximation of the compressible Euler equations recently developed in [7] and [8] (for an overview of the development of the discontinuous Galerkin method we refer the reader to [2]).

The purpose of the current paper is to provide a detailed description of the Jacobian, which is the key ingredient of the Newton iteration for solving the compressible Euler equations but also for the construction of adjoint (dual) problems arising in the context of duality-based *a posteriori* error estimation and adaptivity.

In [7] only parts of the Jacobian are described in detail. In particular, there is no description given of the boundary terms and their Jacobian. Furthermore, the Jacobians of the numerical fluxes are approximated by simply neglecting some of the more complicated terms. In the present paper we now describe and assemble all terms and we demonstrate that the additional terms are actually necessary for obtaining an optimal convergence of the Newton it-

eration. Furthermore, we show that convergence of the Newton iteration is optimal also for higher order elements and on locally refined grids.

Finally, we demonstrate that the duality-based *a posteriori* error representation, see [7, 8], gives sharp error estimation of specific target functionals, when evaluated by solving the so-called adjoint (dual) problem numerically. Again, the key ingredient is the Jacobian that governs the adjoint problem.

The outline of this paper is as follows. In the following two subsections we motivate the importance of the Jacobian in the two fields of application already mentioned, namely the Newton iteration and the duality-based *a posteriori* error estimation and adaptivity. Then, in Section 2, we introduce the compressible Euler equations and formulate their discontinuous Galerkin finite element approximation including the treatment of various different types of boundary conditions. In Section 3, we give the Jacobian of the scheme, including the Jacobian of two specific numerical fluxes, and the treatment of the boundary conditions. The performance of the resulting Newton iteration is then studied in Section 4.1 through a series of numerical experiments on an inviscid flow around the NACA0012 airfoil. In Section 4.2 we summarise some recent results of the *a posteriori* error estimation and goal-oriented grid refinement applied to the discontinuous Galerkin discretisations of a wide range of hyperbolic problems. Then we demonstrate the accuracy of the *a posteriori* error estimation on a sequence of adaptively refined meshes on the example problem already considered in Section 4.1. Finally, in Section 5, we summarise the work presented in this paper and give an outlook.

1.1 The Newton iteration

In this section we introduce the Newton iteration for solving the finite element discretisation of the following nonlinear variational problem: find u in X such that

$$\mathcal{N}(u,v) = 0 \quad \forall v \in Y, \tag{1}$$

where $\mathcal{N}(\cdot,\cdot)$ denotes a semi-linear form (nonlinear in its first argument, but linear in its second), and X and Y denote two Hilbert spaces.

In order to discretise (1) we consider finite–dimensional spaces $X_h \subset X$ and $Y_h \subset Y$ that, for the purposes of this paper, may be thought of as finite element spaces consisting of piecewise polynomial functions on a partition of granularity h. The Galerkin approximation u_h of u is then sought in X_h as the solution of the finite–dimensional problem

$$\mathcal{N}(u_h,v_h) = 0 \quad \forall v_h \in Y_h. \tag{2}$$

For simplicity of presentation, we assume that X_h and Y_h are suitably chosen finite element spaces to ensure the existence of a unique solution u_h to (2).

The nonlinear equation (2) may be solved using the Newton method. This nonlinear iteration generates a sequence of iterates $u_h^n \in X_h$ as follows. Given an iterate u_h^n, the update d_h^n of u_h^n to get to the next iterate

$$u_h^{n+1} = u_h^n + \omega^n d_h^n$$

is given by the following problem: find $d_h^n \in X_h$ such that

$$\mathcal{N}'_u[u_h^n](d_h^n, v_h) = R(u_h^n, v_h) \equiv -\mathcal{N}(u_h^n, v_h) \quad \forall v_h \in Y_h. \tag{3}$$

Here, $0 < w^n \leq 1$ denotes a damping parameter that is $w^n = 1$ for a full (undamped) Newton iteration. $\mathcal{N}'_u[w](\cdot, v)$ denotes the derivative of $u \to \mathcal{N}(u, v)$, for $v \in Y$ fixed, at some w in X, and represents the *Jacobian* of the scheme.

1.2 Duality-based *a posteriori* error estimation

In order to highlight the importance of this Jacobian in the context of duality-based *a posteriori* error estimation, in this section we shortly give the main results of its general theoretical framework. For a more detailed discussion see the series of articles [1, 3, 4] and the references cited therein.

Let $J(\cdot)$ be a target or error functional of the solution; for example, $J(\cdot)$ may represent the mean flow across a line, a point value of the solution, or the drag and lift coefficients of a body immersed into an inviscid fluid. For simplicity of presentation, here we assume that $J(\cdot)$ is linear. By employing a duality argument, the following error representation formula may be deduced, see [7, 8] for example,

$$J(u) - J(u_h) = -\mathcal{N}(u_h, z - z_h), \tag{4}$$

where $z_h \in Y_h$, and z is the solution to the following *dual* or *adjoint* problem: find $z \in Y$ such that

$$\mathcal{M}(u, u_h; w, z) = J(w) \quad \forall w \in X. \tag{5}$$

Here, $\mathcal{M}(u, u_h; \cdot, \cdot)$ denotes the mean–value linearisation of the semi–linear form $\mathcal{N}(\cdot, \cdot)$ given by

$$\begin{aligned} \mathcal{M}(u, u_h; u - u_h, v) &= \mathcal{N}(u, v) - \mathcal{N}(u_h, v) \\ &= \int_0^1 \mathcal{N}'[\theta u + (1-\theta)u_h](u - u_h, v)\, \mathrm{d}\theta \end{aligned}$$

for all v in Y.

In order to obtain a computable *a posteriori* estimate of the error $J(u) - J(u_h)$ in (4) we need to replace the – in general – unknown solution z to the dual problem (5) by a numerical approximation $\tilde{z}_h \in \tilde{Y}_h$. To this end, the dual problem (5) is *linearised*: find $\tilde{z} \in Y$ such that

$$\mathcal{M}(u, u_h; w, \tilde{z}) = \mathcal{N}'[u_h](w, \tilde{z}) = J(w) \quad \forall w \in X, \tag{6}$$

and then *discretised*: i.e. find $\tilde{z}_h \in \tilde{Y}_h$ such that

$$\mathcal{N}'[u_h](w_h, \tilde{z}_h) = J(w_h) \quad \forall w_h \in \tilde{X}_h, \tag{7}$$

yielding the required numerical approximation $\tilde{z}_h \in \tilde{Y}_h$ to z. Here, $\tilde{X}_h$ and $\tilde{Y}_h$ denote spaces that are chosen to be richer than the spaces X_h and Y_h, respectively, as for example realised by function spaces which include polynomials of higher degree than the spaces X_h and Y_h.

In common with each Newton iteration step (3), the linearised and discretised dual problem (7) includes the *Jacobian* of the scheme. In fact, problem (7) gives rise to a linear problem whose matrix is simply the transpose of the matrix of a Newton iteration step, assembled on $\tilde{X}_h$ and $\tilde{Y}_h$ which are potentially larger spaces than X_h and Y_h,.

2 Discontinuous Galerkin discretisation of the compressible Euler equations

We consider the two–dimensional steady state compressible Euler equations of gas dynamics. Writing ρ, (v_1, v_2), p and E to denote the density, Cartesian velocity, pressure and total energy per unit mass, respectively, the equations of motion are given by

$$\nabla \cdot \mathcal{F}(\mathbf{u}) \equiv \sum_{k=1}^{2} \frac{\partial}{\partial x_k} \mathbf{F}_k(\mathbf{u}) = 0 \quad \text{in } \Omega, \tag{8}$$

where Ω is an open bounded domain in $\mathbb{R}^2$, supplemented with appropriate boundary conditions on the boundary Γ of the domain. Here, the vector of conservative variables $\mathbf{u}$ and the fluxes $\mathbf{F}_k$, $k = 1, 2$, are defined by $\mathbf{u} = (\rho, \rho v_1, \rho v_2, \rho E)$, $\mathbf{F}_1 = (\rho v_1, \rho v_1^2 + p, \rho v_1 v_2, \rho H v_1)$ and $\mathbf{F}_2 = (\rho v_2, \rho v_1 v_2, \rho v_2^2 + p, \rho H v_2)$, respectively. Additionally, H is the total enthalpy defined by $H = E + p/\rho$. The equation of state of an ideal gas is given by $p = (\gamma - 1)\rho(E - (u^2 + v^2)/2)$, where γ is the ratio of specific heats which, for dry air, is $\gamma = 1.4$.

To formulate the discontinuous Galerkin finite element method (DGFEM, for short) for (8), we first introduce some notation. Let $\mathcal{T}_h = \{\kappa\}$ be an admissible subdivision of Ω into open quadrilateral domains κ; here h is a piecewise constant mesh function with $h(\mathbf{x}) = \text{diam}(\kappa)$ when $\mathbf{x}$ is in element κ. For $p \in \mathbb{N}_0$, we define the following finite element space

$$S_{h,p} = \left\{ \mathbf{v} \in [L_2(\Omega)]^4 : \mathbf{v}|_\kappa \circ \sigma_\kappa \in [\mathcal{Q}_p(\hat{\kappa})]^4 \quad \forall \kappa \in \mathcal{T}_h \right\},$$

where σ_κ denotes a smooth bijective image of the reference element $\hat{\kappa} = (0, 1)^2$ to the element $\kappa \subset \Omega$, and $\mathcal{Q}_p(\hat{\kappa})$ is the set of tensor product polynomials of degree at most p in each coordinate direction over $\hat{\kappa}$. The DGFEM for (8) is defined as follows: find $\mathbf{u}_h \in S_{h,p}$ such that

$$\mathcal{N}(\mathbf{u}_h, \mathbf{v}_h) \equiv \sum_{\kappa \in \mathcal{T}_h} \left\{ -\int_\kappa \mathcal{F}(\mathbf{u}_h) \cdot \nabla \mathbf{v}_h \, \mathrm{d}\mathbf{x} + \int_{\partial\kappa \setminus \Gamma} \mathcal{H}(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa)\, \mathbf{v}_h^+ \, \mathrm{d}s \right\} + \int_\Gamma \mathcal{H}\left(\mathbf{u}_h^+, \mathbf{u}_\Gamma(\mathbf{u}_h^+), \mathbf{n}_\kappa\right)\, \mathbf{v}_h^+ \, \mathrm{d}s = 0 \quad \forall \mathbf{v}_h \in S_{h,p}. \tag{9}$$

Here, $\mathcal{H}(\cdot,\cdot,\cdot)$ denotes a *numerical flux* function, which depends on both the inner– and outer–trace of $\mathbf{u}_h$ on $\partial\kappa$, $\kappa \in \mathcal{T}_h$, and the unit outward normal vector $\mathbf{n}_\kappa$ to $\partial\kappa$. $\mathcal{H}(\cdot,\cdot,\cdot)$ is assumed to be Lipschitz continuous, consistent and conservative; for example, we give descriptions of the (local) Lax–Friedrichs flux and the Vijayasundaram flux in the following.

The (local) Lax–Friedrichs flux $\mathcal{H}_{LF}(\cdot,\cdot,\cdot)$, is defined by

$$\mathcal{H}_{LF}(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa)|_{\partial\kappa} = \frac{1}{2}\left(\mathcal{F}_k(\mathbf{u}_h^+)\cdot\mathbf{n}_\kappa + \mathcal{F}_k(\mathbf{u}_h^-)\cdot\mathbf{n}_\kappa - \alpha\left(\mathbf{u}_h^- - \mathbf{u}_h^+\right)\right), \tag{10}$$

for κ in $\mathcal{T}_h$, where $\alpha = \alpha(\mathbf{u})$ denotes the largest eigenvalue (in absolute value) of the Jacobi matrix $\mathcal{A}(\mathbf{w},\mathbf{n}) = \mathcal{F}'(\mathbf{w})\cdot\mathbf{n}$ in the neighbourhood of $\partial\kappa$, i.e.

$$\alpha(\mathbf{u}) = \max_{\mathbf{w}=\mathbf{u}^+,\mathbf{u}^-} \tilde{\alpha}(\mathbf{w}) \equiv \max_{\mathbf{w}=\mathbf{u}^+,\mathbf{u}^-} \{|v_n(\mathbf{w})| + c(\mathbf{w})\}, \tag{11}$$

where $v_n(\mathbf{u}) = \mathbf{v}\cdot\mathbf{n}$ and $c(\mathbf{u}) = \sqrt{\frac{\gamma p}{\rho}}$ denote the normal velocity and the speed of sound, respectively.

The Vijayasundaram flux $\mathcal{H}_V(\cdot,\cdot,\cdot)$, is defined by

$$\mathcal{H}_V(\mathbf{u}_h^+, \mathbf{u}_h^-, \mathbf{n}_\kappa)|_{\partial\kappa} = \mathcal{A}^+(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)\mathbf{u}_h^+ + \mathcal{A}^-(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)\mathbf{u}_h^- \quad \text{for } \kappa \in \mathcal{T}_h, \tag{12}$$

where $\mathcal{A}^+(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)$ and $\mathcal{A}^-(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)$ denote the positive and negative parts of the Jacobi matrix $\mathcal{A}(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)$, respectively, i.e.

$$\mathcal{A}^\pm = \mathcal{P}\Lambda^\pm\mathcal{P}^{-1}, \qquad \Lambda^\pm = \mathrm{diag}\{\lambda_i^\pm, i = 1,\ldots,4\}, \tag{13}$$

with $\lambda^+ = \max\{\lambda, 0\}$ and $\lambda^- = \min\{\lambda, 0\}$. Here, λ_i, $i = 1,\ldots,4$, and the columns of $\mathcal{P}$ are the eigenvalues and eigenvectors of the matrix $\mathcal{A}(\hat{\mathbf{u}}_h, \mathbf{n}_\kappa)$, respectively, evaluated at the mean value $\hat{\mathbf{u}}_h = \frac{1}{2}\left(\mathbf{u}_h^+ + \mathbf{u}_h^-\right)$.

Finally, the boundary function $\mathbf{u}_\Gamma(\mathbf{u})$ is given according to the type of boundary condition applied. We set $\mathbf{u}_\Gamma(\mathbf{u}) = \mathbf{u}_D$ on the Dirichlet parts of the boundary with a prescribed boundary function $\mathbf{u}_D$, $\mathbf{u}_\Gamma(\mathbf{u}) = \mathbf{u}$ on supersonic outflow parts of the boundary, and $\mathbf{u}_\Gamma(\mathbf{u}) = (\varrho, \varrho v_1, \varrho v_2, \frac{p_{\mathrm{out}}}{\gamma-1} + \frac{1}{2}\varrho v^2)$ on subsonic outflow parts of the boundary with prescribed pressure p_{out}. Furthermore, we set $\mathbf{u}_\Gamma(\mathbf{u}) = \mathbf{u}_{\mathrm{refl}}(\mathbf{u})$ on reflective (slip-wall) boundaries, where $\mathbf{u}_{\mathrm{refl}}(\mathbf{u})$ originates from $\mathbf{u}$ by simply inverting the sign of the normal velocity component of $\mathbf{u}$, i.e. $\mathbf{v} = (v_1, v_2)$ is replaced by $\mathbf{v}^- = (\mathbf{v} - 2(\mathbf{v}\cdot\mathbf{n})\mathbf{n})$ leading to

$$\mathbf{u}_{\text{refl}}(\mathbf{u}) = \mathbf{u}_{\text{refl}}{}'(\mathbf{u})\mathbf{u} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1-2n_1^2 & -2n_1n_2 & 0 \\ 0 & -2n_1n_2 & 1-2n_2^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{u}, \tag{14}$$

where $n_i, i = 1, 2$, are the components of the unit outward normal vector $\mathbf{n} = (n_1, n_2)$ to the boundary Γ.

3 The Jacobian of the numerical scheme

In this section we give a detailed description of the Jacobian $\mathcal{N}'_{\mathbf{u}}[\mathbf{w}](\cdot, \mathbf{v})$ of the semi-linear form $\mathcal{N}(\mathbf{u}, \mathbf{v})$ defined in (9). In particular,

$$\begin{aligned}
&\mathcal{N}'_{\mathbf{u}}[\mathbf{w}](\boldsymbol{\phi}, \mathbf{v}) = \\
&\sum_{\kappa \in \mathcal{T}_h} \Bigg\{ - \int_\kappa (\mathcal{F}'(\mathbf{w})\boldsymbol{\phi}) \cdot \nabla \mathbf{v} \, \mathrm{d}\mathbf{x} \\
&\qquad + \int_{\partial\kappa \setminus \Gamma} \left(\mathcal{H}'_{\mathbf{u}^+}(\mathbf{w}^+, \mathbf{w}^-, \mathbf{n}_\kappa)\boldsymbol{\phi}^+ + \mathcal{H}'_{\mathbf{u}^-}(\mathbf{w}^+, \mathbf{w}^-, \mathbf{n}_\kappa)\boldsymbol{\phi}^- \right) \mathbf{v}^+ \, \mathrm{d}s \Bigg\} \\
&+ \int_\Gamma \left(\mathcal{H}'_{\mathbf{u}^+}(\mathbf{w}^+, \mathbf{w}_\Gamma(\mathbf{w}^+), \mathbf{n}_\kappa) + \mathcal{H}'_{\mathbf{u}^-}(\mathbf{w}^+, \mathbf{w}_\Gamma(\mathbf{w}^+), \mathbf{n}_\kappa)\mathbf{w}'_\Gamma(\mathbf{w}^+) \right) \boldsymbol{\phi}^+ \, \mathbf{v}^+ \, \mathrm{d}s,
\end{aligned}$$

where $\mathbf{w} \to \mathcal{H}'_{\mathbf{u}^+}(\mathbf{w}^+, \mathbf{w}^-, \mathbf{n}_\kappa)$ and $\mathbf{w} \to \mathcal{H}'_{\mathbf{u}^-}(\mathbf{w}^+, \mathbf{w}^-, \mathbf{n}_\kappa)$ denote the derivatives of the flux function $\mathcal{H}(\cdot, \cdot, \cdot)$ with respect to its first and second arguments, respectively.

Furthermore, $\mathbf{u}'_\Gamma(\mathbf{u})$ denotes the derivative of the boundary function $\mathbf{u}_\Gamma(\mathbf{u})$ with respect to the conservative variables in (i.e. the components of) $\mathbf{u}$. We have $\mathbf{u}'_\Gamma(\mathbf{u}) = 0$ on the Dirichlet parts of the boundary, $\mathbf{u}'_\Gamma(\mathbf{u}) = \mathrm{id} \in \mathbb{R}^{4,4}$ on supersonic outflow parts of the boundary, and

$$\mathbf{u}'_\Gamma(\mathbf{u}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2}v^2 & v_1 & v_2 & 0 \end{pmatrix} \tag{15}$$

on subsonic outflow parts of the boundary. Finally, we have $\mathbf{u}'_\Gamma(\mathbf{u}) = \mathbf{u}'_{\text{refl}}(\mathbf{u})$, see (14), on reflective (slip-wall) boundary parts.

It remains to give an expression for the derivatives $\mathcal{H}'_{\mathbf{u}^+}$ and $\mathcal{H}'_{\mathbf{u}^-}$ of the numerical flux function $\mathcal{H}$. Clearly, they depend on the specific choice of the flux function and do – strictly speaking – not exist for many flux functions, as they typically include some non-differentiable terms such as 'min' and 'max' or absolute value functions, for example. Nevertheless, they can be approximated in practice, and, provided the approximation is sufficiently good, the resulting

Jacobian is still capable of delivering an optimal convergence of the Newton iteration as demonstrated in Section 4.

First, we consider the local Lax-Friedrichs flux. According to (10) its ith component is defined by

$$\mathcal{H}_i(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}) = \frac{1}{2}\left(F_{ki}(\mathbf{u}^+)n_k + F_{ki}(\mathbf{u}^-)n_k + \alpha(\mathbf{u})(u_i^+ - u_i^-)\right).$$

Its derivative with respect to its first argument is then given by

$$\begin{aligned}\left(\mathcal{H}'_{\mathbf{u}^+}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n})\right)_{ij} &= \partial_{u_j^+}\mathcal{H}_i(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}) \\ &= \frac{1}{2}\left(\partial_{u_j}F_{ki}(\mathbf{u}^+)n_k + \alpha(\mathbf{u})\delta_{ij} + \left(\alpha'_{\mathbf{u}^+}(\mathbf{u})\right)_j\left(u_i^+ - u_i^-\right)\right),\end{aligned} \tag{16}$$

where $\alpha'_{\mathbf{u}^+}(\mathbf{u})$ might be approximated by

$$\alpha'_{\mathbf{u}^+}(\mathbf{u}) = \begin{cases}\operatorname{sign}\left(v_n(\mathbf{u}^+)\right)v'_n(\mathbf{u}^+) + c'(\mathbf{u}^+) & \text{for } \tilde{\alpha}(\mathbf{u}+) \geq \tilde{\alpha}(\mathbf{u}-), \\ 0 & \text{else,}\end{cases} \tag{17}$$

with $\tilde{a}(\mathbf{u}) = |v_n(\mathbf{u})| + c(\mathbf{u})$, see also (11). Similarly, we compute the derivative $H'_{\mathbf{u}^-}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n})$ with respect to the second argument,.

Finally, we consider the Vijayasundaram flux. According to (12) the ith component of this flux is given by

$$\mathcal{H}_i(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}) = \mathcal{A}_{ij}^+(\hat{\mathbf{u}}, \mathbf{n})u_j^+ + \mathcal{A}_{ij}^-(\hat{\mathbf{u}}, \mathbf{n})u_j^-,$$

where the derivative with respect to the first argument is given by

$$\left(\mathcal{H}'_{\mathbf{u}^+}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n})\right)_{ij} = \mathcal{A}_{ij}^+(\hat{\mathbf{u}}, \mathbf{n}) + \partial_{u_j^+}\mathcal{A}_{ik}^+(\hat{\mathbf{u}}, \mathbf{n})u_k^+ + \partial_{u_j^+}\mathcal{A}_{ik}^-(\hat{\mathbf{u}}, \mathbf{n})u_k^-.$$

Due to the involved dependence of $\mathcal{A}^\pm(\hat{\mathbf{u}}, \mathbf{n})$ on $\mathbf{u}^+$ and $\mathbf{u}^-$, see definition in (13), its derivative $\partial_{u_j^+}\mathcal{A}_{ik}^\pm$ may be very complicated if computed explicitly. Instead, we approximate it by difference quotients, i.e.

$$\begin{aligned}\partial_{u_k^+}\mathcal{A}_{ij}^\pm(\bar{\mathbf{u}}, \mathbf{n}) &= \partial_{u_k^+}\mathcal{A}_{ij}^\pm\left(\tfrac{1}{2}\left(\mathbf{u}^+ + \mathbf{u}^-\right), \mathbf{n}\right) \\ &= \tfrac{1}{2\varepsilon}\left(\mathcal{A}_{ij}^\pm(\tfrac{1}{2}\left(\mathbf{u}^+ + \mathbf{u}^- + \varepsilon\mathbf{e}_k\right), \mathbf{n}\right) \\ &\quad -\mathcal{A}_{ij}^\pm(\tfrac{1}{2}\left(\mathbf{u}^+ + \mathbf{u}^- - \varepsilon\mathbf{e}_k\right), \mathbf{n})) + \mathcal{O}\left(\varepsilon^2\right),\end{aligned} \tag{18}$$

where $\mathbf{e}_k$, $i = 1, 2$, are the unit vectors and $0 < \epsilon << 1$. Again, $\mathcal{H}'_{\mathbf{u}^-}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n})$ is computed accordingly.

4 Numerical results

As described in the last section the Jacobian of the discontinuous Galerkin discretisation (9) can be assembled approximately only. In the following two

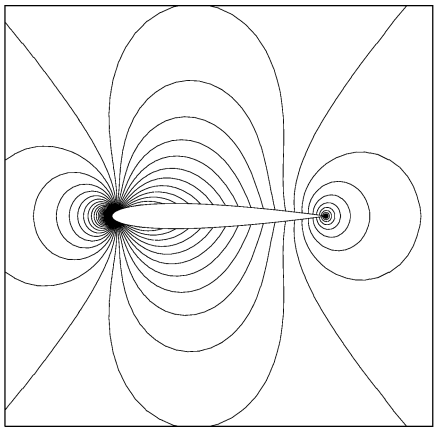

Fig. 1. $M = 0.5$, $\alpha = 0$ flow around the NACA0012 airfoil: Mach number isolines $M = \frac{i}{100}$, $i = 1, 2, \ldots$.

subsections we now present some numerical examples to show that in spite of this approximation the Jacobian is still accurate enough for its two applications mentioned in Sections 1.1 and 1.2 of the introduction. In particular, we will demonstrate the performance of the Newton iteration based on the Jacobian described in Section 4.1. Finally we show in Section 4.2 that the duality-based *a posteriori* error representation gives sharp error estimations when evaluated by solving the adjoint (dual) problem numerically.

4.1 Performance of the Newton iteration

In order to demonstrate the performance of the Newton iteration we first consider a subsonic flow around a NACA0012 airfoil. The length of the profile is rescaled to one. The outer boundary of the computational domain Ω consists of a circle of radius 10 units. There, we prescribe a Mach 0.5 flow at zero angle of attack, with farfield density $\rho = 1$ and pressure $p = 1$. The solution to this problem consists of a strictly subsonic flow, symmetric about the x–axis, cf. Figure 1. The first computation uses piecewise bilinear ansatz and test functions $u_h, v_h \in S_{h,1}$, i.e. DG(1), and the Vijayasundaram flux, see (12). The computation is performed on a sequence of 5 globally refined quadrilateral meshes, starting on a coarse mesh with 39 cells and 624 degrees of freedom up to the finest mesh with 9984 cells and 159744 degrees of freedom. Starting on the coarsest mesh with freeflow conditions the nonlinear problem is solved by the Newton iteration as described in Section 1.1. When the nonlinear residual gets below 10^{-10}, the mesh is globally refined once, the discrete solution is interpolated onto the new mesh and is taken as start solution for the Newton iteration on the new mesh. For each Newton step a linear problem must be

solved. Employing the GMRES solver with Block-Gauss-Seidel preconditioning the residual of each linear problem is reduced by a factor of 10^{-8}.

Table 1 shows the history of the nonlinear solution process. The Newton iteration proceeds with full Newton steps (damping parameter $w^n = 1$) leading to a very fast convergence of the Newton iteration. On each mesh the rate of convergence increases significantly clearly indicating a superlinear convergence. In fact, on several single Newton steps the residual is reduced by a factor of more than 10^3. On all meshes the nonlinear residual is reduced below the tolerance in at most five Newton steps.

Table 1. Nonlinear residual and convergence rates for DG(1).

mesh 1		mesh 2		mesh 3		mesh 4		mesh 5	
res	rate	res.	rate	res.	rate	res.	rate	res.	rate
1.8-01	-	5.8-02	-	2.1-02	-	8.5-03	-	3.5-03	-
1.3-02	14	1.8-03	32	1.2-03	19	6.8-04	12	2.4-04	15
5.5-04	23	1.6-05	110	5.7-05	20	6.2-05	11	1.6-05	15
9.1-07	610	1.0-08	1599	7.5-08	760	1.1-07	540	3.9-08	423
7.6-10	1192	4.6-12	2218	5.5-11	1361	1.6-10	695	5.4-11	718
4.2-13	1820					1.5-13	1060		

Additionally, this history is displayed in Figure 2 which shows a plot of the nonlinear residuals over the total number of Newton steps. In this plot the increase in the rate of convergence is clearly seen in the increase of the slope of the convergence on each mesh. Each time the residual crosses the 10^{-10} tolerance line the solution is interpolated onto the next mesh and the residual jumps back up as it is then meassured on the new mesh.

Table 2 collects the number of steps the Newton iteration requires for reaching the given tolerance on meshes 1-4 for DG(p), $1 \leq p \leq 4$. We see that only about 4 to 5 steps are required for linear elements but similarly few steps also for higher order elements. Naturally, the number of iterations on the coarsest mesh is higher than on the following meshes as the Newton iteration first needs to get into the range of fast (quadratic) convergence. In order to accelerate this process, a DG(1) pre-iterated solution was taken as start solution for DG(4). Furthermore, we see that the number of Newton steps required stays virtually constant while the mesh is successively refined. A variance of at most one step as for example on the fourth mesh in DG(1) occurs by chance as sometimes the fourth iteration step happens to be just below or above the tolerance, see also Figure 2. In the latter case an additional step is needed for converging below the tolerance.

Several numerical tests have been performed for decreasing the computing time of the Newton iteration. In the following we shortly summarise a few of the experiences made.

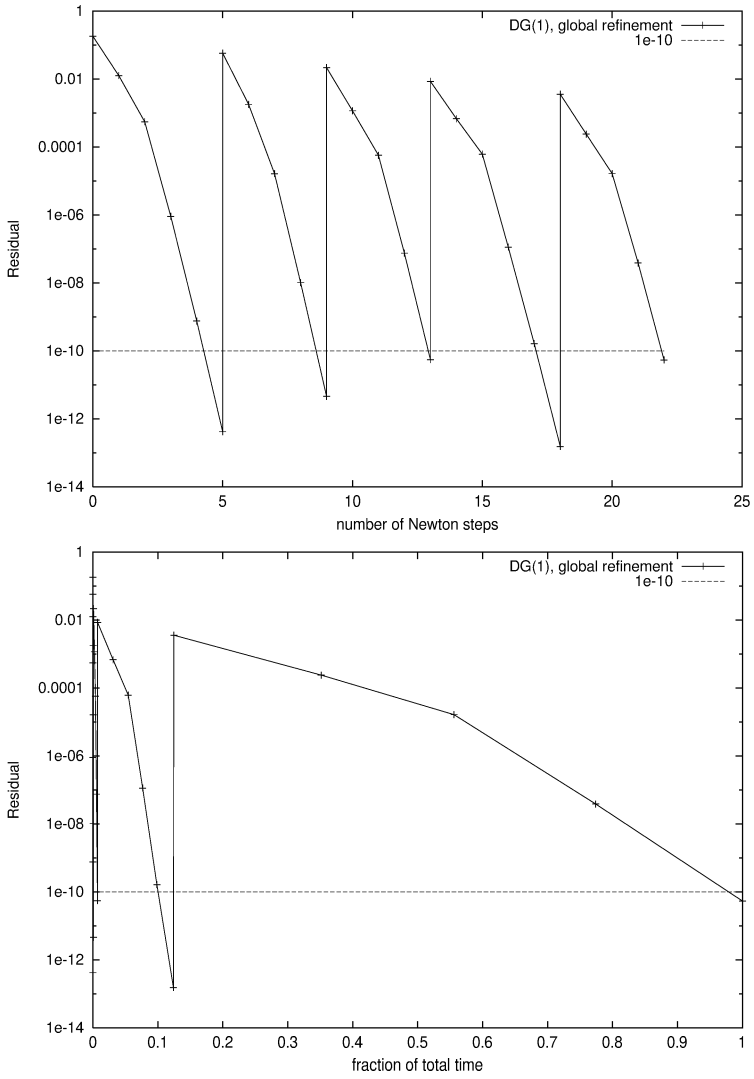

Fig. 2. Convergence of the residual under global refinement plotted over (a) the number of Newton steps (b) the fraction of the total time.

Table 2. Number of Newton steps on each global refinement level for DG(p), $1 \leq p \leq 4$. (* pre-iteration on DG(1))

mesh	DG(1)	DG(2)	DG(3)	DG(4)
1	5	5	6	$6^\star$
2	4	4	5	5
3	4	4	5	5
4	5	4	5	6

1. Simplifying the Jacobian of the numerical fluxes by e.g. neglecting the $\alpha'_{\mathbf{u}^+}(\mathbf{u})$ term in (16) or the $\partial_{u_k} \mathcal{A}^{\pm}_{ij}(\bar{\mathbf{u}}, \mathbf{n})$ terms in (18), as done in [7, 8], leads to small time savings in the assembly of the Jacobian but also to a significant increase of the numbers of Newton steps required. As the overall time is dominated by the time of the linear solver rather than by the assembly time of the Jacobian, this leads to an significant increase of computing time; numerical tests showed an increase of more than 50%. Consequently, the Jacobian should be approximated as accurate as possible.
2. In the numerical tests performed above the linear systems are solved by reducing the linear residual by a factor of 10^{-8}. Numerical tests show that reducing the linear residual on each step by a factor of 10^{-3} or 10^{-4} only, two or three additional Newton steps are required, but the overall computing time is decreased significantly. In numerical tests time savings of up to 65% have been encountered.
3. When the solution on the finest mesh is of interest only, the solutions on the coarser meshes do not need to be iterated below the given tolerance. In fact, performing only one Newton step on all but the finest mesh in the numerical example above gives a start solution for the Newton iteration on the finest mesh which is already in the range of fast convergence, see Figure 3. But, the gain of solving the solution on coarser meshes incorrectly by performing one Newton step only, is bounded by the time of the (exact) solution process on the coarser meshes. In the numerical test above, see Figure 2 (b), this is less than 15% of the total computing time only.

Finally, we demonstrate the performance of the Newton iteration solving the DG(1) discretisation on a sequence of locally refined meshes. Figure 4 (a) shows the convergence of the residual plotted over the number of Newton steps. Again, as in the case of global refinement, see Figure 2, the residual converges below the tolerance in about 4 steps. On the coarsest mesh it takes an additional step for reaching the range of fast convergence, on the finer meshes only 3 steps are required.

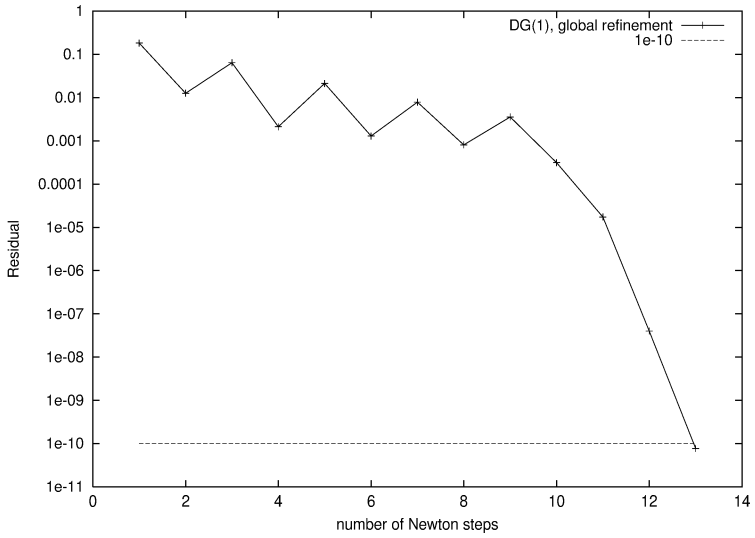

Fig. 3. Convergence of the residual plotted over the number of Newton steps on a sequence of globally refined meshes. On all meshes but the finest only one Newton step is performed.

4.2 Accuracy of the *a posteriori* error estimation

In this section we demonstrate the accuracy of the *a posteriori* error estimation with respect to arbitrary target functionals $J(\cdot)$. Starting from the error representation formula (4) the unknown solution z of the dual problem (5) is replaced by an approximate solution $\tilde{z}_h$ of the linearised and discretised dual problem (7) which results in following approximate error representation

$$J(u) - J(u_h) \approx -\mathcal{N}(u_h, \tilde{z}_h - z_h) = \sum_{\kappa \in \mathcal{T}_h} \eta_\kappa =: \eta, \tag{19}$$

which can be decomposed as a summation of local error indicators η_κ, $\kappa \in \mathcal{T}_h$, over the elements κ in the computational mesh $\mathcal{T}_h$. Employing these so-called *weighted* indicators $|\eta_\kappa|$, $\kappa \in \mathcal{T}_h$, for driving an adaptive mesh refinement algorithm, very economical meshes can be produced, that are specifically tailored to the efficient computation of the quantity $J(u)$ of interest. First results of the *a posteriori* error estimation and goal-oriented adaptivity approach described above applied to simple hyperbolic problems have been published in [5] which includes test cases for the linear advection equation and 1D inviscid Burgers equations. In a sequence of publications, see [7, 8, 10, 11], these results have been extended to the 2D Euler equations, on a variety of problems including the Ringleb flow problem, supersonic flow past a wedge, flows through a nozzle, and sub-, trans- and supersonic flows around different airfoil geometries;

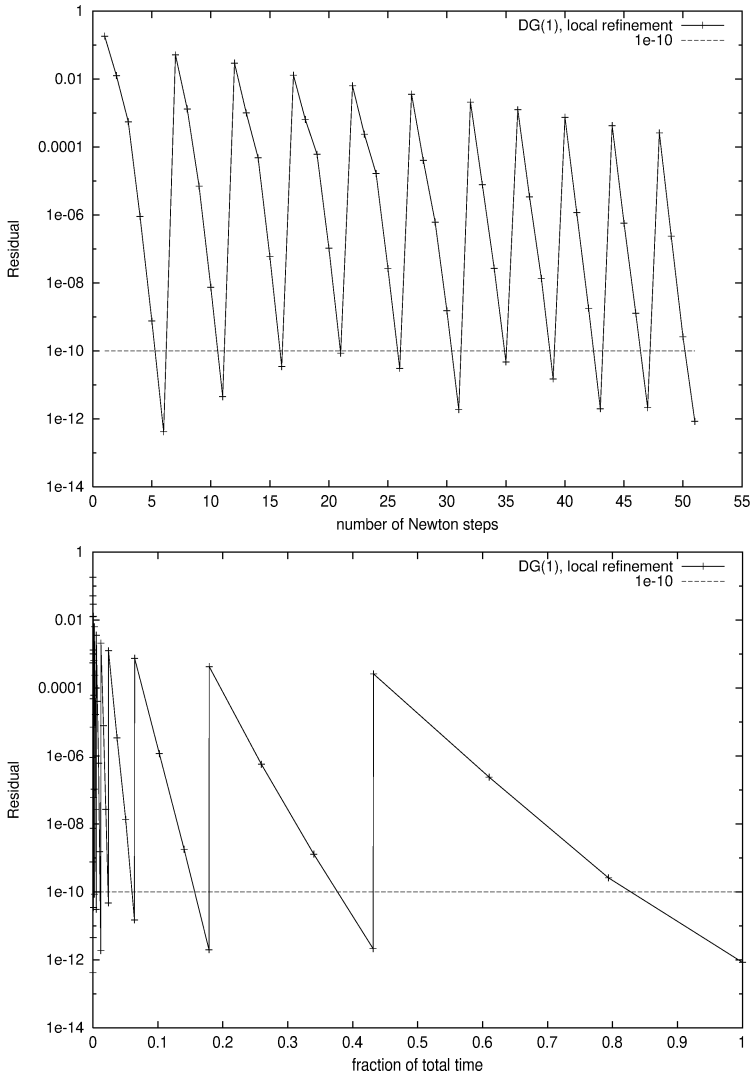

Fig. 4. Convergence of the residual under local refinement plotted over (a) the number of Newton steps (b) the fraction of the total time.

for an overview of typical test cases also including the Buckley-Leverett equation and the 1D Euler equations, see the thesis [6]. Furthermore, this approach has been generalised to the case of multiple target functionals, see [9].

In these publications it has been demonstrated that the meshes produced using the weighted indicators are much more efficient in computing accurate values of the target quantities than meshes that simply rely on *ad hoc* indicators that do not require the solution of a dual problem. Here, efficiency is meassured in terms of number of cells but also in terms of total computing time that includes the time for solving the additional (but linear) dual problem. Furthermore, it has been shown that the approximate error representation yields a good approximation to the true error meassured in terms of the target functional.

In the following, we give an examples of the accuracy of the *a posteriori* error estimation. To this end, we consider the inviscid Mach 0.5 flow at a zero angle of attack around the NACA0012 airfoil, i.e. the test case already considered in the last subsection. Here, we consider the case of estimating the drag on the surface of the airfoil. As for this subsonic flow the exact value of the drag is known to be zero, the true value of the target quantity is $J(u) = 0$. The primal (nonlinear) problem is approximated with discontinuous piecewise bilinear functions, i.e. $u_h \in S_{h,1}$, and the dual (linear) problem is approximated with discontinuous piecewise biquadratic functions, i.e. $\tilde{z}_h \in \tilde{Y}_h \equiv S_{h,2}$. Starting on a coarse mesh the mesh is successively refined using the weighted indicators, where 20% and 10% of the cells are flagged for refinement and coarsening, respectively; for a more detailed discussion of the adaptive refinement algorithm, see [7] or [6].

In Table 3, we demonstrate the performance of the adaptive algorithm. Here, we show the number of elements, the true error in the target functional, $J(u)-J(u_h)$, the computed error representation $\eta = \sum_\kappa \tilde{\eta}_\kappa$, and the effectivity index $\theta = \eta/(J(u) - J(u_h))$. We see that initially on the coarsest meshes the quality of the computed error representation formula η, see (19), is poor, in the sense that θ is not close to one; however, as the mesh is refined the effectivity index θ is very close to unity indicating that the computed error representation gives a very close approximation to the true error.

By employing a block-Gauss-Seidel preconditioned GMRES solver, see [6], the linear residual of each dual problem has been reduced by a factor of 10^{-8}. Numerical tests have shown that it is not necessary to solve the dual problem up to this accuracy. Reducing the linear residual by a factor of 10^{-3} only, mainly retains the accuracy of the error estimation and the efficiency of the locally refined meshes, see Table 4. In fact, the values of the computed error representation are changed slightly only; also the sequence of adaptively refined meshes is almost the same. In this test case, time savings of up to 90% have been encountered for computing the dual solution when its linear residual is reduced by a factor of 10^{-3} instead of reducing it by a factor of 10^{-8}.

Table 3. Subsonic flow around a NACA0012 airfoil. Adaptive algorithm for the evaluation of the drag coefficient. Linear residuals of the dual problems are reduced by factor of 10^{-8}.

# el.	# DoFs	$J(u) - J(u_h)$	$\eta = \sum_\kappa \eta_\kappa$	θ
39	624	-3.902e-02	-2.727e-02	0.70
66	1056	-2.005e-02	-5.485e-03	0.27
120	1920	-1.359e-02	-1.794e-03	0.13
204	3264	-9.147e-03	-4.600e-03	0.50
348	5568	-4.244e-03	-3.321e-03	0.78
570	9120	-1.583e-03	-1.441e-03	0.91
927	14832	-5.760e-04	-5.531e-04	0.96
1557	24912	-2.287e-04	-2.256e-04	0.99
2538	40608	-9.725e-05	-9.592e-05	0.99
4152	66432	-4.850e-05	-4.674e-05	0.96
6747	107952	-2.359e-05	-2.138e-05	0.91

Table 4. Subsonic flow around a NACA0012 airfoil. Adaptive algorithm for the evaluation of the drag coefficient. Linear residuals of the dual problems are reduced by factor of 10^{-3}.

# el.	# DoFs	$J(u) - J(u_h)$	$\eta = \sum_\kappa \eta_\kappa$	θ
39	624	-3.902e-02	-2.727e-02	0.70
66	1056	-2.005e-02	-5.481e-03	0.27
120	1920	-1.359e-02	-1.796e-03	0.13
204	3264	-9.147e-03	-4.589e-03	0.50
348	5568	-4.244e-03	-3.322e-03	0.78
570	9120	-1.583e-03	-1.440e-03	0.91
930	14880	-5.747e-04	-5.547e-04	0.97
1566	25056	-2.277e-04	-2.244e-04	0.99
2565	41040	-9.523e-05	-9.403e-05	0.99
4203	67248	-4.767e-05	-4.456e-05	0.93
6816	109056	-2.331e-05	-2.055e-05	0.88

5 Concluding Remarks

In this paper we have provided a full description of the Jacobian to the discontinuous Galerkin discretisation of the compressible Euler equations which is key ingredient of the Newton iteration for solving the compressible Euler equations and for the construction of adjoint (dual) problems arising in the context of duality-based *a posteriori* error estimation and adaptivity.

In various numerical tests we demonstrated the performance of the Newton iteration. We encountered full Newton convergence on globally as well as on locally refined grids. Similar results were obtained for higher order elements. Furthermore, we demonstrated the accuracy of the *a posteriori* error

estimation for the error in the drag of an airfoil immersed in an inviscid fluid. Numerical tests have shown that solving the linear systems of the Newton iteration steps and the dual problems with lower accuracy, only, may lead to a significant speed up in the solution process without reducing the accuracy of the numerical solution and the *a posteriori* error estimation.

Acknowledgments

The author acknowledges the financial support of the DFG Priority Research Program "Analysis and Numerics of Conservation Laws", the SFB 359 "Reactive Flows, Diffusion and Transport" at the IWR, University of Heidelberg, and the German Aerospace Center (DLR), Braunschweig.

References

1. R. Becker and R. Rannacher. An optimal control approach to error estimation and mesh adaptation in finite element methods. *Acta Numerica*, 10:1–102, 2001.
2. B. Cockburn, G. Karniadakis, and C.-W. Shu. The development of discontinuous Galerkin methods. In B. Cockburn, G. Karniadakis, and C.-W. Shu, editors, *Discontinuous Galerkin Methods*, volume 11, pages 3–50. Springer, 1999.
3. K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for differential equations. *Acta Numerica*, pages 105–158, 1995.
4. M. Giles and E. Süli. Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality. *Acta Numerica*, 2002.
5. R. Hartmann. Adaptive FE Methods for Conservation Equations. In H. Freistühler and G. Warnecke, editors, *Hyperbolic Problems: theory, numerics, applications: eighth international conference in Magdeburg, February, March 2000*, volume 2 of *International series of numerical mathematics; Vol. 141*, pages 495–503. Birkhäuser, Basel, 2001.
6. R. Hartmann. *Adaptive Finite Element Methods for the Compressible Euler Equations*. PhD thesis, University of Heidelberg, 2002.
7. R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. *SIAM J. Sci. Comp.*, 24:979–1004, 2002.
8. R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *J. Comp. Phys.*, 183:508–532, 2002.
9. R. Hartmann and P. Houston. Goal-oriented a posteriori error estimation for multiple target functionals. In T. Y. Hou and E. Tadmor, editors, *Hyperbolic problems: theory, numerics, applications*, pages 579–588, Springer, 2003.
10. P. Houston and R. Hartmann. Goal–oriented a posteriori error estimation for compressible fluid flows. In F. Brezzi, A. Buffa, S. Corsaro, and A. Murli, editors, *Num. Mathematics and Advanced Applications*, pages 775–784. Springer, 2003.
11. P. Houston, R. Hartmann, and A. Süli. Adaptive discontinuous Galerkin finite element methods for compressible fluid flows. In M. Baines, editor, *Numerical methods for Fluid Dynamics VII, ICFD*, pages 347–353, 2001.