Gradient-based and Adjoint-based Sensors for R-refinement - Application and Comparison

Mateusz Gugala*,Jan Hückelheim
Queen Mary, University of London, The School of Engineering and Material Science, Mile End Road, London, E1 4NS, UK
Email: m.gugala@qmul.ac.uk

Shenren Xu
University of Liverpool, School of Engineering, The Quadrangle, Brownlow Hill L69 3GH, UK
Email: s.xu@qmul.ac.uk

Jens Dominik Müller
Queen Mary, University of London, The School of Engineering and Material Science, Mile End Road, London, E1 4NS, UK
Email: j.mueller@qmul.ac.uk

03 May 2015

Summary

Mesh adaptation using *r*-refinement techniques is applied during iterative solution process in order to achieve an improved mesh. The geometric multigrid technique of the flow solver is exploited for error estimation, from which unweighted Hessian as well as adjoint-weighted adaptation sensors are implemented and evaluated. The linear elasticity mesh deformation methodology is applied to relocate mesh nodes while keeping mesh quality at an acceptable level. Results are presented for a fuel efficient vehicle test case in 3D viscous flow using adapted and uniformly refined meshes.

Keywords: *mesh refinement, adjoint, r-refinement, refinement sensors, CFD, viscous flow.*

1 Introduction

Numerical optimisation using CFD and adjoint methods is an important area of CFD methods development and industrial application. Depending on the optimisation algorithm the computational cost of the optimal solution is between 5 and 100 times the cost of a single CFD evaluation with fixed parameters. Minimising the computational cost of the simulation for a given level of accuracy is hence an important aspect to obtain an efficient optimisation method.

Solution-adaptive mesh refinement is a recognised approach to achieve this. The main challenges are in the definition of an adaptation indicator that prescribes the local size and possibly anisotropy of the discretisation molecule, which in turn needs to be based on a local error estimator.

Another important aspect is to develop efficient mesh adaptation techniques that then modify the mesh to respond to the adaptation sensor, while maintaining mesh quality. Four groups of mesh adaptation techniques can be distinguished: h-refinement[1] locally adds or removes cells, typically done in a hierarchic fashion to maintain locality; r-refinement[2] retains the topology of the mesh but relocates nodes, hence avoids expensive data movement p-refinement[3] increases the order of accuracy of the discretisation, which is difficult to achieve for typical finite volume methods; and mesh-regeneration[4] produces a new mesh which avoids strong gradients in the cell sizes, but requires interpolation of the solution.

Here we focus on *r*-refinement due to its low impact on the data structure and parallel partitioning, as well as its low computational cost making it suitable for frequent application in one-shot design algorithms[5] and unsteady flows.

The paper is structured in the following manner. First the methods for discretisation error estimation as well as the currently used indicators are discussed in Sec. 2. The refinement methodology is described in Sec. 3. Sec. 4 presents the in-house solver and Sec. 5 the test case considered concluding with discussion in Sec. 6.

2 Error estimation and indicators

2.1 Error estimation

Assuming that the physical model used in computations is valid, the overall simulation error is related to discretisation error - the difference between the exact (continuous) and the discrete solution. The exact quantification of this error requires the knowledge of exact solution to the continuous problem, which for most cases in CFD, especially those of industrial relevance is unknown. That raises the question on how the error could be approximated. The most popular methods of error estimation are:

- the feature based methods that link gradients and the Hessian of user-specified flow variables to the discretisation error,
- output based methods,[1,2,6–8] which quantify the error of an integral quantity of interest L (e.g. lift or drag) with the use of adjoint solution.

In both cases the starting point for error approximation is the expansion of the specified quantity into Taylor series. For gradient/Hessian-based estimation the solution of the system of flow equations (9) is expanded as presented in Eq. (1).

$$u = u^H + \left.\frac{\partial u}{\partial x}\right|_H (H) + \dots \tag{1}$$

where u is e.g. a chosen scalar variable from the exact solution vector U of the system of flow equations (9) and u^H the approximate solution variable on the discretised space with characteristic mesh size H. From that, an approximation for the error can be derived for gradient (2) and Hessian (3) error estimators.

$$\varepsilon_\nabla \approx H\nabla u \tag{2}$$

$$\varepsilon_{\nabla^2} \approx H^T \left(\nabla^2 u\right) H \tag{3}$$

The starting point for output-based error estimation using the adjoint solution is the expansion of integral quantity L (e.g. lift or drag) into Taylor series (Eq. (4)),

$$L_h(U_h) = L_h(U_h^H) + \left.\frac{\partial L_h}{\partial U_h}\right|_{U_h^H} \left(U_h - U_h^H\right) + \dots \tag{4}$$

where U is an exact solution to the system of Partial Differential Equations under consideration (e.g. (9)). The superscript H and subscript h refer to a characteristic length in coarse and refined discrete spaces, respectively (e.g. edge length). In this manner U^H is a solution on the coarse and U_h on the refined grid. In case $h \to 0$, the exact solution is reached. The notation U_h^H refers to the approximate solution on the coarse grid H expressed on the refined mesh h via a carefully chosen reconstruction or projection operator I_h^H.[1] The discretisation error can the be estimated as (see e.g. Venditti[1] or Fidkowski[6] for details of the derivation)

$$L_h(U_h) \approx L_h(U_h^H) - \left(v_h|_{U_h^H}\right)^T R_h(U_h^H) \tag{5}$$

where v is the adjoint variable (10) and R is the residual of flow equations (9).

Eqs. (1) and (4) can be used to either calculate an exact discretisation error when the solution $U_h|_{h\to 0}$ is known, or to estimate the error when approximate solutions are evaluated i.e. one on the coarse mesh H and a second on the refined mesh h. As was mentioned already, the exact solution is unknown and computationally not achievable. However, when properly used, the Eq. (1) and (4) can be successfully applied for discretisation error estimation as the leading error terms are usually associated with the first and second derivative terms of Taylor expansion series. The key requirement is that the discretisation space H has to be fine enough in order to capture relevant flow physics and for the solution U^H to be within the asymptotic range.[1]

In this paper the set of multi-grid meshes used in the in-house flow solver described briefly in section 4 will be reused for the error estimation procedure extended from the work of Venditi[1] or Fidkowski.[6]

2.2 Gradient/Hessian-based indicator

This type of indicators are broadly used and available in most commercial solvers (e.g. ANSYS Fluent[1]). They are based either on the error estimated with Eqs. (2) and (3), or mixed weighted approaches for a chosen flow quantity (e.g. pressure). The final indicator $I_{e_{ij}}$ is calculated for mesh edges e_{ij} by taking the average of gradient or Hessian from two forming nodes i, j (Eqs. (6), (7)).

$$I_{\nabla e_i j} = \left|\frac{\nabla u_i + \nabla u_j}{2} e_{ij}\right| \tag{6}$$

$$I_{\nabla^2 e_i j} = \left|e_{ij}^T \left(\nabla^2 u_{e_{ij}}\right) e_{ij}\right| \tag{7}$$

The advantage of this approach is its simplicity and low computational cost. However in practical application the accuracy improvement is not always achieved.[2] Moreover, What is more this these indicators act locally without taking into account more complex flow dependencies.[9]

2.3 Adjoint-weighted indicator

The main idea behind using adjoint based/weighted indicators is to quantify the effect that a local error has on the objective function, hence promising to overcome problems related to the solution-based indicators. The adjoint solution allows to locate areas in the computational domain that have a strong influence on the objective function of interest (e.g. lift, drag).

In that manner not only the mesh is optimised with respect to defined cost function of interest, but also the information provided has a more global nature i.e. the influence of various flow features on each other is well predicted (opposed to local action of gradient-based indicators).[2] That in principle can lead to optimal ratio of achieved cost function estimation accuracy to computational cost and make adjoint based sensors a very good candidate for adaptation indicator.

[1] http://www.ansys.com

The successful application of adjoint weighted sensors was done by e.g. Fidkowski[6] on a range of inviscid, laminar and RANS[2] cases.

3 R-refinement methodology

The *r*-refinement technique is used to drive the adaptation process. There are several deformation algorithms broadly used in CFD field, amongst which the most popular are spring analogy and linear elasticity technique.
Spring analogy or Laplacian smoothing is attractive due to its low cost and simple implementation, but it usually fails when larger deformations are required and in particular when high aspect ratio cells are present in the mesh. It can easily generate negative volumes[10] and lead to failure of the adaptation procedure.

In order to keep the mesh quality at an acceptable level and to decrease the likelihood of creating negative volumes, the linear elasticity formulation (8) is used here.[10,11] The adaptation procedure can be successfully done by applying non-zero body force f as described by e.g. Dwight.[12]

$$\nabla\sigma = f \qquad \text{on} \qquad Omega, \tag{8}$$

where f is some body force, σ is a stress tensor which is a function of strain tensor ε and *Omega* is the computational domain. The detailed description of how the term f is derived from the adaptation sensors will be presented in the final paper.

4 Flow and adjoint solver

For the purpose of this work the in-house flow and adjoint solver mgOpt[13] was used. mgOpt is a typical finite volume, vertex-centred flow solver with an edge-based data structure. It is able to simulate both inviscid as well as viscous flows, for the later the Spalart Allmaras turbulence model[14] is used. The system of equations can be written in short notation (9)

$$R(U,x) = 0 \tag{9}$$

Where R stand for residuals for each out of 5/6 (inviscid/viscous) equations that are drive to zero and U and x are the vector of flow variables and mesh coordinates respectively. The equations are advanced in time using an implicit Runge-Kutta scheme with multi-grid acceleration on the outside and a GMRES solver on the inside[15]

The discrete adjoint solver is derived using automatic differentiation via source transformation with the AD tool Tapenade.[16] The adjoint system (10) is obtained in a semi-automatic fashion, i.e. first the user selected routines are differentiated and then appropriately assembled using hand-written drivers.[13,17] The first term in Eq. (10) is the Jacobian of the system (9), v is a vector of adjoint variables and the last term is the so called adjoint source term.

$$\left(\frac{\partial R}{\partial U}\right)^T v = \left(\frac{\partial L}{\partial U}\right)^T \tag{10}$$

[2] Reynolds Averaged Navier-Stokes

The system (10) is solved by reusing iterative solver from primal code.

5 Test case

The geometry of a fuel efficient vehicle shown in Fig. 1 designed at Warsaw University of Technology is used as a test case. The car is design by the members of Students Association of Vehicle Aerodynamics[3] and take part in the international Shell Eco-marathon competition[4]. The engineering quantity of interest is the total drag force.

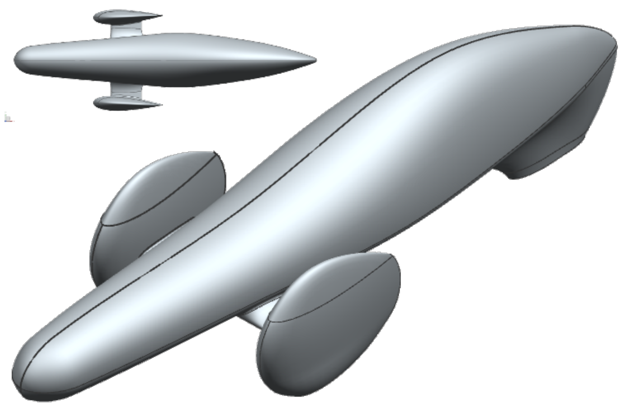

Figure 1: *Geometry of fuel efficient vehicle.*

The test case is a realistic three dimensional case with relatively complex multi-body shape. In order to capture physics properly the viscous flow solver of mgOpt in-house code is used. The initial flow analysis reveals presence of complex flow features at the intersection between vehicle main body and wheel support (figure 2), the so called horse-shoe vortex. It has a potential to be a good test for adaptation sensors and r-refinement algorithm.

The initial mesh created for presented geometry will be (approximately) uniformly distributed at the vehicle surface and inside the computational domain in the vicinity of the vehicle. The more refined boundary layer - in the normal surface direction, will be created with prism elements. It is expected that the refinement algorithm will try to relocate mesh towards the wheel support and regions of high geometrical curvature of the concept-car where the vortex from Fig. 1 is formed. The changes in estimated error will be gathered at each r-refinement step and presented on the graph. Additionally the attempt to generate uniformly refined mesh around the vehicle with the target error similar to one estimated will be made and the computational cost compared to r-refined grid.

[3] http://www.skap.meil.pw.edu.pl/
[4] http://www.shell.com/global/environment-society/ecomarathon.html

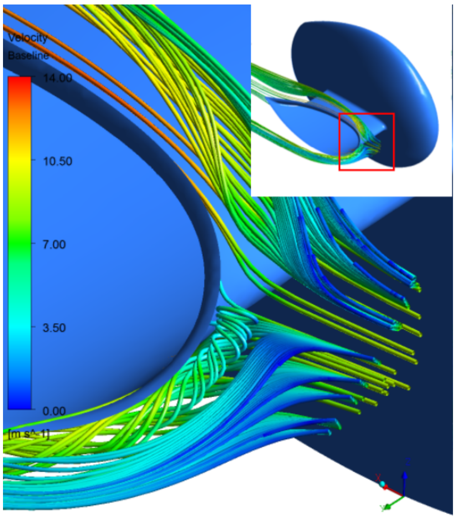

Figure 2: *Horse shoe vortex.*

6 Discussion

The geometric multigrid method is an integral part of the in-house mgOpt code used in this work. Having a geometric multi-grid method at our disposal suggest to use the difference between coarse and fine grids as embodied in the multi-grid right-hand side correction term as an error estimator and to perform adjoint driven mesh adaptation.

The analysis and comparison between various sensors amongst those mentioned in section 2 is performed on a realistic concept-car test case. While performing the comparison and analysing the result of various adaptation indicators the author will try to come up with alternative ways of defining adjoint based sensors with a main goal to achieve even better indication for adaptation process.

7 Acknowledgement

This research utilised Queen Mary's MidPlus computational facilities, supported by QMUL Research-IT and funded by EPSRC grant EP/K000128/1.

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no [317006].

References

[1] Venditti, D. A. and Darmofal, D. L. Grid adaptation for functional outputs: application to two-dimensional inviscid flows. *Journal of Computational Physics* **176**(1), 40–69 Nov (2002).

[2] Dwight, R. P. Heuristic a posteriori estimation of error due to dissipation in finite volume schemes and application to mesh adaptation. *Journal of Computational Physics* **227**(5), 2845–2863 (2008).

[3] Yano, M., Modisette, J. M., and Darmofal, D. L. The importance of mesh adaptation for higher-order discretizations of aerodynamic flows. In *20th AIAA CFD Conference, AIAA-2011*, volume 3852, (2011).

[4] Majewski, J. An anisotropic adaptation for simulation of compressible flows. *Mathematical Modelling and Analysis* **7**(1), 127–134 (2002).

[5] Hazra, S. and Jameson, A. One-shot pseudo-time method for aerodynamic shape optimization using the navier-stokes equations. *AIAA Paper* **1470** (2007).

[6] Fidkowski, K. J. and Darmofal, D. L. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA journal* **49**(4), 673–694 Apr (2011).

[7] Venditti, D. A. and Darmofal, D. L. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics* **187**(1), 22–46 (2003).

[8] Pierce, N. A. and Giles, M. B. Adjoint recovery of superconvergent functionals from pde approximations. *SIAM review* **42**(2), 247–264 (2000).

[9] Warren, G. P., Anderson, W. K., Thomas, J. L., and Krist, S. L. Grid convergence for adaptive methods. *AIAA paper* **1592**, 1991 (1991).

[10] Nielsen, E. J. and Anderson, W. K. Recent improvements in aerodynamic design optimization on unstructured meshes. *AIAA journal* **40**(6), 1155–1163 (2002).

[11] Jasak, H. and Weller, H. Application of the finite volume method and unstructured meshes to linear elasticity. *International journal for numerical methods in engineering* **48**(2), 267–287 (2000).

[12] Dwight, R. P. Robust mesh deformation using the linear elasticity equations. In *Computational Fluid Dynamics 2006*, 401–406. Springer (2009).

[13] Christakopoulos, F. *Sensitivity computation and shape optimisation in aerodynamics using the adjoint methodology and automatic differentiation*. PhD thesis, Queen Mary, University of London, (2012).

[14] Allmaras, S. R. and Johnson, F. T. Modifications and clarifications for the implementation of the spalart-allmaras turbulence model. In *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, 1–11, (2012).

[15] Xu, S., Radford, D., Meyer, M., and Müller, J.-D. Stabilisation of discrete steady adjoint solvers. **submitted for publication** (2014).

[16] Hascoët, L. and Pascual, V. The Tapenade Automatic Differentiation tool: Principles, Model, and Specification. *ACM Transactions On Mathematical Software* **39**(3) (2013).

[17] Christakopoulos, F., Jones, D., and Müller, J.-D. Pseudo-timestepping and verification for automatic differentiation derived cfd codes. *Computers & Fluids* **46**(1), 174–179 (2011).