Adaptive Mesh Generation for Solution of Incompressible Fluid Flows using High Order Gradients

by

Randall Shortridge

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved July 2011 by the
Graduate Supervisory Committee:

Kang Ping Chen, Co-Chair
Marcus Herrmann, Co-Chair
Valana Wells
Huei-Ping Huang
Hans Mittelmann

ARIZONA STATE UNIVERSITY

August 2011

ABSTRACT

A new method of adaptive mesh generation for the computation of fluid flows is investigated. The method utilizes gradients of the flow solution to adapt the size and stretching of elements or volumes in the computational mesh as is commonly done in the conventional Hessian approach. However, in the new method, higher-order gradients are used in place of the Hessian. The method is applied to the finite element solution of the incompressible Navier-Stokes equations on model problems. Results indicate that the method facilitates the use of high solution orders with gradient-based adaptive meshes, and that a significant efficiency benefit is realized. In addition, the improved accuracy affords the use of unstructured surface meshes. This promises improvements in automation of the mesh generation process.

To my family

TABLE OF CONTENTS

Section Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

1. INTRODUCTION

Most practical fluid flows involve complex geometries and localized flow features. The accurate prediction of such flows therefore requires methods that are flexible and efficient in adapting to both the geometry of the flow, and the flow solution itself. Although the geometry of the flow may be known *a priori*, the nature and distribution of flow features such as boundary layers, wakes, eddies, and shed vortices are not. This is especially true in problems where flow features interact with the flow boundaries, or with each other. To capture these features, the flow domain must be discretized finely enough to resolve them. Selective refinement of the mesh in the vicinity of flow features allows their resolution without incurring the computational cost of solving the flow equations on a uniformly refined mesh.

Increasingly, such selectively refined meshes are generated using solution-adaptive techniques (Frink *et al*. 2010, Sahni *et al*. 2010, Kim *et al*. 2010). A flow solution, obtained typically with a finite element or finite volume flow solver on a crude mesh, is used to obtain criteria for generation of an adapted mesh. The resulting mesh is then used to obtain a more accurate flow solution and so on.

Perhaps the most commonly used method of adaptive mesh refinement is the Hessian approach (Sahni *et al*. 2010, 2005, Remaki & Habashi 2009, Frey & Alauzet 2005). This gradient-based method uses local error estimates based on the second gradient, or Hessian, of one or more scalar flow quantities at points within the flow domain. By taking the eigenvalue decomposition of a Hessian

matrix, the principal values and directions of the second gradient are obtained. These are used as criteria for local refinement and/or coarsening of the mesh.

This method is limited in that it may only be used effectively with flow solvers with orders of accuracy equal to two or less. This is because, as a simple Taylor series expansion shows, the dominant error term with methods of third or higher order of accuracy involves third or higher gradients. These gradients are third or higher order tensors, which cannot be treated with the conventional eigenvalue decomposition.

It appears, to this researcher, that this limitation has restricted adaptive finite element methods in the literature to only constant or linear interpolation functions. If this is true, then it has prevented efficiency gains that should be associated with the use of higher-order elements.

Additionally, second and higher order gradients of the vector velocity are third and higher order tensors. Thus adaptation based on these gradients has not been practical. This has disallowed the use of the complete vector representation of the flow in adapting the mesh.

Results of this work show that the recently discovered multilinear singular value decomposition (multilinear SVD) may be used to obtain the relevant principal components of solution gradient tensors of arbitrary order. In the context of mesh adaptation, this should eliminate the above-mentioned restrictions on the selection of elements and error indicators. Application of the multilinear SVD to adaptive finite element mesh generation will be referred to herein as the new method. In order to illustrate the potential advantages of the new method, it

will be compared with some other methods in current use. To facilitate this comparison, a few subjects relating to mesh generation are reviewed below.

1.1 Structured and Unstructured Meshes (Grids)

Early works in CFD commonly employed the finite difference method on so-called structured grids. The finite difference method requires a certain amount of grid structure to take advantage of efficiencies that result from the inherent accuracy of derivatives computed on symmetrically distributed patterns of nodal solution values. The first grids used were rectangular or circular, utilizing Cartesian or polar coordinates.

Computational grids were later stretched to follow the shapes of complex flow domains, and the flow equations solved using appropriate coordinate transformations. Adaptation of these grids to the flow was achieved by local stretching or compression to minimize the solution error for a given computational effort. Difficulties with fitting these structured, curvilinear grids into certain complex domains led to the use of multiple grids. Additional flexibility was afforded by allowing the grids to overlap. This approach is still commonly used in CFD.

Overlapping structured grids can be used to discretize flow regions of any shape, and are used extensively in industry for flow analyses around aircraft for example. A major shortcoming of this approach is the large effort required to fit such grids to complex shapes, and to specify the appropriate spacing within each

grid so that the resulting flow solution is sufficiently accurate. Another shortcoming is the limited ability of the structured grids to adapt to the solution.

Unstructured mesh methods were applied to CFD to allow the solution of flows in complex domains using a single mesh. Employing triangular elements in two dimensions and tetrahedral elements in three dimensions, unstructured meshes can be used to fill any flow domain without difficulty. This gives them a distinct advantage over structured meshes. Another advantage that unstructured meshes afford is the ability to adapt more fully to flow features. The example meshes in Figure 1 illustrate this fact. Note that the unsteady viscous wake appears to be well resolved only by the unstructured mesh in the case shown.

1.2 Adaptation Strategies

Although many forms of mesh adaptation have been developed, the prevalent ones are the traditional Hessian approach mentioned above, and the newer adjoint method, which uses optimization principles. The adjoint method is not nearly as widely used as the Hessian approach, but is gaining popularity. Each method has very distinct advantages in different applications. For this reason, both methods will probably be used extensively into the foreseeable future. Approaches that combine the two methods show promise as well. The adjoint method will be discussed briefly first, and then the Hessian approach will be discussed in some detail.

1.2.1 Adjoint-Based Mesh Adaptation

The adjoint method utilizes optimization principles to generate computational meshes that minimize the error of a functional (Venditti & Darmofal 2003, Frink, *et al*. 2010). A starting mesh is analyzed to determine the sensitivity of the functional to perturbations in the geometry of the mesh. If the functional is sensitive to mesh perturbations at a given point, the mesh is refined at that point to improve the accuracy of the calculation of that functional. The chosen functional is often an output to be maximized or minimized in the optimization of a design, such as the lift or drag of an aircraft wing. The method is well suited to such optimization, and is, in fact, a procedure for optimizing the mesh itself.

A drawback to the use of the adjoint method is that it will only display detailed flow results in those regions that most strongly affect the functional output based on the computed solution. Thus the method may not be best suited to learning about the details of the nature of a given flow.

Another drawback has to do with the nonlinear character of the flow equations, which arises from the convective aspect of fluid flow. Unless the mesh is sufficiently refined in a certain region, its impact on the output functional may be underestimated. Additionally, in unsteady problems, the effect of a change in the mesh may be difficult to assess, because its effect may only be evident at a later time. For example, vortices that are shed by one component of an aircraft may later interact with another component of the aircraft structure. This was amply illustrated in a recent investigation of tail buffet in the F/A-18 fighter aircraft (Morton et al. 2007). A computation must preserve, and therefore resolve, the

vortex feature during transit from one portion of the aircraft to another, when its effect on the structure may be quite small. A gradient based mesh will therefore be more effective in such a case.

The adjoint method has not yet demonstrated the ability to effectively generate anisotropic (elongated) elements or volumes to suit the flow. The adjoint method has been combined with gradient based adaptation to overcome this limitation (Venditti & Darmofal 2003).

A significant advantage of the adjoint method has been that it allows the use of higher orders of solution approximation. This is because it does not depend on gradients of the flow for information with which to adapt the mesh. This benefit has been somewhat nullified, however, by the inability of the method to handle significant anisotropy on its own. The new method to be discussed herein shows the potential to combine high order and anisotropic capabilities in one simple approach.

1.2.2 Gradient based Adaptation – The Hessian Approach

Gradient based adaptation utilizes a gradient of one or more scalar flow variables as an error indicator to drive the adaptation process. Usually the Hessian, or second gradient, of the Mach number or the magnitude of the velocity is used for this purpose, although the pressure may also be used. The eigenvalue decomposition is used to determine the principal gradients which are then used for local error estimation. This is commonly referred to as the Hessian approach, and is probably the most widely used method of unstructured grid adaptation in CFD

today (Sahni *et al*. 2005, 2010, Frey & Alauzet 2005, Dompierre *et al*. 2002, Remaki & Habashi 2009). The Hessian approach may be used with both structured and unstructured grids (meshes). Unstructured meshes allow a much greater degree of local refinement or coarsening to suit local flow conditions.

1.2.2.1 Problems with the Hessian Approach

Although the Hessian approach is probably the most popular method at present for mesh adaptation, it is subject to serious limitations. Those that are apparent to this researcher are outlined below.

1.2.2.1.1 Low Solution Order

The order of the gradient used as an error indicator limits the order of accuracy of the flow solution. This is because the gradient used should reflect the leading error term in a Taylor series expansion of the solution within an element or volume. The leading error term is the first and largest term among those that are truncated to form the approximate representation of the solution.

For example, in a one-dimensional problem, and if piecewise–constant finite elements are used, the leading error term contains the first derivative of the solution. Thus the first derivative or first gradient of the solution at a point is the appropriate gradient for local error estimation. With finite elements, a piecewise constant representation generally leads to first-order accuracy. That is, reducing the element size by one-half reduces the local error by one-half as well. For this case then, we can see that the gradient order required is equal to the order of

accuracy of the representation. This is found to be true as well for higher order representations.

Thus we can see that the Hessian approach, which relies on second derivatives of the solution to estimate local errors, is limited to use with solution methods that are second-order accurate.

It appears to this researcher that this limitation imposed by the Hessian approach is the reason that gradient-based, unstructured methods with order of accuracy greater than two do not seem to be evident in the literature or in industry.

The efficiency of a method of solving the flow equations is strongly tied to its spatial order of accuracy. Because of this, and because the solution of many flow problems requires a considerable expenditure of computing resources, the order of accuracy limitation imposed by the Hessian approach is of particular concern.

Because the viscous shear stress on a surface results from derivatives of the velocity, the accuracy of the calculated stress is generally not as good as that of the velocity field. For example, when finite elements are used, the order of accuracy of the stress is one less than that of the velocity. Because the Hessian approach limits the order of accuracy of the velocity field to two, the maximum order of accuracy of the surface stress is one. This adversely affects the accuracy of important surface forces such as the lift and drag on aircraft.

1.2.2.1.2 Loss of Vector Information

The gradient of the Mach number or magnitude of the velocity is used almost universally in the application of the Hessian approach. Work by this researcher and others (Venditti & Darmofal 2003) indicates that the loss of directional information contained in the vector velocity field results in significant degradation of computational meshes. In regions that are largely inviscid, such as in external flows outside boundary layers, elements are created that are excessively large. This results in a lack of accuracy in such regions, as well as a sharp transition in element size at boundary layer edges.

Evidence collected by this researcher indicates that the loss of directional information also contributes to a kind of mesh instability inside boundary layers, near the solid surfaces. This instability has been encountered with some regularity when gradients of a scalar are used as in the Hessian approach. An example of this behavior is given in the results section of this work.

Also, when steady flow reversal occurs, as within eddies, there exist locations where the relative flow velocity is small. At these locations, the Mach number or magnitude of the velocity can display a singularity or near singularity that results in runaway mesh refinement. This singularity is akin to that present in the function $y=|x|$ at the origin, and results from taking the magnitude of the vector velocity. As the mesh is refined, the singularity is resolved with increasing clarity, and the demand for additional refinement continues without bound.

Performing a simultaneous reduction of quadratic forms to combine the information from the individual velocity components (Castro-Diaz 1997,

Borouchaki *et al.* 1997, Remaki & Habashi 2009) does not guarantee improvement with respect to the above problems. A significant loss of vector information still results, and in fact the mesh becomes coordinate-system dependent.

1.2.2.1.3 Failure of the Hessian Approach in Boundary Layers near Surfaces

In boundary layer flows, the second gradient of the velocity normal to the surface is driven by the streamwise pressure gradient. If the streamwise pressure gradient is small, which is a common occurrence, the second derivative in the normal direction becomes small as well, as in Figure 2. This can be seen from the steady two dimensional momentum equation in a boundary layer:

$$u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=-\frac{1}{\rho}\frac{\partial p}{\partial x}+\nu\frac{\partial^2 u}{\partial y^2}$$

Near the surface, both u and v are small. Thus, near the surface,

$$\frac{1}{\rho}\frac{\partial p}{\partial x}\approx\nu\frac{\partial^2 u}{\partial y^2}$$

So when $\frac{\partial p}{\partial x}\approx 0$, then $\frac{\partial^2 u}{\partial y^2}\approx 0$.

When this second derivative is small, the Hessian approach, which relies on the second gradient for error indication, produces large elements near the surface. This, in conjunction with the first-order accuracy of surface stresses mentioned above, can result in large errors in computed surface forces such as lift and drag.

To deal with this problem, researchers and commercial CFD code developers use special techniques to generate acceptable meshes near boundaries. One

approach uses a hybrid structured/unstructured mesh incorporating a structured region near any solid surface, and an unstructured mesh further away. The Advancing Layers Method (ALM) is a popular method of this type (Frink 2010, Mavriplis & Pirzadeh 1999, Pirzadeh 1994, 1996). A structured mesh is generated in layers, first adjacent to the surfaces, and then further away. At some distance from the surface, the mesh generation continues in an unstructured manner. Such methods require user input of parameters for the generation of the structured regions, and require complex coding to deal with intersecting surfaces. At both inside and outside corners, the structured pattern breaks down, making necessary the removal or addition of elements to eliminate collisions or voids (Aubry & Lohner 2009). Examples of structured surface meshes are shown in Figure 3 and Figure 4.

The continued energy that researchers are putting into development of ALM, and the prevalence of structured surface meshes in industry are indications of the shortcomings of current adaptive grid methodology, and in particular, the Hessian approach.

1.3 Multilinear SVD

Recently (2000), De Lathauwer published a generalization of the singular value decomposition (SVD) to tensors of order greater than two. He refers to the generalization as both a multilinear singular value decomposition and as the high-order singular value decomposition (HOSVD). The Multilinear SVD was

developed for application to high-order statistics, and has also found application in communications, robotics, and facial recognition.

The multilinear SVD is of interest with respect to adaptive mesh generation in CFD because of the potential to extend gradient–based adaptation beyond the Hessian approach to the utilization of higher order velocity gradient tensors. This will allow the use of higher-order solution methods with mesh adaptation, and in conjunction with such methods, appears to have the potential to alleviate all of the above mentioned problems that plague the Hessian approach.

The Multilinear SVD is simply obtained by applying the SVD to an unfolded two dimensional matrix form of the original tensor. Thus the multilinear SVD is computed using existing SVD algorithms which are efficient and reliable.

2. METHOD

The finite element method was implemented in a FORTRAN program to solve the 2D, laminar, incompressible Navier-Stokes equations:

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \mu \nabla^2 u$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \mu \nabla^2 v$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

where

$$\nabla^2 \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

2D problems possess all of the essential characteristics that are required for the assessment of the new method. The simplicity of application that is afforded by

working in only two dimensions reduces the number of variables that can cloud the assessment of the method.

Time stepping is achieved using the second-order Crank-Nicolson scheme. The non-linearity in the equations is handled using Picard's method. This method was chosen because it is extremely stable. This stability is helpful when experimenting with discretization schemes.

Both steady and unsteady flows are treated using a fully implicit frontal solver. (Hood, 1976) This solver reduces access to permanent storage, which speeds computation. The fully implicit approach allows the use of a Galerkin scheme, as opposed to the commonly-used streamwise-upwind Petrov-Galerkin scheme (SUPG) (Hughes *et al*. 1986). Thus solutions are free from upwinding errors in the form of artificial viscosity.

2.1 Finite Elements

Triangular finite elements were used exclusively in this study because of their ability to fill any two-dimensional domain, regardless of shape, and their ability to accommodate large gradients in mesh density for the efficient resolution of flow features.

Elements were used which satisfy the Ladyshenskaya-Babuska-Brezzi (LBB) stability condition. This avoids the use of problem-dependent stabilization factors which would introduce additional variables into the computational experiment.

Linear-Linear elements, that is, elements with linear shape functions for the representation of both the velocity and pressure fields, are commonly used in

CFD, and are used exclusively in the context of gradient based adaptation. Thus this combination was selected for the low-order element pair. To create an LBB-stable configuration, four velocity elements are enclosed within each pressure element. Three velocity sub-elements occupy the corners of the parent pressure element, and have two corners on the adjacent midsides of the pressure element. A fourth velocity element occupies the remaining space in the central region of the pressure element. All the elements in this scheme have three nodes each, which are located at their corners. The LBB-stability of the scheme seems to derive from the ability of the midside velocity nodes to define the flow in and out of the sides of the pressure element (Gunzburger, 1989).

For the higher-order elements, quadratic-linear Taylor-Hood finite elements were chosen. These elements combine a quadratic velocity representation with a linear pressure representation. In this scheme, an equal number of velocity and pressure elements are collocated. The velocity elements each incorporate six nodes, distributed at the corners and midsides. This distribution supports a full quadratic polynomial velocity representation. Only three nodes are used in each pressure triangle, located at the corners. Thus the nodal representation within a given triangular element is identical to that within a pressure parent element in the linear scheme described above. Therefore it seems that the LBB-stability of the scheme may be explained in the same manner as for the linear-linear element combination above.

The elements chosen support a quadratic spatial description. This allows for curved-sided elements, and adds to the accuracy of solution of flows over curved surfaces.

2.2 Mesh Generation

Computational meshes were generated using another FORTRAN program coded by this researcher. It employs the advancing front method (Peraire et al. 1987, 1988, Elliott & Peraire 1997, Probert et al. 1991, Jin & Wiberg 1990) to generate anisotropic triangular elements that fill the computational domain.

Elements are sized and stretched according the principle of error equidistribution. It has been shown, for certain model problems, that if the spatial discretization error contribution of the mesh elements are equalized through the appropriate selection of element geometry, the global error is minimized (D'Azevedo & Simpson 1991, Babuska & Rheinboldt 1979, Remaki & Habashi 2009). The required size and shape of an element is determined from the local gradients that comprise the leading term in the truncation error as expressed by a Taylor series expansion of the solution within the element. The selected gradients comprise a gradient tensor which is referred to as the *error indicator*.

The program is novel in that it employs the multilinear singular value decomposition, in place of the eigenvalue decomposition, for the extraction of the principal components of the error indicator tensor. Thus, unlike the traditional approach, which is restricted to the use of the Hessian as an error indicator, the new method may be used with symmetric and non-symmetric gradient tensors of

any order. The solution gradients that are required for forming these gradient tensors are obtained from the solution using least-squares approximations to local data. The elements surrounding a particular node are identified, and their nodal values fitted. The gradient values are taken from the resulting polynomial representation, and then assembled into an unfolded matrix form of the desired gradient tensor. This will be described in detail in later sections.

The results of the SVD of this unfolded matrix form are then used to create a second order metric tensor. The required characteristic of this metric tensor is that its eigenvalues and eigenvectors be the same as the singular values and left singular vectors of the unfolded gradient tensor. If $A_{(n)}$ is the selected unfolded matrix representation of the gradient tensor **A**, the SVD of $A_{(n)}$ is given by

$$U\ S\ V^T = A_{(n)}$$

where U is the matrix of left singular vectors, S is a diagonal matrix of singular values, and V is the matrix of right singular vectors. If *d* is the number of space dimensions in the flow domain then U is of dimension *d* x *d*. As long as a proper unfolding was selected, the singular values are the *principal gradients* of the velocity field, and U contains the *principal directions* along which these gradients occur.

The required node spacing or element size *h* in each of the principal directions is obtained by considering the dominant error term left by the approximation scheme. Consider a Taylor series expansion for a flow field variable u along the *x*-direction:

$$u = u_0 + \frac{\partial u}{\partial x}\Delta x + \frac{\partial^2 u}{\partial x^2}\frac{\Delta x^2}{2} + \frac{\partial^3 u}{\partial x^3}\frac{\Delta x^3}{3!} + \ldots$$

If we set the *x*-direction along the line connecting two mesh nodes 1 and 2 at which the variable u is represented, and if we let *x* = 0 at node 1 and x = h at node 2, then at the midpoint between the nodes,

$$u_m = u_1 + \frac{\partial u}{\partial x}\frac{h}{2} + \frac{1}{2!}\frac{\partial^2 u}{\partial x^2}\left(\frac{h}{2}\right)^2 + \frac{1}{3!}\frac{\partial^3 u}{\partial x^3}\left(\frac{h}{2}\right)^3 + \ldots$$

$$u_m = u_1 + \frac{\partial u}{\partial x}\frac{h}{2} + \frac{\partial^2 u}{\partial x^2}\frac{h^2}{8} + \frac{\partial^3 u}{\partial x^3}\frac{h^3}{48} + \ldots$$

If the solution representation is linear, then the leading error term at the midpoint is

$$e = \frac{\partial^2 u}{\partial x^2}\frac{h^2}{8}$$

Solving for 1/*h*,

$$\frac{1}{h} = \left(\frac{1}{8e}\frac{\partial^2 u}{\partial x^2}\right)^{1/2}$$

Similarly, for a quadratic solution representation, the leading error term is cubic, and we have

$$e = \frac{\partial^3 u}{\partial x^3}\frac{h^3}{48}$$

$$\frac{1}{h} = \left(\frac{1}{48e} \frac{\partial^3 u}{\partial x^3} \right)^{1/3}$$

These formulas are used to compute the value of $1/h$ required in each principal direction to set the dominant error term equal to a spatial error tolerance e. A matrix representation of a metric tensor $\mathbf{H}^{-1}$ is constructed from the left singular vectors of the gradient tensor **A**, and a diagonal matrix D whose entries are the values of 1/h computed from the principal gradient values on the diagonal of S:

$$H^{-1} = U\, D\, U^T$$

Finally the matrix representation of the metric tensor **H** is obtained by inverting the metric tensor $\mathbf{H}^{-1}$. The metric tensor **H** is used as in conventional gradient-based mesh generation algorithms for the sizing and stretching of elements (Sahni *et al*. 2005, 2010, Frey & Alauzet 2005, Dompierre *et al*. 2002, Remaki & Habashi 2009, Castro-Diaz 1997, Borouchaki *et al.* 1997). For example, to size and stretch an equilateral triangle according to local mesh requirements, the position vector of each node may be simply premultiplied by the matrix H.

The method described above requires a flow solution from which the required gradients are obtained. To obtain an initial solution, a starting mesh is generated from purely geometric considerations. The main geometric requirement is that the flow geometry be adequately resolved by the starting mesh. An example of a starting mesh is shown in Figure 5a. Once a flow solution has been obtained from the starting mesh, a second mesh may be generated using the present method. This mesh will reflect the flow gradients, and can then be used to generate an

improved solution, and so on. As this cycle is repeated, and as the flow solution converges spatially, the mesh properties stabilize as well. An example of a final computational mesh is shown in Figure 5b.

In this work, fine-grid solutions were generated which approximate the exact solution as closely as possible considering available computing resources. These fine grid solutions were then used directly to calculate **H** and generate the new mesh, using the different error indicators. This was done to avoid the iterative procedure outlined above and to not unfairly bias the Hessian approach by requiring it to rely on a less accurate solution for refinement. The third velocity gradient tensor (3VGT) error indicator and quadratic-linear finite elements were used to generate the fine grid solutions. This was done because the spatial convergence thus afforded was found to be superior to that of the other methods examined. In some cases, fine grid solutions obtained using the Hessian were used to verify that the use of 3VGT-based fine-grid solutions did not bias the results. In some other cases, additional fine-grid solutions obtained with a slightly larger mesh tolerance were used to verify that the fine grid solutions used in the study were sufficiently fine. That is, the fine-grid solutions were sufficiently close to the exact solution to avoid any bias.

The remaining subsections are dedicated to familiarizing the reader with the essential aspects of the multilinear SVD that must be understood for its application to gradient-based mesh adaptation.

2.3 Matrix Representation of Velocity Gradient Tensors

When traditional matrix operations are to be used, such as multiplication of matrices and vectors, the velocity gradient tensor is usually written

$$\nabla \mathbf{u} = \begin{bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial z} \end{bmatrix} \begin{bmatrix} u & v & w \end{bmatrix} = \begin{bmatrix} \dfrac{\partial u}{\partial x} & \dfrac{\partial v}{\partial x} & \dfrac{\partial w}{\partial x} \\ \dfrac{\partial u}{\partial y} & \dfrac{\partial v}{\partial y} & \dfrac{\partial w}{\partial y} \\ \dfrac{\partial u}{\partial z} & \dfrac{\partial v}{\partial z} & \dfrac{\partial w}{\partial z} \end{bmatrix}$$

where u, v and w are the components of the velocity in the mutually orthogonal x, y and z directions. Here the row number in the matrix is associated with the direction of differentiation while the column number is associated with a particular velocity component. However, this tensor may also be expanded in a manner that is more consistent with the use of Einstein's indicial notation:

$$u_{i,j} = \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} \\ u_{2,1} & u_{2,2} & u_{2,3} \\ u_{3,1} & u_{3,2} & u_{3,3} \end{bmatrix}$$

The first index is associated with the components of the velocity vector, and the index following the comma indicates the direction of differentiation. When the tensor form is written out as a matrix, it is often represented as shown, with the first tensor index associated with the matrix row number. In terms of the ordered basis $\mathbf{X} = (x, y, z)$,

$$[u_{i,j}]_{\mathbf{X}} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{bmatrix}$$

If we take the gradient of the velocity gradient tensor, we obtain the second velocity gradient tensor. This is a tensor of third order, and may be written as *n* matrices stacked on one another, where *n* is the number of spatial dimensions represented. Here the presentation is restricted to two space dimensions for clarity:

$$u_{i,jk} = \begin{bmatrix} u_{1,11} & u_{1,21} \\ u_{2,11} & u_{2,21} \end{bmatrix} \begin{bmatrix} u_{1,12} & u_{1,22} \\ u_{2,12} & u_{2,22} \end{bmatrix}$$

The lower-left matrix in each matrix pair above is to be interpreted as being stacked above or "in front of" the upper right matrix, or nearer the reader's eye. The first index is the row number, the second index is the column number, and the third index refers to the level, with the front, or lower-left matrix being at level 1. In terms of the ordered basis $\mathbf{X} = (x, y, z)$,

$$[u_{i,jk}]_{\mathbf{X}} = \begin{bmatrix} u_{xx} & u_{yx} \\ v_{xx} & v_{yx} \end{bmatrix} \begin{bmatrix} u_{xy} & u_{yy} \\ v_{xy} & v_{yy} \end{bmatrix}$$

This third order tensor, or any other tensor of order three or greater, may be "unfolded" to obtain a matrix. In accordance with De Lathauwer, the three matrix unfoldings of a third order tensor are illustrated in Figure 6.

In this figure, the dimensions of the tensor are $I_1 = I_2 = I_3 = 4$. The nth matrix unfolding is referred to as the n-mode unfolding, and the number n is the tensor index that becomes the row number in the unfolded matrix. The column numbers are obtained by cycling through the remaining indices. For a matrix unfolding with dimension I_c x $I_a I_b$, the index i_a varies more slowly than i_b. If we unfold the second velocity gradient tensor accordingly, we obtain the three modes:

$$(u_{i,jk})_{(1)} = \left[\begin{array}{cc|cc} u_{1,11} & u_{1,12} & u_{1,21} & u_{1,22} \\ u_{2,11} & u_{2,12} & u_{2,21} & u_{2,22} \end{array}\right] \quad \text{or} \quad [(u_{i,jk})_{(1)}]_{\mathbf{X}} = \left[\begin{array}{cc|cc} u_{xx} & u_{xy} & u_{yx} & u_{yy} \\ v_{xx} & v_{xy} & v_{yx} & v_{yy} \end{array}\right]$$

$$(u_{i,jk})_{(2)} = \left[\begin{array}{cc|cc} u_{1,11} & u_{2,11} & u_{1,12} & u_{2,12} \\ u_{1,21} & u_{2,21} & u_{1,22} & u_{2,22} \end{array}\right] \quad \text{or} \quad [(u_{i,jk})_{(2)}]_{\mathbf{X}} = \left[\begin{array}{cc|cc} u_{xx} & v_{xx} & u_{xy} & v_{xy} \\ u_{yx} & v_{yx} & u_{yy} & v_{yy} \end{array}\right]$$

$$(u_{i,jk})_{(3)} = \left[\begin{array}{cc|cc} u_{1,11} & u_{1,21} & u_{2,11} & u_{2,21} \\ u_{1,12} & u_{1,22} & u_{2,12} & u_{2,22} \end{array}\right] \quad \text{or} \quad [(u_{i,jk})_{(3)}]_{\mathbf{X}} = \left[\begin{array}{cc|cc} u_{xx} & u_{yx} & v_{xx} & v_{yx} \\ u_{xy} & u_{yy} & v_{xy} & v_{yy} \end{array}\right]$$

Note that because $u_{xy} = u_{yx}$ and $v_{xy} = v_{yx}$, the 2-mode and 3-mode matrix unfoldings above consist of identical sets of column vectors, but which are ordered differently in each unfolding. In general, velocity gradient tensors, of any order, will give only two matrix unfoldings with distinct sets of column vectors.

The pattern of unfolding used above applies to higher order tensors as well. Let us consider the third velocity gradient tensor. The 1-mode unfolding is given by:

$$(u_{i,jkl})_{(1)} = \left[\begin{array}{cc|cc|cc|cc} u_{1,111} & u_{1,112} & u_{1,121} & u_{1,122} & u_{1,211} & u_{1,212} & u_{1,221} & u_{1,222} \\ u_{2,111} & u_{2,112} & u_{2,121} & u_{2,122} & u_{2,211} & u_{2,212} & u_{2,221} & u_{2,222} \end{array}\right]$$

or

$$[(u_{i,jkl})_{(1)}]_{\mathbf{X}} = \left[\begin{array}{cc|cc|cc|cc} \frac{\partial^3 u}{\partial x^3} & \frac{\partial^3 u}{\partial x^2 y} & \frac{\partial^3 u}{\partial x^2 y} & \frac{\partial^3 u}{\partial x y^2} & \frac{\partial^3 u}{\partial x^2 y} & \frac{\partial^3 u}{\partial x y^2} & \frac{\partial^3 u}{\partial x y^2} & \frac{\partial^3 u}{\partial y^3} \\ \frac{\partial^3 v}{\partial x^3} & \frac{\partial^3 v}{\partial x^2 y} & \frac{\partial^3 v}{\partial x^2 y} & \frac{\partial^3 v}{\partial x y^2} & \frac{\partial^3 v}{\partial x^2 y} & \frac{\partial^3 v}{\partial x y^2} & \frac{\partial^3 v}{\partial x y^2} & \frac{\partial^3 v}{\partial y^3} \end{array}\right]$$

As was the case with the second velocity gradient tensor, because of the symmetry of differentiation, the remaining modes of unfolding result in matrices which share the same column vectors, although the column vectors in each unfolding are ordered differently. The remaining unfoldings are listed below:

$$(u_{i,jkl})_{(2)} = \left[\begin{array}{cc|cc|cc|cc} u_{1,111} & u_{2,111} & u_{1,112} & u_{2,112} & u_{1,121} & u_{2,121} & u_{1,122} & u_{2,122} \\ u_{1,211} & u_{2,211} & u_{1,212} & u_{2,212} & u_{1,221} & u_{2,221} & u_{1,222} & u_{2,222} \end{array}\right]$$

$$(u_{i,jkl})_{(3)} = \left[\begin{array}{cc|cc|cc|cc} u_{1,111} & u_{1,211} & u_{2,111} & u_{2,211} & u_{1,112} & u_{1,212} & u_{2,112} & u_{2,212} \\ u_{1,121} & u_{1,221} & u_{2,121} & u_{2,221} & u_{1,122} & u_{1,222} & u_{2,122} & u_{2,222} \end{array}\right]$$

$$(u_{i,jkl})_{(4)} = \left[\begin{array}{cc|cc|cc|cc} u_{1,111} & u_{1,121} & u_{1,211} & u_{1,221} & u_{2,111} & u_{2,121} & u_{2,211} & u_{2,221} \\ u_{1,112} & u_{1,122} & u_{1,212} & u_{1,222} & u_{2,112} & u_{2,122} & u_{2,212} & u_{2,222} \end{array}\right]$$

2.4 Matrix representation of gradients of a scalar

The above matrix unfoldings have been applied to gradients of the velocity vector *u*. If we consider the gradients of a scalar quantity, considerable simplification results due to added symmetry. The first gradient of a scalar *u* is given by

$$u_{,i} = \begin{bmatrix} u_{,1} \\ u_{,2} \\ u_{,3} \end{bmatrix} \qquad \text{or} \qquad [u_{,i}]_{\mathbf{X}} = \nabla u = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} u = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}$$

Applying the gradient operator again, we obtain the second gradient or Hessian of the scalar *u*, which is a second order tensor:

$$u_{,ij} = \begin{bmatrix} u_{,11} & u_{,12} & u_{,13} \\ u_{,21} & u_{,22} & u_{,23} \\ u_{,31} & u_{,32} & u_{,33} \end{bmatrix}$$

or

$$[u_{,ij}]_{\mathbf{X}} = \nabla(\nabla u) = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 u}{\partial x^2} & \frac{\partial^2 u}{\partial x \partial y} & \frac{\partial^2 u}{\partial x \partial z} \\ \frac{\partial^2 u}{\partial y \partial x} & \frac{\partial^2 u}{\partial y^2} & \frac{\partial^2 u}{\partial y \partial z} \\ \frac{\partial^2 u}{\partial z \partial x} & \frac{\partial^2 u}{\partial z \partial y} & \frac{\partial^2 u}{\partial z^2} \end{bmatrix} = \begin{bmatrix} u_{xx} & u_{xy} & u_{xz} \\ u_{yx} & u_{yy} & u_{yz} \\ u_{zx} & u_{zy} & u_{zz} \end{bmatrix} = \mathbf{H}(u)$$

Applying the gradient operator a third time gives the third gradient of the scalar u. Here we must abandon the notation of vector and matrix multiplication. For clarity the two dimensional case is presented:

$$u_{,ijk} = \begin{bmatrix} u_{,111} & u_{,121} \\ u_{,211} & u_{,221} \end{bmatrix} \begin{bmatrix} u_{,112} & u_{,122} \\ u_{,212} & u_{,222} \end{bmatrix} \qquad \text{or} \qquad [u_{,ijk}]_{\mathbf{X}} = \begin{bmatrix} u_{xxx} & u_{xyx} \\ u_{yxx} & u_{yyx} \end{bmatrix} \begin{bmatrix} u_{xxy} & u_{xyy} \\ u_{yxy} & u_{yyy} \end{bmatrix}$$

Because of the symmetry of the cross derivatives, this tensor possesses pair-wise symmetry. This means that the tensor is symmetric with respect to any two

chosen indices. Due to this pair-wise symmetry, all three modes of matrix unfolding are identical:

$$(u_{,ijk})_{(1)} = \left[\begin{array}{cc|cc} u_{,111} & u_{,112} & u_{,121} & u_{,122} \\ u_{,211} & u_{,212} & u_{,221} & u_{,222} \end{array}\right]$$

$$(u_{,ijk})_{(2)} = \left[\begin{array}{cc|cc} u_{,111} & u_{,211} & u_{,112} & u_{,212} \\ u_{,121} & u_{,221} & u_{,122} & u_{,222} \end{array}\right]$$

$$(u_{,ijk})_{(3)} = \left[\begin{array}{cc|cc} u_{,111} & u_{,121} & u_{,211} & u_{,221} \\ u_{,112} & u_{,122} & u_{,212} & u_{,222} \end{array}\right]$$

or

$$[(u_{,ijk})_{(1)}]_{\mathbf{X}} = [(u_{,ijk})_{(2)}]_{\mathbf{X}} = [(u_{,ijk})_{(3)}]_{\mathbf{X}} = \left[\begin{array}{cc|cc} u_{xxx} & u_{xxy} & u_{xxy} & u_{xyy} \\ u_{xxy} & u_{xyy} & u_{xyy} & u_{yyy} \end{array}\right]$$

2.5 Eigenvalue Decomposition of the Hessian of a Scalar Field

Mesh adaptation for computational studies of fluid flow is commonly achieved by finding the principal values and directions of the second gradient, or Hessian, of a scalar field. Many papers and books printed within the last 5 to 10 years refer to this, without reservation, as the method of choice. For incompressible flows, the scalar quantity chosen is usually the magnitude of the local flow velocity, while for compressible flows, the local pressure or Mach number are often used. In three dimensions, the Hessian of a scalar quantity *u* is

$$\mathbf{H}(u) = \begin{bmatrix} \frac{\partial^2 u}{\partial x^2} & \frac{\partial^2 u}{\partial x \partial y} & \frac{\partial^2 u}{\partial x \partial z} \\ \frac{\partial^2 u}{\partial y \partial x} & \frac{\partial^2 u}{\partial y^2} & \frac{\partial^2 u}{\partial y \partial z} \\ \frac{\partial^2 u}{\partial z \partial x} & \frac{\partial^2 u}{\partial z \partial y} & \frac{\partial^2 u}{\partial z^2} \end{bmatrix} = \begin{bmatrix} u_{xx} & u_{xy} & u_{xz} \\ u_{yx} & u_{yy} & u_{yz} \\ u_{zx} & u_{zy} & u_{zz} \end{bmatrix}$$

Because the Hessian is real and symmetric, it possesses real eigenvalues and eigenvectors. These correspond to the principal values and directions. The principal directions are mutually orthogonal, and may be interpreted as the coordinate directions that would give the largest and smallest values of the local second derivative of the chosen scalar.

2.6 Singular Value Decomposition of the Velocity Gradient Tensor

Mesh adaptation for computational studies of fluid flow is commonly achieved In terms of the ordered basis $\mathbf{X} = (x, y, z)$, the velocity gradient tensor may be written in matrix form as

$$[u_{i,j}]_{\mathbf{X}} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{bmatrix}$$

where u, v and w are the velocity components in the x, y and z directions. This tensor is, in general, not symmetric, and is not, therefore, amenable to analysis using the eigenvalue decomposition, as was the Hessian. The eigenvalues and eigenvectors are, in general, complex, and do not necessarily represent principal

values and directions of the velocity gradient. However, work by this researcher indicates that the singular value decomposition may be used to obtain information that is equally useful in characterizing the gradients and in specifying computational mesh properties.

2.7 Multilinear (High Order) Singular Value Decomposition (HOSVD)

In recent years, a great deal of attention has been given to the development of methods for analyzing high-order tensors. This has led to the development of a generalization of the Singular value decomposition that is applicable to tensors of arbitrary order (De Lathauwer, 2000). This decomposition is referred to as the Multilinear or high-order singular value decomposition (HOSVD).

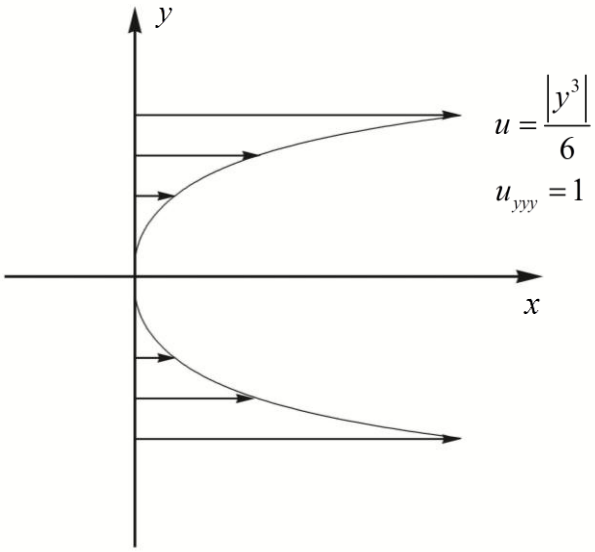
As was outlined above in the section on matrix representation of velocity gradient tensors, a tensor of order *n* may be unfolded in *n* different ways. The HOSVD is obtained by applying the singular value decomposition (SVD) to each of the *n* matrix unfoldings. The resulting *n* sets of singular values and left singular vectors are the singular values and singular vectors of the original tensor.

If a tensor **T** possesses symmetry with respect to certain index pairs, then certain matrix unfoldings will have identical column vectors, and therefore identical SVDs. In the case where **T** is symmetric with respect to all pairs of indices, it is referred to as being pair-wise symmetric. A pair-wise symmetric tensor has only one matrix unfolding with a distinct set of column vectors, and therefore, only one distinct set of singular values and singular vectors. In this

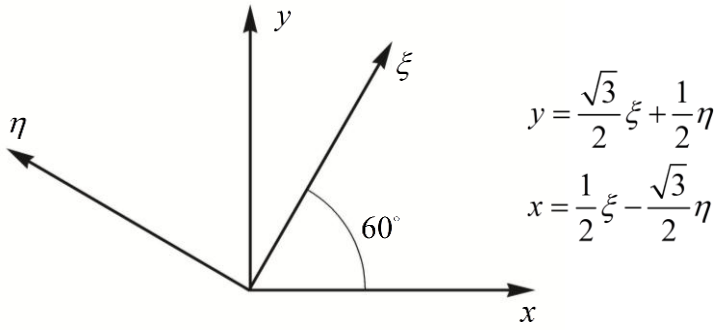
case, the HOSVD is analogous to the eigenvalue decomposition of a symmetric matrix.

Example 1. Application of the multilinear SVD to a symmetric third order tensor

Consider the following one dimensional flow:



Also consider the following coordinate transformation:



Considering positive values of y, the velocity u may be expressed in terms of the new coordinates as

$$
\begin{aligned}
u &= \frac{y^3}{6} \\
&= \frac{1}{6}\left(\frac{\sqrt{3}}{2}\xi + \frac{1}{2}\eta\right)^3 \\
&= \frac{1}{48}\left(3\sqrt{3}\xi^3 + 9\xi^2\eta + 3\sqrt{3}\xi\eta^2 + \eta^3\right)
\end{aligned}
$$

So

$$u_{\xi\xi\xi} = \frac{3\sqrt{3}}{8} = .6495$$
$$u_{\xi\xi\eta} = \frac{3}{8} = .3750$$
$$u_{\xi\eta\eta} = \frac{\sqrt{3}}{8} = .2165$$
$$u_{\eta\eta\eta} = \frac{1}{8} = .1250$$

The third gradient tensor associated with this velocity profile is

$$A = \begin{bmatrix} u_{\xi\xi\xi} & u_{\xi\xi\eta} \\ u_{\xi\xi\eta} & u_{\xi\eta\eta} \end{bmatrix} \begin{bmatrix} u_{\xi\xi\eta} & u_{\xi\eta\eta} \\ u_{\xi\eta\eta} & u_{\eta\eta\eta} \end{bmatrix}$$

Since we are considering only one component of the velocity, which is a scalar, the third gradient tensor is symmetric. Thus it is also pair-wise symmetric and, therefore, all three matrix unfoldings are identical:

$$(u_{,ijk})_{(1)} = (u_{,ijk})_{(2)} = (u_{,ijk})_{(3)} = \left[\begin{array}{cc|cc} u_{\xi\xi\xi} & u_{\xi\xi\eta} & u_{\xi\xi\eta} & u_{\xi\eta\eta} \\ u_{\xi\xi\eta} & u_{\xi\eta\eta} & u_{\xi\eta\eta} & u_{\eta\eta\eta} \end{array}\right] = \left[\begin{array}{cc|cc} .6495 & .3750 & .3750 & .2165 \\ .3750 & .2165 & .2165 & .1250 \end{array}\right]$$

The Singular values of this 2x8 matrix are 1 and 0. The singular vectors are

$$\begin{bmatrix} .8660 \\ .5000 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 \\ 1/2 \end{bmatrix}$$

And

$$\begin{bmatrix} -.5000 \\ .8660 \end{bmatrix} = \begin{bmatrix} -1/2 \\ \sqrt{3}/2 \end{bmatrix}$$

The first singular value is equal to $u_{yyy} = 1$. Inspection of the first singular vector reveals that it is in the y-direction. This result corresponds identically to the given flow. Thus we can conclude that for this model flow, the multilinear SVD correctly revealed the principal quantities of the third gradient.

2.8 Application of the Multilinear SVD to Velocity Gradient Tensors

To fully take advantage of information provided by a flow solution in adapting the computational mesh, it is necessary to account for the full vector nature of the velocity field. To do this, we must apply the multilinear SVD to a velocity gradient tensor of appropriate order. If we use linear finite elements, we must use the second velocity gradient tensor, which is a tensor of order three. For quadratic elements, the third velocity gradient tensor would be used, which is a fourth order tensor.

The major complication, however, in utilizing the full vector representation of the velocity, lies in the fact that the velocity gradient tensors, of all order, are, in general, non-symmetric. This means, as was mentioned above, that the matrix unfolding of a velocity gradient tensor of selected order is not unique. There are, in fact, *n* unfoldings for an *n*th order non-symmetric tensor, and each of these unfoldings gives, in general, a different set of singular values and singular vectors. The situation is analogous to the case of a non-symmetric matrix, which possesses both left and right eigenvectors, and therefore does not possess a unique set of principal directions.

Examination of the properties of certain non-symmetric, third-order tensors seems to reveal, however, that despite the additional complexity of having multiple sets of singular values and singular vectors, these sets truly reveal the character of the tensors in a meaningful and useful way. Just as the SVD of a non-symmetric matrix reveals a dual nature through the left and right singular vectors, the Multilinear SVD reveals a multiplicity of personalities for a given

high-order tensor that show themselves only when the tensor is viewed through the proper matrix unfolding.

Velocity gradient tensors are symmetric in all indices except one. This is because of the symmetry of cross-differentiation. The asymmetry of these tensors is due to the components of the vector velocity, which are essentially independent. Again, because of this partial symmetry, only two distinct sets of column vectors result from the complete set of n matrix unfoldings, where n is the order of the velocity gradient tensor. In Figure 6, if I_1 corresponds to the leading index, which in Einstein's notation corresponds to the vector component, then the unfolding $\mathbf{A}_{(1)}$ gives a set of column vectors that is different from all the remaining unfoldings. The remaining unfoldings all share the same set of column vectors, and the same SVD. These remaining unfoldings contain the information that is required for determining the desired mesh properties in a computational scheme. That is, the SVD of these unfoldings gives left singular vectors that indicate the principal directions of movement through the physical space that result in maximal and minimal rates of change of the vector velocity. The first unfolding $\mathbf{A}_{(1)}$ appears to reveal specific information about exactly how the vector velocity changes.

Note that for a second order tensor, the first unfolding is simply the matrix representation of the tensor, and the second unfolding is its transpose. When a matrix is transposed, its left and right singular vectors switch sides. Thus, in the case of a second order tensor, the right singular vectors of its matrix

representation are identical to the left singular vectors of its second unfolding, and therefore may be used equally well for mesh generation.

Example 2: Multilinear SVD of a non-symmetric velocity gradient tensor

Let $u_{xx} = v_{xx} = 1$ where u and v are the Cartesian components of the velocity vector $\boldsymbol{u}$.

We may then represent the second velocity gradient tensor in a stacked matrix form as

$$\overset{\overset{I_3}{\nearrow}}{I_1\begin{bmatrix} u_{xx} & u_{yx} \\ v_{xx} & v_{yx} \end{bmatrix}}_{I_2}\begin{bmatrix} u_{xy} & u_{yy} \\ v_{xy} & v_{yy} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The three modes of unfolding according to De Lathauwer are then

Mode 1

$$I_1\overbrace{\left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{array}\right]}^{I_3}_{I_2}$$

Mode 2

$$I_2\overbrace{\left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]}^{I_1}_{I_3}$$

Mode 3

$$I_3\overbrace{\left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]}^{I_2}_{I_1}$$

The MATLAB singular value decomposition routine SVD is accessed using the following format:

[u, s, v] = SVD (A)

Where the left singular vectors are contained in the matrix u, the singular values are contained in the matrix s, and the right singular vectors are contained in the matrix v. Applying the MATLAB SVD to this example we obtain for the three modes

Mode 1

$$u = \begin{bmatrix} \begin{pmatrix} .707 \\ .707 \end{pmatrix} \begin{pmatrix} -.707 \\ .707 \end{pmatrix} \end{bmatrix} \qquad s = \begin{bmatrix} (1.414) & 0 & 0 & 0 \\ 0 & (0) & 0 & 0 \end{bmatrix}$$

Mode 2

$$u = \begin{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{bmatrix} \qquad s = \begin{bmatrix} (1.414) & 0 & 0 & 0 \\ 0 & (0) & 0 & 0 \end{bmatrix}$$

Mode 3

$$u = \begin{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{bmatrix} \qquad s = \begin{bmatrix} (1.414) & 0 & 0 & 0 \\ 0 & (0) & 0 & 0 \end{bmatrix}$$

Only the singular values and *left* singular vectors contain information that is relevant for our purposes. Thus the right singular vectors are not shown.

As can be seen, the results for the second and third modes are identical. This is because of the symmetries of differentiation within the second velocity gradient tensor. In fact, the only asymmetry in that tensor is associated with the first index, which specifies the x or y component of the velocity vector.

3. RESULTS

An Investigation of the mathematical character of the multilinear SVD was conducted in the context of potential application to metric tensors used in adaptive grid generation for CFD. Test cases revealed that the multilinear SVD provides the principal quantities of the high order gradient tensors of interest, and the means of obtaining and interpreting them.

The multilinear SVD was applied to adaptive unstructured computational mesh generation for solution of the Navier-Stokes equations within arbitrary two-dimensional domains. The resulting computational meshes were compared with those obtained using a traditional method (the Hessian approach).

Results indicate that use of a third gradient tensor in place of the Hessian (second gradient) results in more regular and appropriately sized elements within boundary layers, especially near the surface. Also, use of a gradient of the velocity vector in lieu of a scalar gradient appears to show additional benefits within boundary layers, as well as in every other flow region. The combination of these factors appears to result in a dramatic improvement in mesh quality, particularly near solid surfaces.

Drag values computed on a circular cylinder and a NACA 0012 airfoil indicate that the use of the third velocity gradient tensor in place of the Hessian gives results that are significantly more accurate. Thus fewer elements or degrees of freedom are required with the higher-order error indicator. These results are covered in more detail below.

3.1 Mathematical Results

A tensor of order *n* may be unfolded *n* different ways to obtain two dimensional matrix representations in accordance with the method of De Lathauwer. In the case of velocity gradient tensors, which are symmetric about all indices but one, all but one of the matrix unfoldings contain the same column vectors of length n, and give rise to the same principal components (singular values and vectors).

It was found by experimentation that the matrix unfoldings that share column vectors are the ones whose singular values and singular vectors contain the information needed to adapt the computational mesh in accordance with the error indicator associated with the selected velocity gradient.

Preliminary results seem to indicate that the SVD of the remaining matrix unfolding provides information about how the velocity vector changes with movement through the velocity field along the direction that gives the maximum rate of change in the vector velocity. This information does not appear to be useful for mesh adaptation at this time.

3.2 Qualitative Results

A FORTRAN program was written and tested that utilizes the multilinear SVD to automatically generate unstructured finite element meshes for solution of the incompressible Navier-Stokes equations on arbitrary two-dimensional domains. This program is used iteratively, in conjunction with a finite-element solver, to concurrently adapt the mesh and resolve the flow solution. The program was used

to investigate the characteristics of meshes obtained using four different error indicators: the Hessian, which is a second order tensor; the second velocity gradient tensor, which is a third order tensor; the second gradient of the magnitude of the velocity, which is also a third order tensor; and the third velocity gradient tensor, which is a fourth order tensor.

Comparison of meshes obtained using gradients of the magnitude of the velocity with those obtained using gradients of the vector velocity showed significant differences. These differences are illustrated in Figure 7 through Figure 9.

Comparison of the meshes in Figure 7 reveals that, everywhere outside the boundary layer, the elements obtained using the Hessian are larger than those obtained using the second velocity gradient tensor. This result is to be expected, considering that the Hessian, which is a gradient of a scalar, contains no vector information and is thus insensitive to flow curvature. Thus elements that are inappropriately large appear to result. An abrupt transition in elements size and stretching at the edge of the boundary layer is also evident in the mesh obtained using the Hessian. This abrupt change appears to result in irregularities in element geometry. The transition is visibly smoother in the mesh obtained with the second velocity gradient tensor, and element geometries at the transition appear more ordered.

Meshes generated using scalar gradients sometimes display irregularities at locations within a boundary layer and adjacent to a solid surface. One such irregularity is displayed in Figure 8. In this figure the top three meshes were

generated using the third gradient of a scalar quantity, the magnitude of the velocity. Each of these three meshes where generated using a different error tolerance. The top mesh was generated using the tightest error tolerance, and is the most refined. This mesh shows no irregularity. The two meshes below it were generated with larger error tolerances and show an increasing degree of irregularity with increasing error tolerance. The three bottom meshes in this figure were generated using the third velocity gradient tensor. These meshes show no irregularity. A similar pattern was observed in other test cases. This appears to indicate that the inclusion of vector information in the error indicator contributes to the stability of the mesh.

Figure 9 illustrates a dramatic difference in the way that scalar gradient and vector gradient error indicators handle regions of flow reversal. The upper mesh was generated using the third gradient of the magnitude of the velocity. This mesh displays a V-shaped region of massive refinement downstream of the airfoil. The lower mesh was generated using the third gradient of the vector velocity, and shows no such over refinement. The region of over refinement in the upper mesh is due to the inability of the scalar gradient to account for the vector nature of the flow, thus interpreting a region of zero velocity as a singularity. The over refinement seen is unnecessary and wasteful of computing resources.

Comparison of meshes obtained using gradients of different orders reveals significant differences as well. In particular, the use of third gradients rather than second gradients as error indicators results in boundary layer meshes with a larger percentage of the elements near the wall. This should result in more accurate

surface stress calculations. This effect is readily apparent in Figure 10. It is also apparent to a lesser degree in Figure 7.

3.3 Quantitative Results

Benchmark problems were solved to check the validity of flow computations, and to assess the effectiveness of the new method. The steady flow around a circular cylinder in a uniform flow field was solved at Reynolds numbers of 20, 40 and 100. Also, the steady flow around a NACA 0012 airfoil was solved at a Reynolds number of 10,000, and at angles of attack from zero to three degrees. These flows were solved using computational meshes that were generated using three different error indicators: the Hessian of the magnitude of the velocity, the second velocity gradient tensor (2VGT), and the third velocity gradient tensor (3VGT).

The 3VGT error indicator was applied in conjunction with finite elements incorporating quadratic interpolation functions for the velocity field, and linear interpolation functions for the pressure. The 2VGT and Hessian error indicators were implemented with linear interpolation for both the velocity and pressure fields.

The circular cylinder flows were compared with cases from the literature to assess the accuracy of drag calculations produced herein. Both the circular cylinder and NACA 0012 cases were used as a basis for comparison of the computational efficiency associated with the use of the three selected error indicators.

3.3.1 Circular Cylinder Results

The results of drag calculations for the circular cylinder are shown in Figure 11 through Figure 19. Computational results obtained herein are shown as data points connected by spline fits. In Figure 11 through Figure 16, drag values are displayed as a function of the number of variables, or *degrees of freedom*, used in the computations. The curves shown are thus a convergence history with respect to system size or computational effort. In the case of the total drag calculations shown in Figure 14 through Figure 16, drag values from the literature are shown as dashed horizontal lines for reference. The total drag values from the literature and obtained herein using the 3VGT error indicator are also listed in Table 1. In all cases, the most recent listed values from the literature (Fornberg, 1980, 1985 and 1991), and the converged values obtained herein show close agreement. The largest difference is seen at a Reynolds number of 100, that difference being approximately 0.12%.

Log plots of estimated error versus a mesh size parameter proportional to element edge length are shown for the circular cylinder in Figure 17 through Figure 19. In these plots, the slope of a linear approximation to a collection of data points would be approximately equal to the spatial order of accuracy of the method employed. Reference lines are shown in the plots for comparison. Note that, at a Reynolds number of 40, results are given from fine grid solutions (FGS's) obtained with the Hessian, as well as the third velocity gradient tensor, (3VGT) to show that the FGS used does not bias the results.

Comparison of the results obtained using the different error indicators shows that surface friction forces computed with the 3VGT with quadratic-linear elements are more accurate than those computed with the 2VGT and Hessian error indicators with linear-linear elements by at least one order of accuracy. The 3VGT-quadratic-linear combination appears to be at least second-order accurate, while the Hessian-linear-linear and 2VGT-linear-linear combinations appear to be approximately first-order accurate. This result exceeds the single order improvement expectation based on the Taylor series truncation errors associated with the methods.

Surface pressure forces also appear to be computed more accurately using the 3VGT error indicator, although the improvement appears to be in the form of a shift on the log plots, rather than an order of accuracy improvement, which would be evidenced by a difference in slope..

Regression of the numerical data was not performed because of the relatively small number of data points gathered, and because of the need for intelligent interpretation of the data. For example, round off errors were observed to limit the accuracy of solutions. The most accurate and rapidly convergent methods were observed to "hit the wall" when spatial discretization relative errors had been reduced to around 10^{-5}. This was apparently a result of the limitations of the fully-implicit frontal solver algorithm on the large, ill-conditioned system matrices that routinely result from fine discretizations of the Navier-Stokes equations, and does not reflect a characteristic of the method investigated here.

It is also important to note that a small indicated error does not necessarily indicate convergence, because the errors often "crossed the axis", creating very small apparent errors. Thus it is best to look primarily at the peaks in the error plots, rather than the valleys, for rate-of-convergence information.

Order of convergence does not tell the whole story with regard to the accuracy of a method. We must look also at the computational effort required to achieve a specific error. Table 2 through Table 4 show the number of degrees of freedom required to solve the cylinder flow and achieve estimated relative errors of 1%, 0.25% or 0.1%, depending on the information available from the results. The availability of comparison data was limited primarily by the poor accuracy and spatial convergence of the Hessian-linear-linear combination.

The estimated errors were obtained using the converged solutions from the 3VGT-quadratic-linear method as the "correct" solution. This was done because of the extremely rapid spatial convergence of the method, the inability of the Hessian-linear-linear and 2VGT-linear-linear methods to achieve satisfactory convergence with respect to the friction drag, and the close agreement between the chosen results and the most recent ones from the literature.

Information in these tables indicates a dramatic reduction in the number of degrees of freedom required when the allowable errors are small. For example, with an error tolerance of 0.25% on the total drag force, the number of degrees of freedom required with the 3VGT indicator and Quadratic-Linear elements is roughly *one twentieth* the number required with the conventional method. The

demand for a tighter tolerance would further increase the savings with the new method.

3.3.2 NACA 0012 Airfoil Results

The results of drag calculations for the NACA 0012 airfoil at Re=10,000 are shown in Figure 20 through Figure 27. These results indicate that the improvements seen with the 3VGT error indicator in the low-Reynolds number cylinder cases extend to higher Reynolds numbers as well. As with the cylinder cases, use of the 3VGT indicator results in increase of at least one order of accuracy in the computation of surface friction drag. A modest improvement in pressure drag accuracy is seen, also in accordance with the cylinder results.

A significant difference is seen between the NACA 0012 and cylinder results with respect to the 2VGT indicator, however. The NACA 0012 results indicate that the 2VGT indicator gives pressure forces that are significantly more accurate than those obtained with the Hessian. In fact, the improvement appears to be comparable to the shift seen with the 3VGT indicator. It is not known whether this selective improvement is associated with the difference in geometry, or with the much higher Reynolds number chosen for the NACA 0012 cases. Further research will be required to assess this.

Note that, at zero angle of attack, results are given from fine grid solutions (FGS's) obtained with two different tolerances to show that the FGS's are sufficiently accurate to avoid biasing the results.

3.3.3 Computational Cost

The additional computational effort associated with the application of the new method was almost negligible compared with that for solution of the flow. Computation of the required derivatives and application of the SVD to the small matrices associated with the velocity gradient tensors accounted for less than 2% of the total effort in all cases. This cost should be roughly proportional to the number of degrees of freedom, since the size of the local systems does not grow with problem size.

Since, in the case of the Navier-Stokes equations, the effort or cost of solution of the global system of flow equations often increases as a power of two or more applied to the number of degrees of freedom, the potential for savings in computational effort, with the new method, appears substantial.

4. CONCLUSION

The problem of extending solution-adaptive techniques for the analysis of low-speed fluid flows to higher-order accurate approximations was studied. The motivation for this work was the large potential benefit in terms of computational efficiency that might thus be afforded. To allow this transition, a higher-order approach for the generation of computational meshes was needed. In particular, a means of extracting principal gradients from tensors of order greater than two was desired. This would allow extension of the traditional Hessian approach to higher orders.

Methods of higher-order tensor analysis were studied, including the multilinear singular value decomposition (SVD). An investigation of the mathematical character of this decomposition was conducted by applying it to simple hypothetical velocity fields. This showed that the relevant principal components of high-order velocity and speed gradient tensors could be extracted using the decomposition. It also revealed the proper unfolding of the gradient tensors that is required to extract them.

Computational meshes were generated by applying the decomposition to higher-order gradients of computed flow solutions. The gradients examined were the Hessian, which is a second order tensor; the second velocity gradient tensor, which is a third order tensor; the second gradient of the magnitude of the velocity, which is also a third order tensor; and the third velocity gradient tensor, which is a fourth order tensor. These gradients were used as local error indicators for the sizing and stretching of triangular elements used for decomposing the flow domain.

Study of the meshes thus generated revealed significant differences that depended on the gradient tensor used. It was seen that gradients of the magnitude of the velocity, which is a scalar quantity, gave elements in inviscid regions that are significantly larger than those obtained using full velocity gradient tensors. The difference appears to be due to the fact that important information is lost in taking the magnitude of the velocity. For example, in the case of second gradients, the flow curvature information is lost. This can be verified by studying the gradients present in simple potential flows.

The loss of vector information when scalar gradients are used displayed itself in other ways. Regions of grid instability resulting in clumps of overly-large elements appeared in the boundary layer beneath a NACA 0012 airfoil at non-zero angles of attack. This would be expected to impact the accuracy of surface force calculations. Also, over-refinement resulted in the vicinity of near-singularities in the gradient field associated with eddies. This would impact the efficiency of computation by wasting resources on the resulting regions of high mesh density.

Because the conventional approach utilizes the Hessian, which is the second gradient of the magnitude of the velocity, it is subject to these limitations. The ability to utilize gradients of tensors of order higher than two, which is afforded by the multilinear SVD, permits the use of second and higher gradients of the full vector representation of the velocity field as error indicators. Thus the above problems can be effectively solved.

Comparison of meshes around the NACA 0012 airfoil also revealed that the use of third gradients rather than second gradients of the velocity or its magnitude results in meshes with a larger density near solid surfaces. This should have a favorable impact on the accuracy of surface force calculations.

Flow solutions were obtained from meshes generated using the Hessian approach, and from meshes generated using higher order gradients. The use of gradients of order higher than two, in conjunction with higher order finite elements as appropriate, is referred to as the new method.

Flow solutions thus obtained with the new method, and in particular with the third velocity gradient tensor (3VGT), displayed convergence, with respect to spatial discretization, that is more rapid than that with the traditional Hessian approach. Friction forces computed on a circular cylinder and a NACA 0012 airfoil displayed convergence of at least one additional spatial order of accuracy.

The absolute accuracy of both friction and pressure forces was observed to improve as well, using the new method with the 3VGT error indicator. This means that fewer elements, and thus fewer flow variables (degrees of freedom), are required for a given accuracy with the new method.

Flow solutions obtained with the second velocity gradient tensor gave mixed results. A significant benefit was observed in the NACA 0012 test cases, but not in the cylinder flows. It is not known whether the benefit is dependent on geometry or Reynolds number. Further research will be required to assess this. The benefit, when present, however, is apparently due to the improvement in mesh quality associated with the inclusion of the flow curvature information in the error indicator.

Indications are that the reduction in computational effort promised by the new method is substantial, particularly when higher-order approximations are used. In addition, the accuracy of the new method offers increased flexibility and simplicity of implementation, by allowing the use of unstructured surface meshes even when highly accurate surface stress calculations are required. This should aid in the automation of the mesh generation process.

The new method, which is gradient-based, has demonstrated high-order anisotropic capability. The adjoint method has demonstrated high-order capability as well, but on isotropic meshes. A combination of the methods could prove most effective, combining the efficiencies of high orders of approximation, mesh anisotropy and mesh optimization.

Circular Cylinder Drag Coefficient			
	Re=20	Re=40	Re=100
Dennis & Chang (1970)	2.045	1.522	1.056
Dennis (1976)	1.998	1.494	———
Fornberg (1980)	2.0001	1.4980	1.058
Fornberg (1985)	———	———	1.060
Fornberg (1991)	———	———	1.060
Present Work	2.0009	1.4988	1.0587

Table 1. Computed drag coefficient on a circular cylinder from several sources including the present work.

Degrees of Freedom Required				
C_{d_f} Tolerance	Re	Hessian with Linear-Linear Elements	2VGT with Linear-Linear Elements	3VGT with Quadratic-Linear Elements
1%	20	44,000	75,000	2640
	40	61,600	32,000	3310
	100	42,600	35,000	2380

Table 2. Approximate number of degrees of freedom (variables) required to achieve a prescribed level of estimated error in the computation of the friction drag coefficient on a circular cylinder.

Degrees of Freedom Required				
C_{d_p} Tolerance	Re	Hessian with Linear-Linear Elements	2VGT with Linear-Linear Elements	3VGT with Quadratic-Linear Elements
1%	20	2870	2600	2110
	40	2490	3200	1980
	100	3300	2800	2370
0.25%	20	6080	9000	3060
	40	7590	13,000	2750
	100	5790	8000	3450
0.1%	20	46,100	10,700	3630
	40	17,200	16,000	3070
	100	11,200	25,000	5720

Table 3. Approximate number of degrees of freedom (variables) required to achieve prescribed levels of estimated error in the computation of the pressure drag coefficient on a circular cylinder.

Degrees of Freedom Required					
C_d Tolerance	Re	Hessian with Linear-Linear Elements	2VGT with Linear-Linear Elements	3VGT with Quadratic-Linear Elements	Ratio of Hessian to 3VGT
1%	20	7230	5100	2090	3.5
	40	4900	7000	2070	2.4
	100	4560	7800	2390	1.9
0.25%	20	68,700	95,000	3210	21.4
	40	92,500	44,000	3580	25.8
	100	56,700	14,000	2710	20.9

Table 4. Approximate number of degrees of freedom (variables) required to achieve prescribed levels of estimated error in the computation of the total drag coefficient on a circular cylinder.

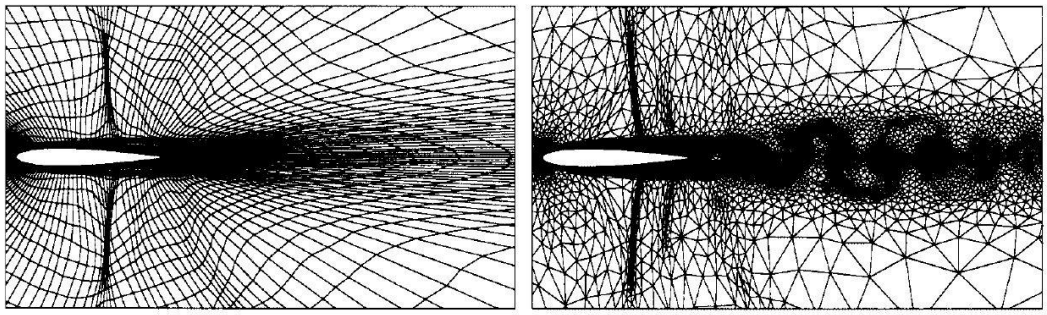


Figure 1. Structured and unstructured meshes for computing unsteady, viscous, compressible flow over a NACA 0012 airfoil at M=.85, Re=5000 (Ait-Ali-Yahia, 2002)

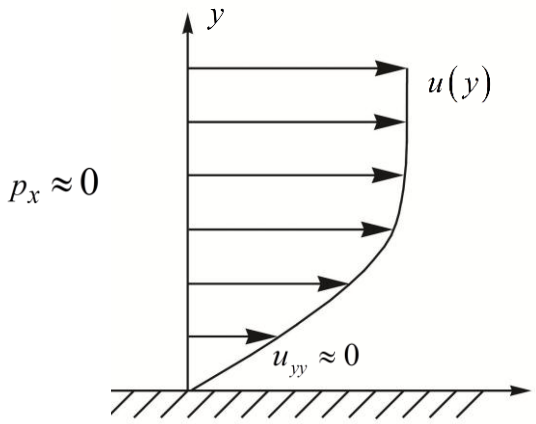


Figure 2. Boundary layer flow. When the streamwise pressure gradient is small, large elements result using the Hessian

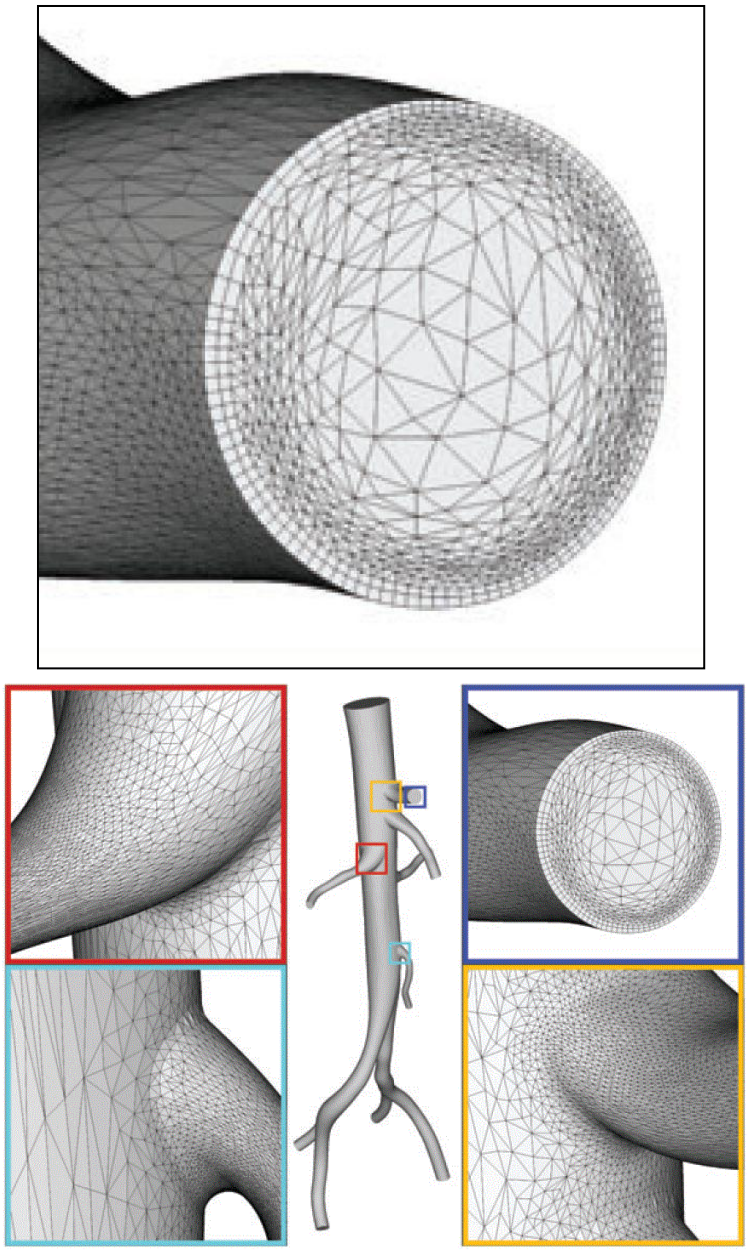


Figure 3. Discretization of the abdominal aorta. Structured quadrilateral layers of elements are visible near the wall. (Taylor, 2009)

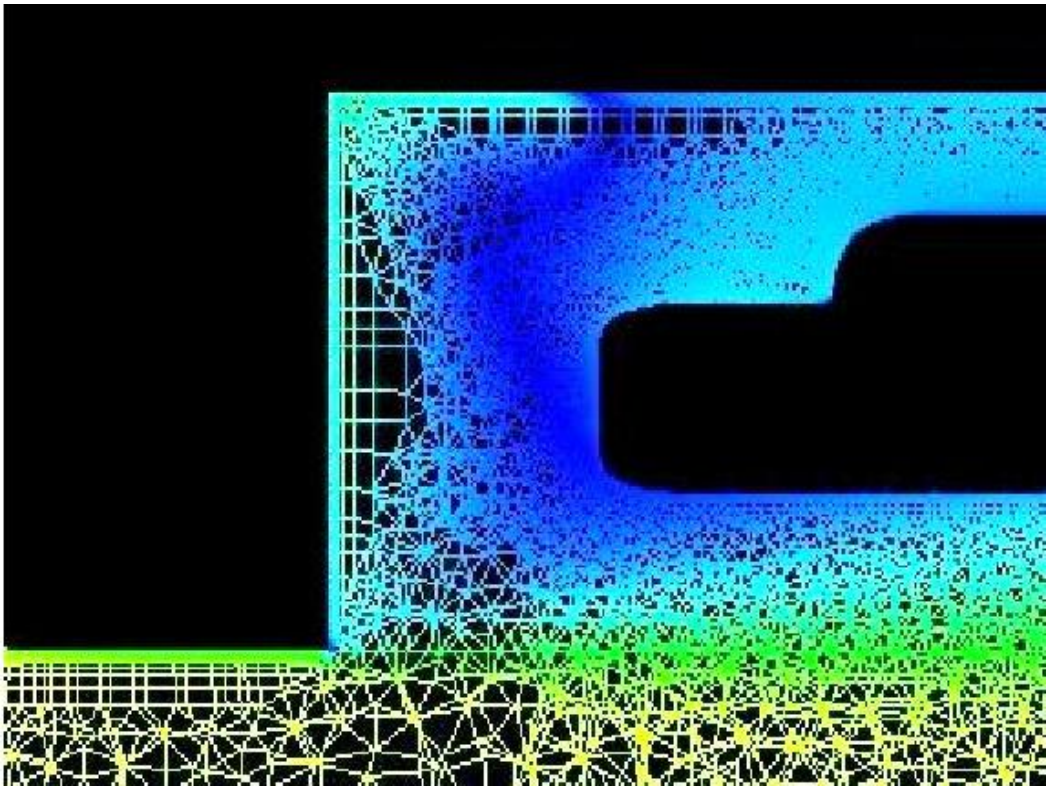
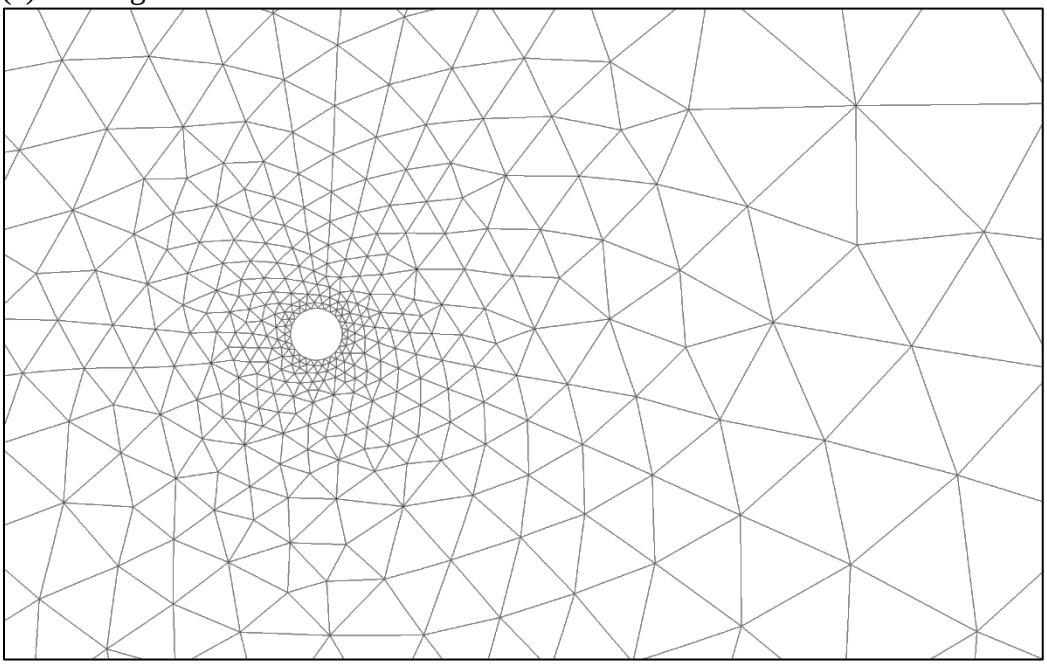


Figure 4. Torpedo in bay of P-3 Orion. Structured layers of quadrilateral elements are visible on the bay surfaces and on the lower surface of the Torpedo (Pointwise, Inc.)

(a) Starting mesh



(b) Final mesh

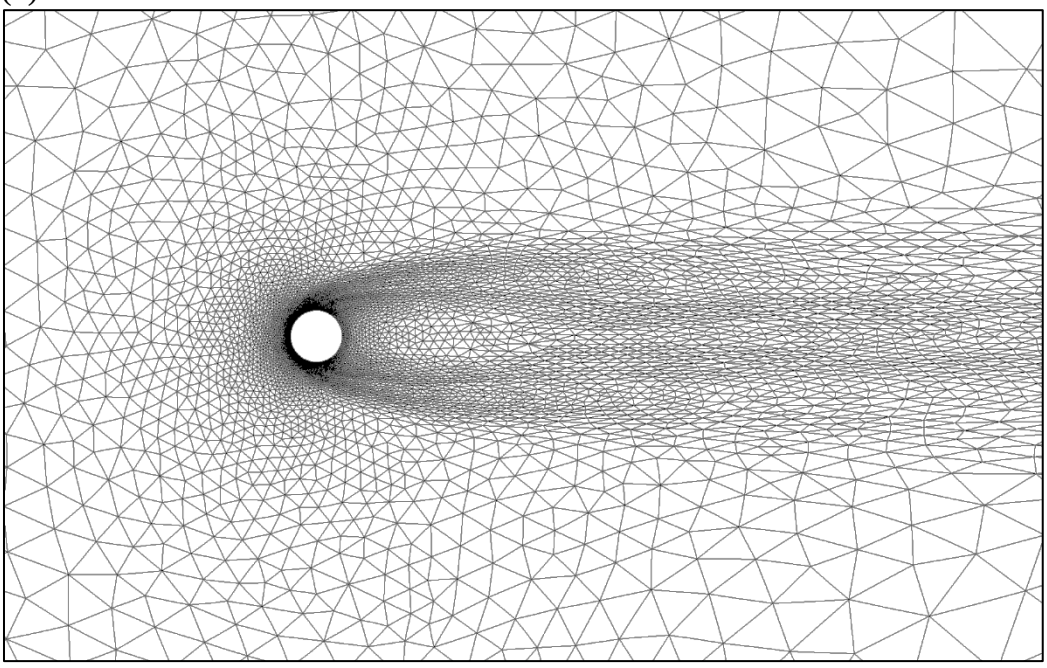


Figure 5. Starting and final meshes for steady circular cylinder flow at Re=100. The elemental error tolerance used for the final mesh was 0.00005.

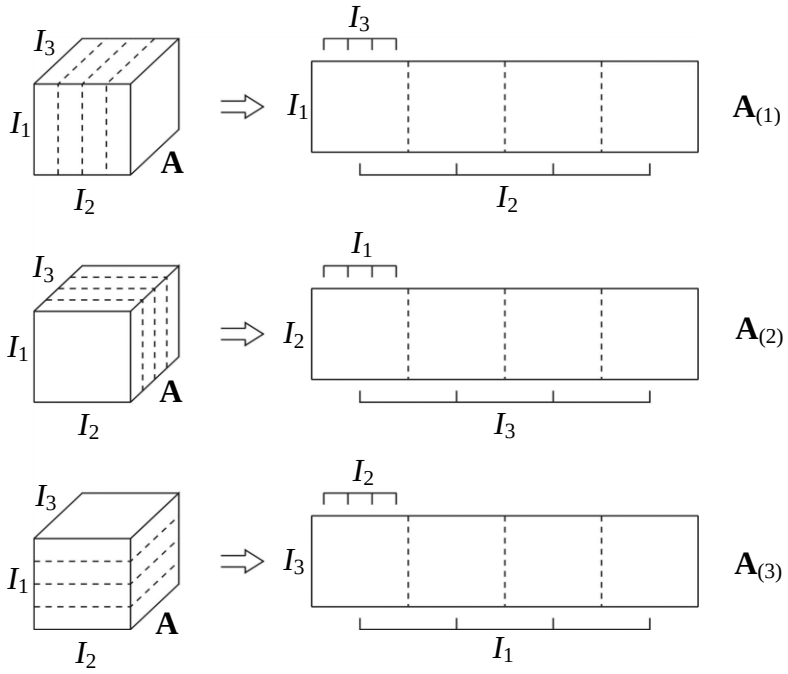
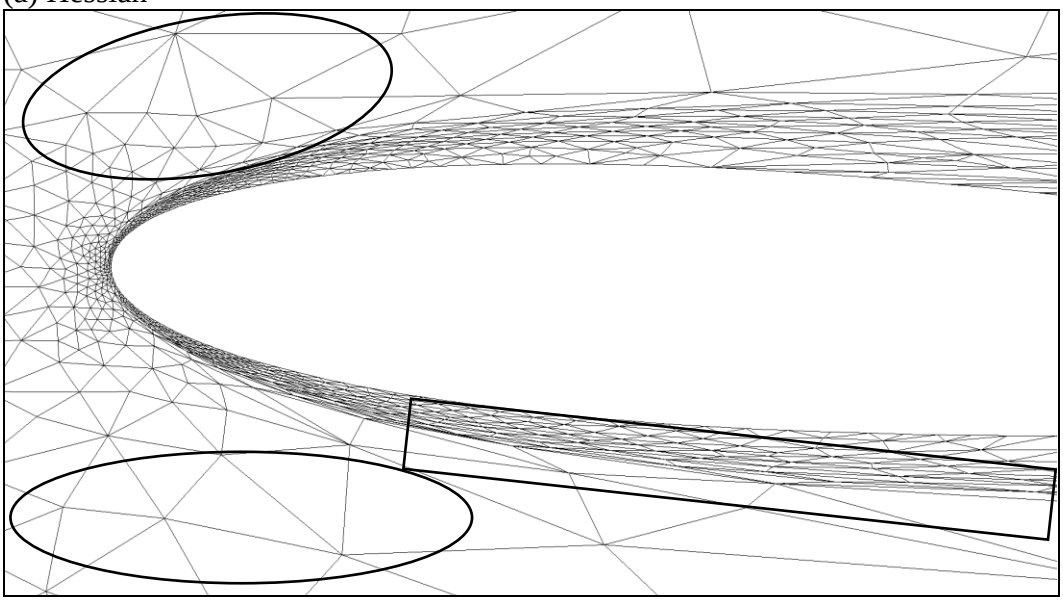


Figure 6. Matrix unfoldings of a third order tensor. From De Lathauwer.

(a) Hessian



(b) Second velocity gradient tensor

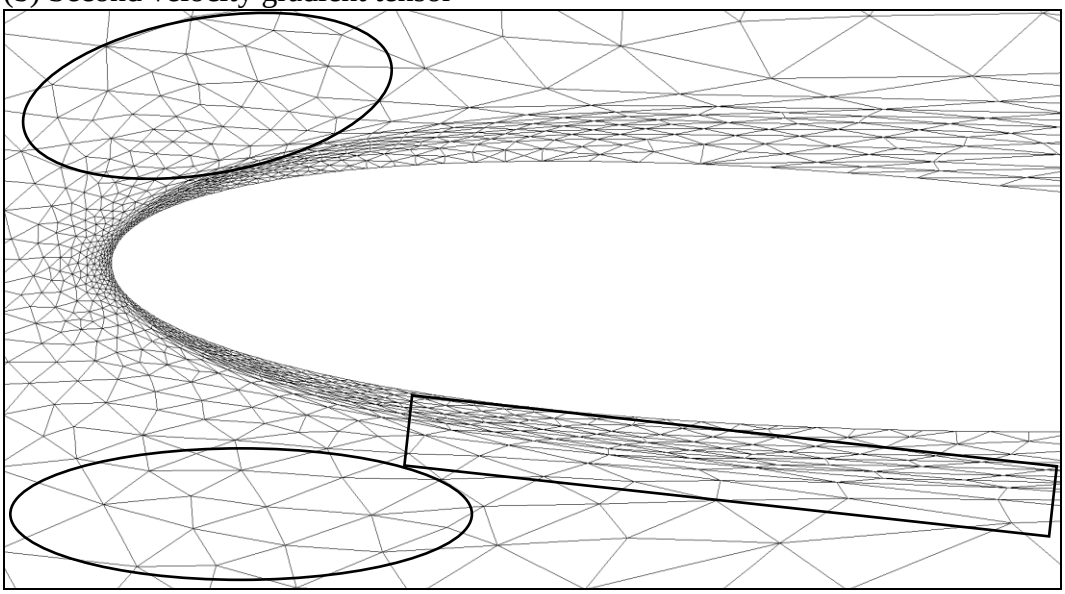
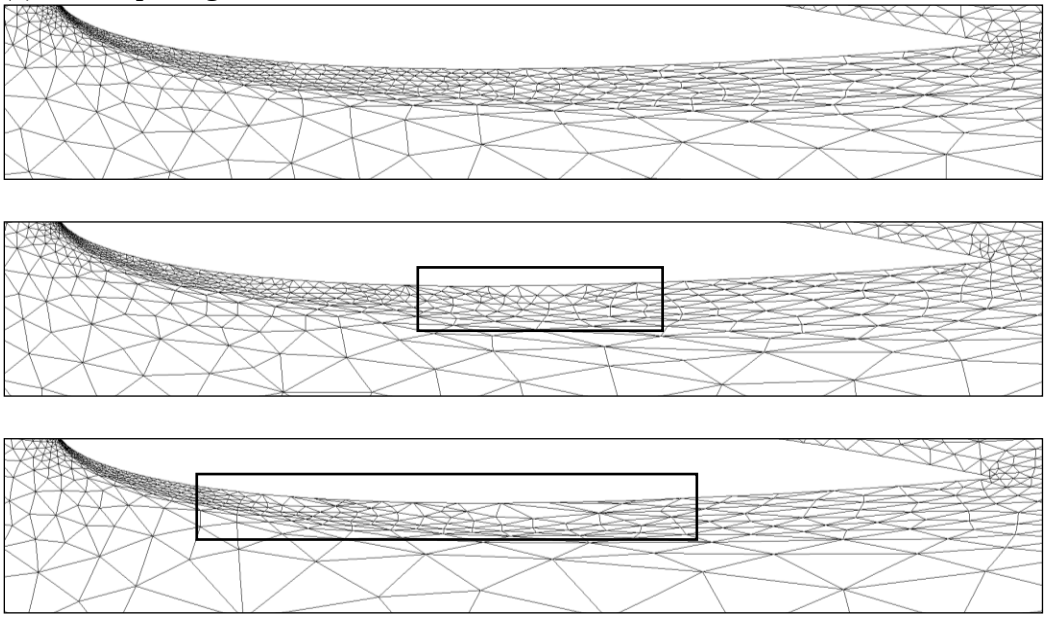


Figure 7. Scalar vs. vector gradients as error indicators in inviscid regions. Both meshes shown are adapted to the flow around a NACA 0012 airfoil at Re = 10,000 and $\alpha = 3°$. The upper mesh was obtained using the Hessian approach. An abrupt transition in element size appears at the edge of the boundary layer. This is particularly visible below the lower surface of the airfoil. The lower mesh was obtained using the second velocity gradient tensor. This mesh shows a much smoother transition between the viscous boundary layer and the essentially inviscid region beyond. The difference is due to the inability of the Hessian to detect flow curvature, which results in over-sized elements in inviscid regions such as those enclosed by the ellipses in the figure.

(a) Third speed gradient tensor



(b) Third velocity gradient tensor

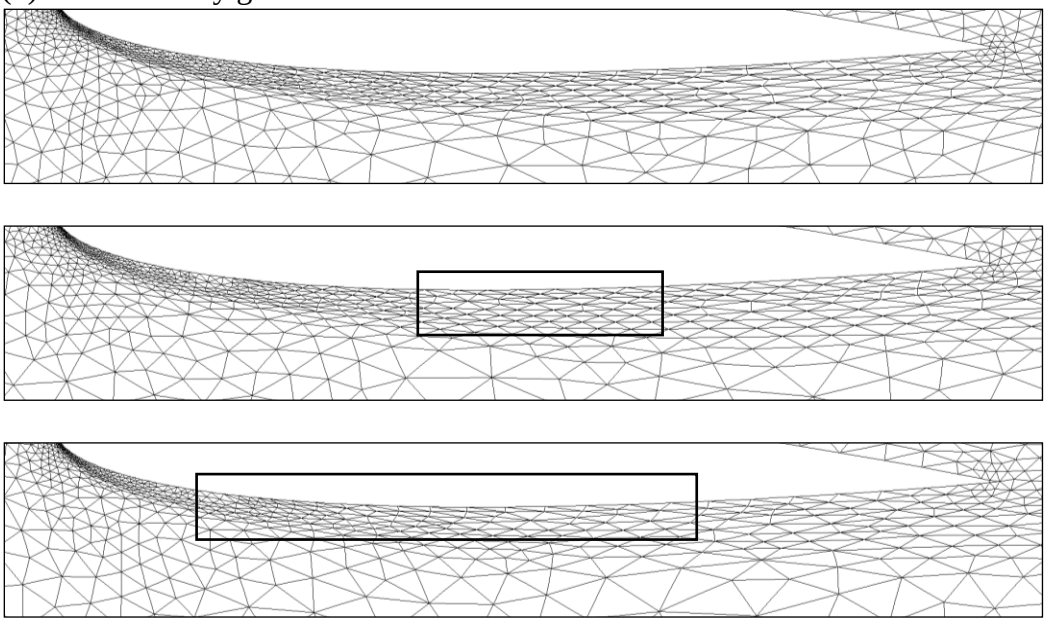
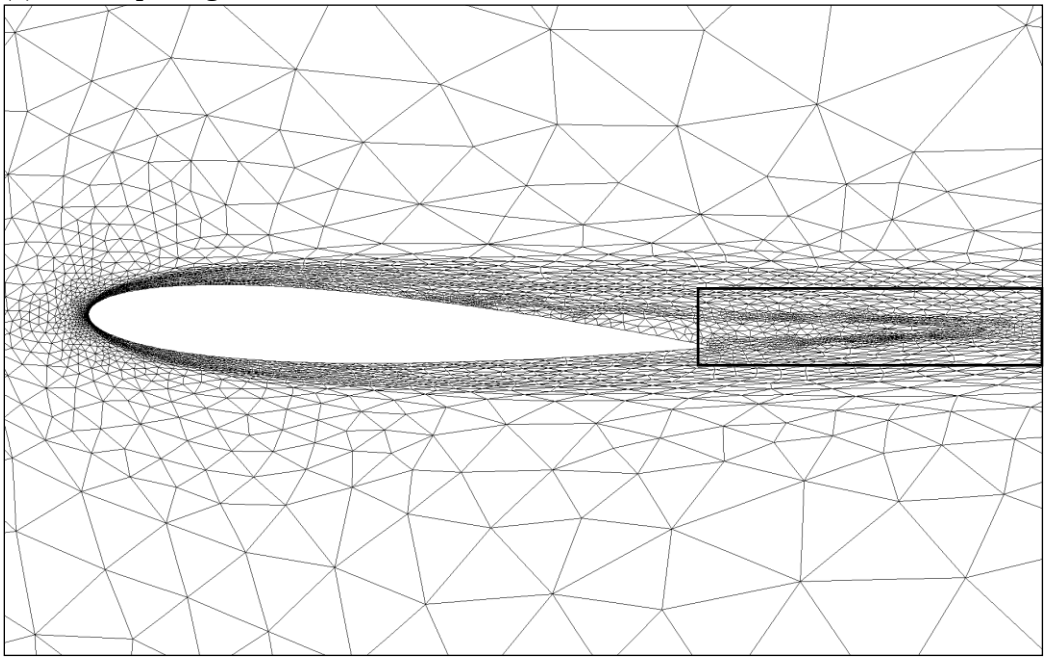


Figure 8. Lower surface anomaly in boundary layer with scalar error indicator. NACA 0012 airfoil at Re=10,000 and α=3°. The top three meshes were obtained using the third speed gradient tensor, and the bottom three meshes were obtained using third velocity gradient tensor. Within each group of three meshes, the top mesh is the finest, and each below it is successively coarser. Anomalous large elements are evident on the lower airfoil surface in the two coarsest meshes obtained with third speed gradient tensor. The most obvious of these appear at mid-chord. A nearly identical pattern resulted in simulations at α=2°.

(a) Third speed gradient tensor



(b) Third Velocity gradient tensor

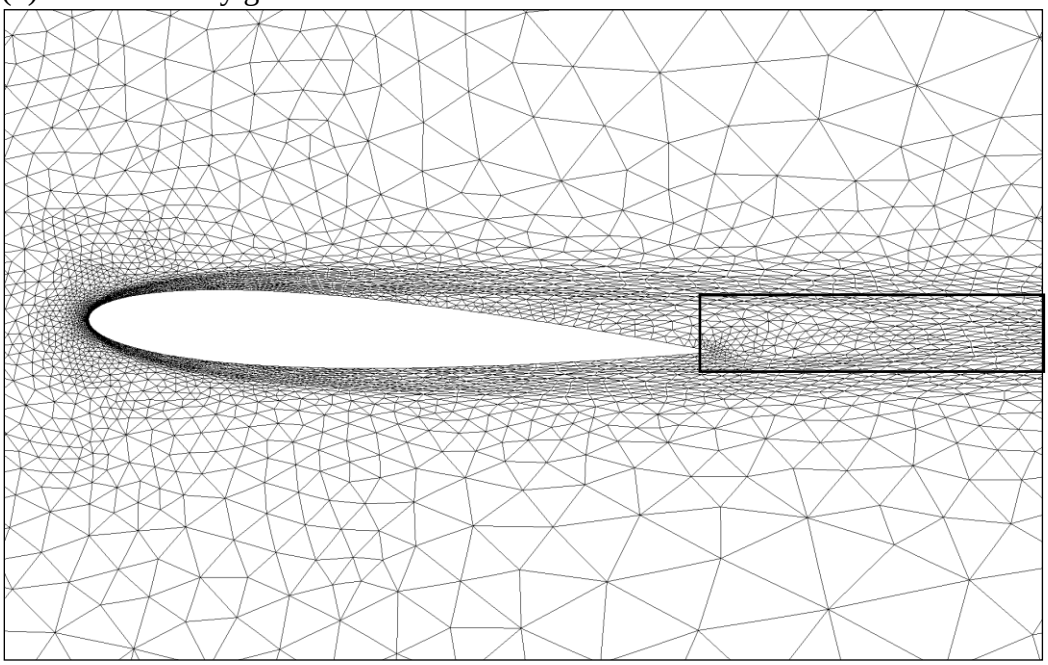
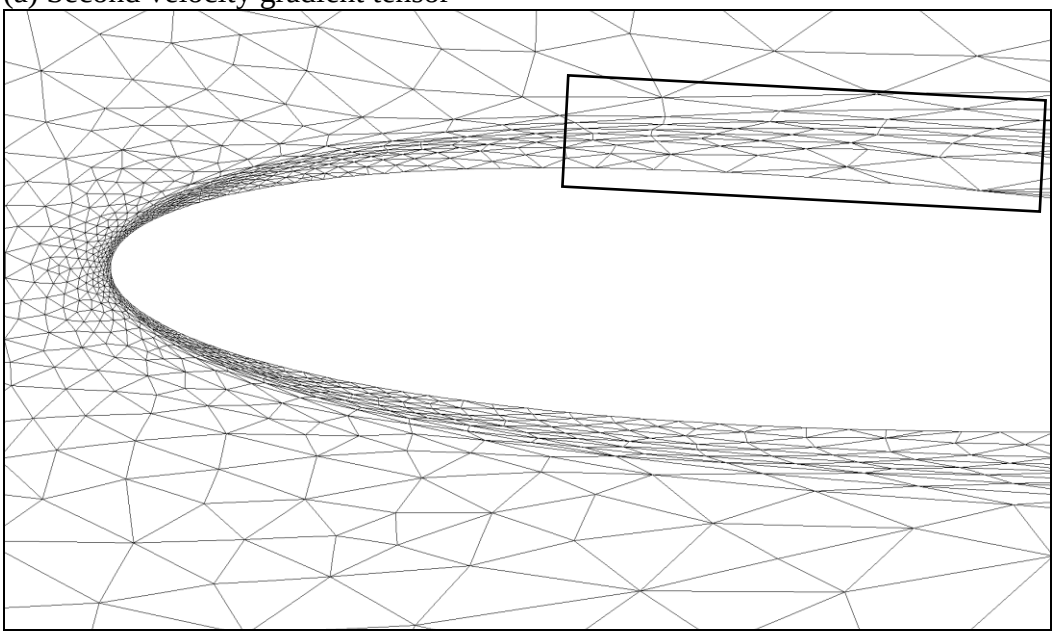


Figure 9. Excessive refinement in eddy with third scalar gradient error indicator. Upper mesh shows excessive refinement in low-speed region of eddy with third scalar gradient error indicator. The lower mesh was obtained with the third velocity gradient tensor and is free of this effect.. NACA 0012 airfoil at Re = 10,000 and $\alpha = 3°$.

(a) Second velocity gradient tensor



(b) Third velocity gradient tensor

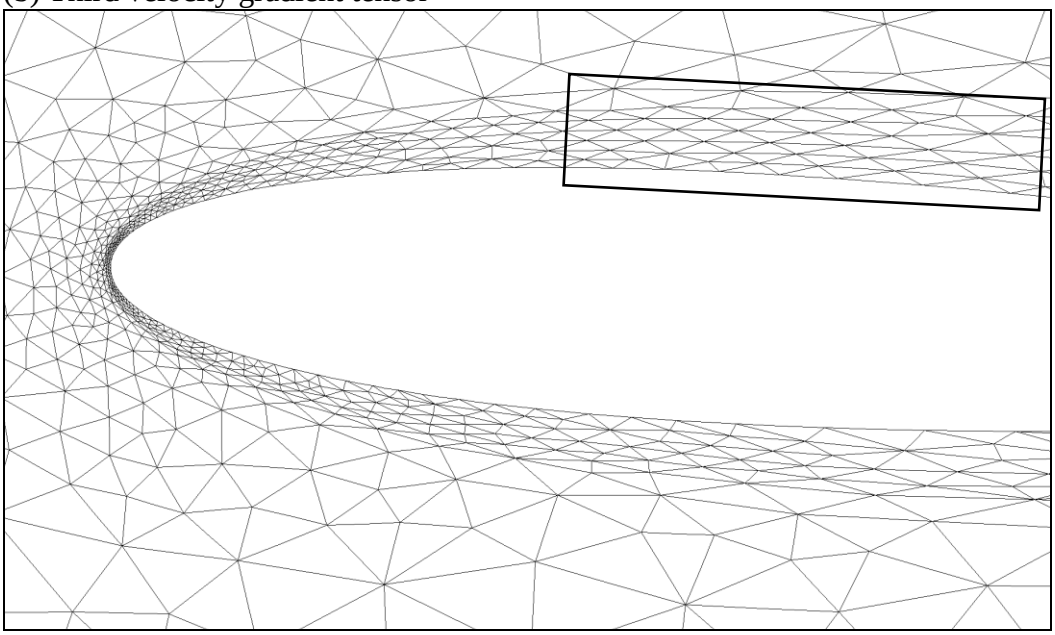
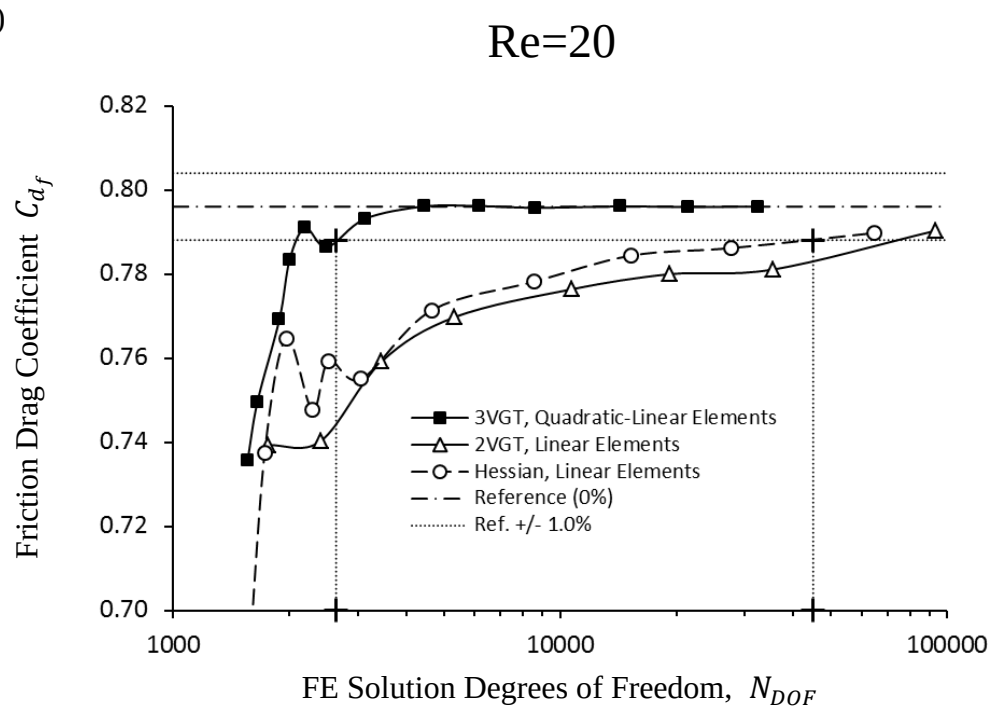


Figure 10. Comparison of element density and height distribution in boundary layer with second and third velocity gradients used as error indicators. Upper mesh was obtained with the second gradient, and shows a lower element density and greater element height near the surface than further away in the boundary layer. Lower mesh was obtained with the third gradient, and shows a much more uniform element density and height as a function of distance from the airfoil surface. NACA 0012 airfoil at a Reynolds number of 10,000 and an angle of attack α of 3°

(a)



(b)

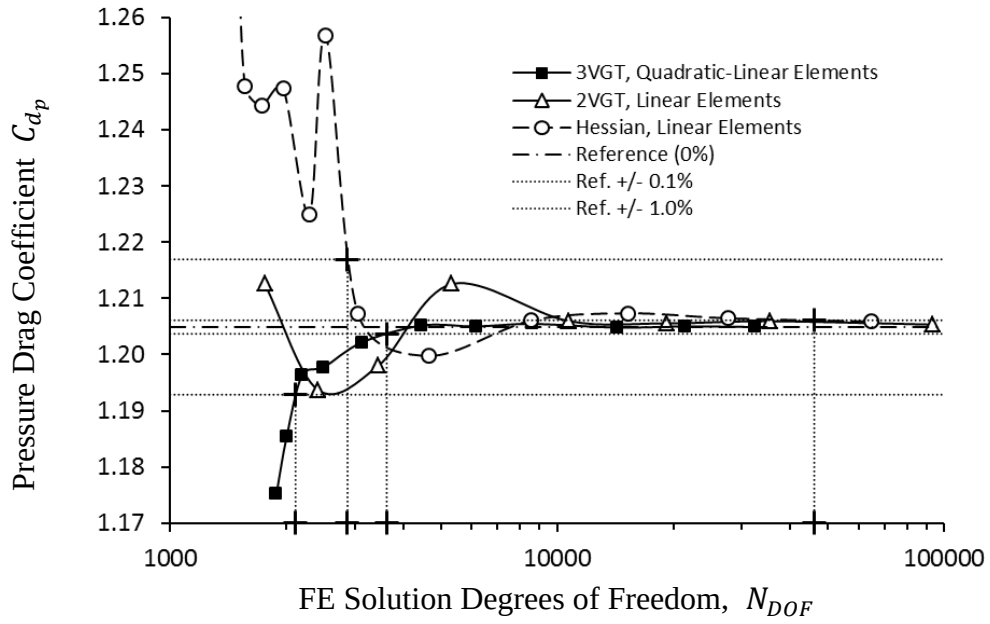
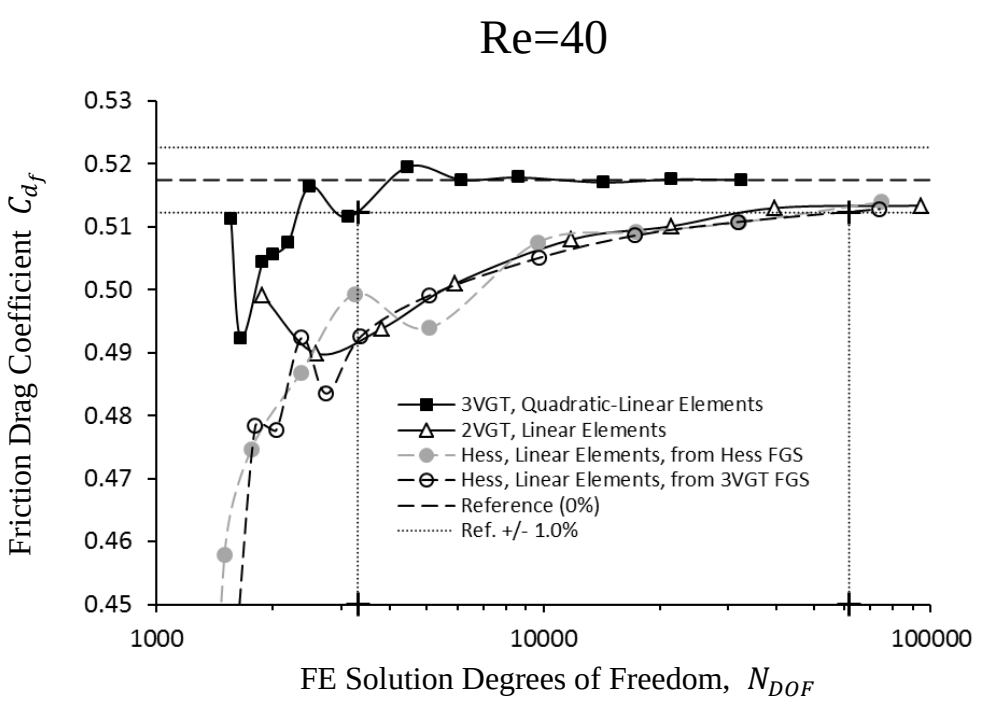


Figure 11. Friction and pressure drag coefficients of circular cylinder in uniform flow at Reynolds number Re=20.

(a)



(b)

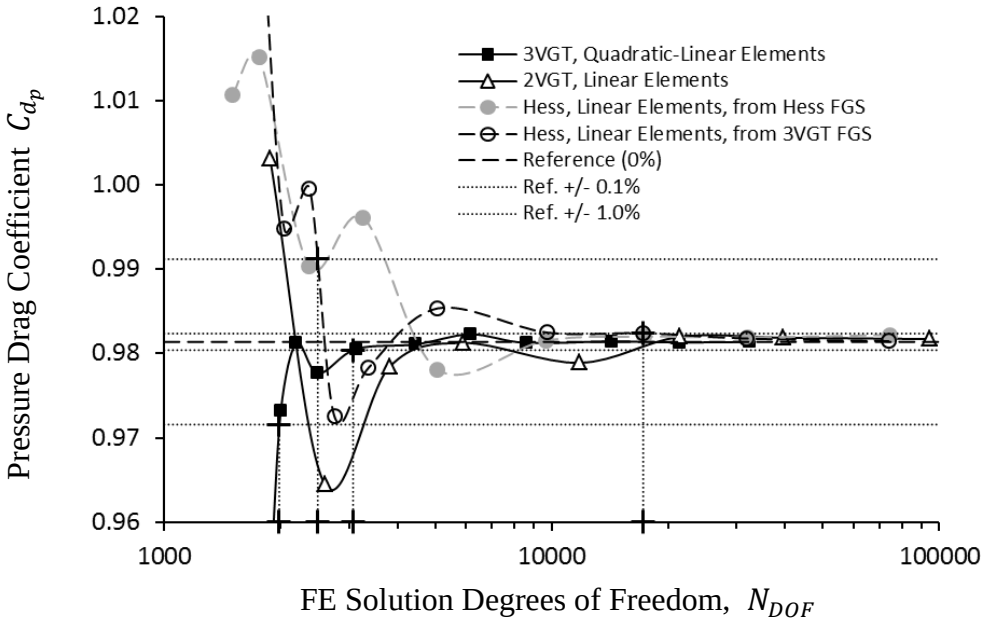
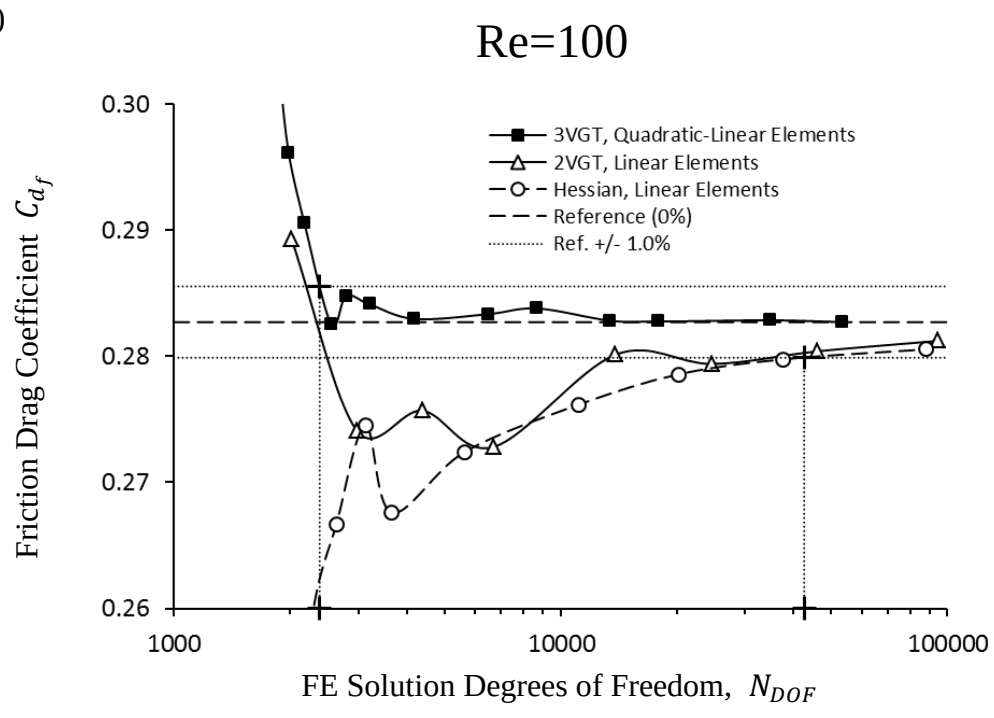


Figure 12. Friction and pressure drag coefficients of circular cylinder in uniform flow at Reynolds number Re=40.

(a)



(b)

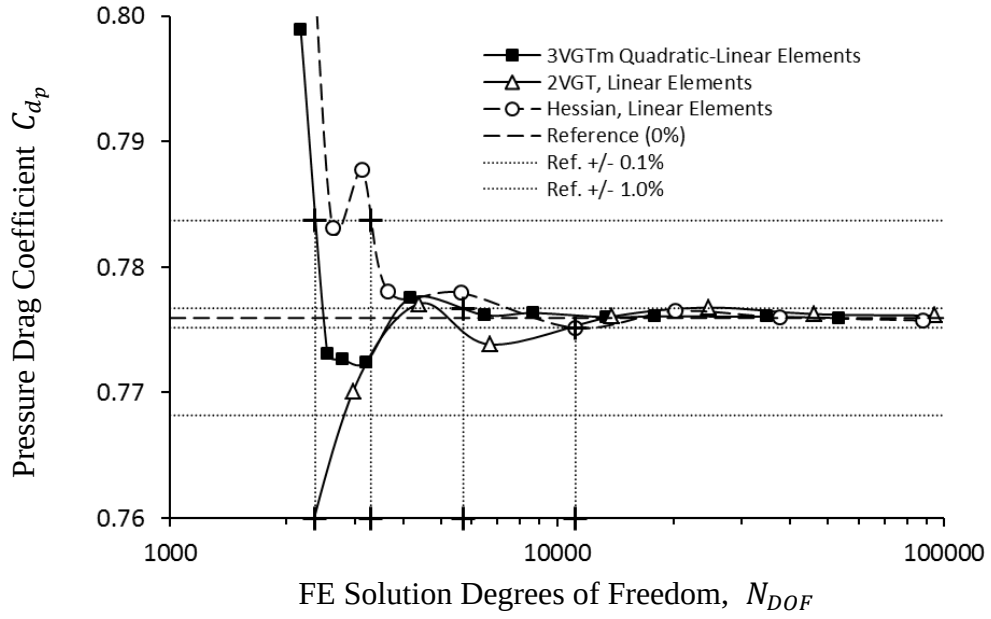


Figure 13. Friction and pressure drag coefficients of circular cylinder in uniform flow at Reynolds number Re=100.

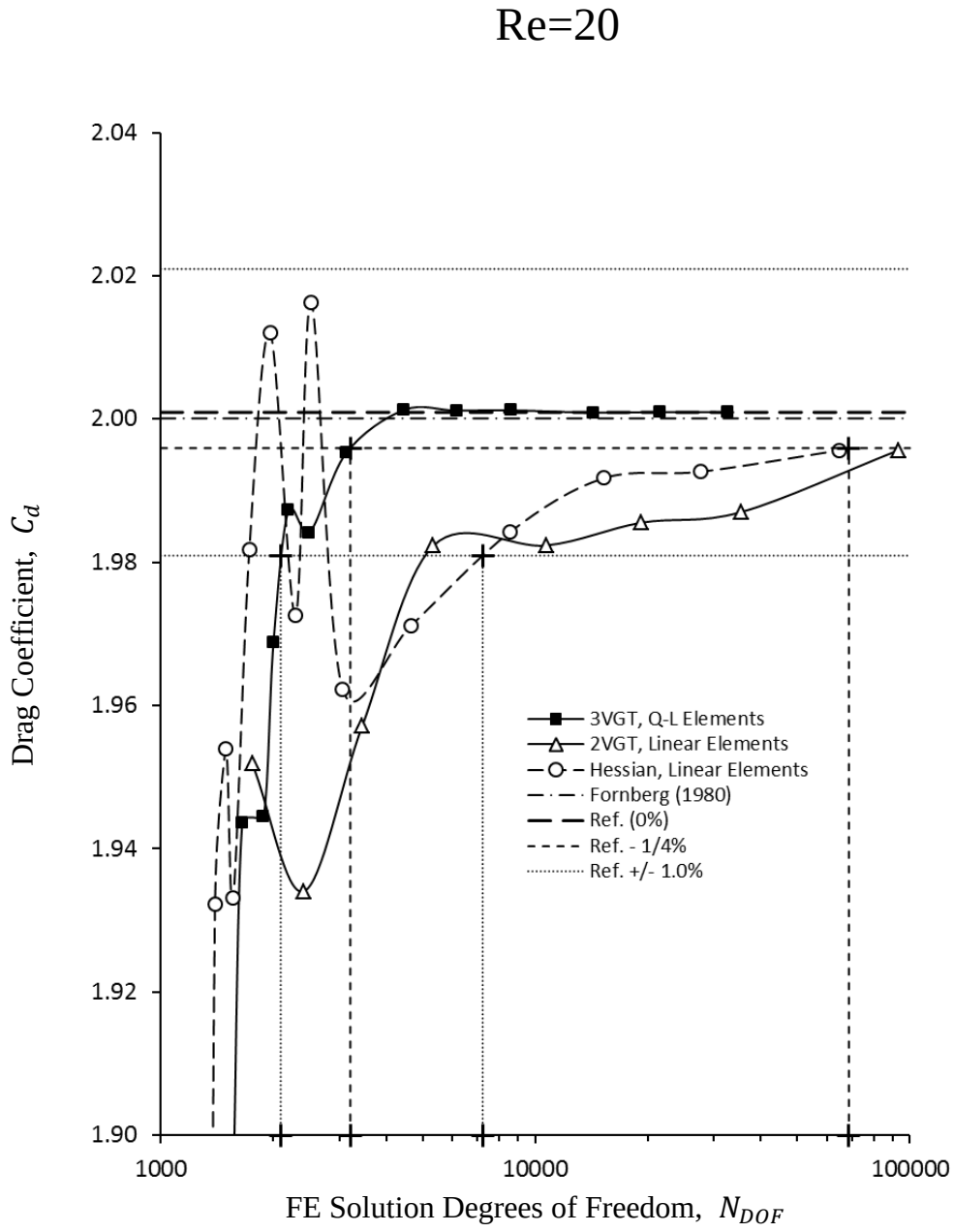


Figure 14. Computed total drag coefficient of a circular cylinder in a uniform flow at Reynolds number Re=20.

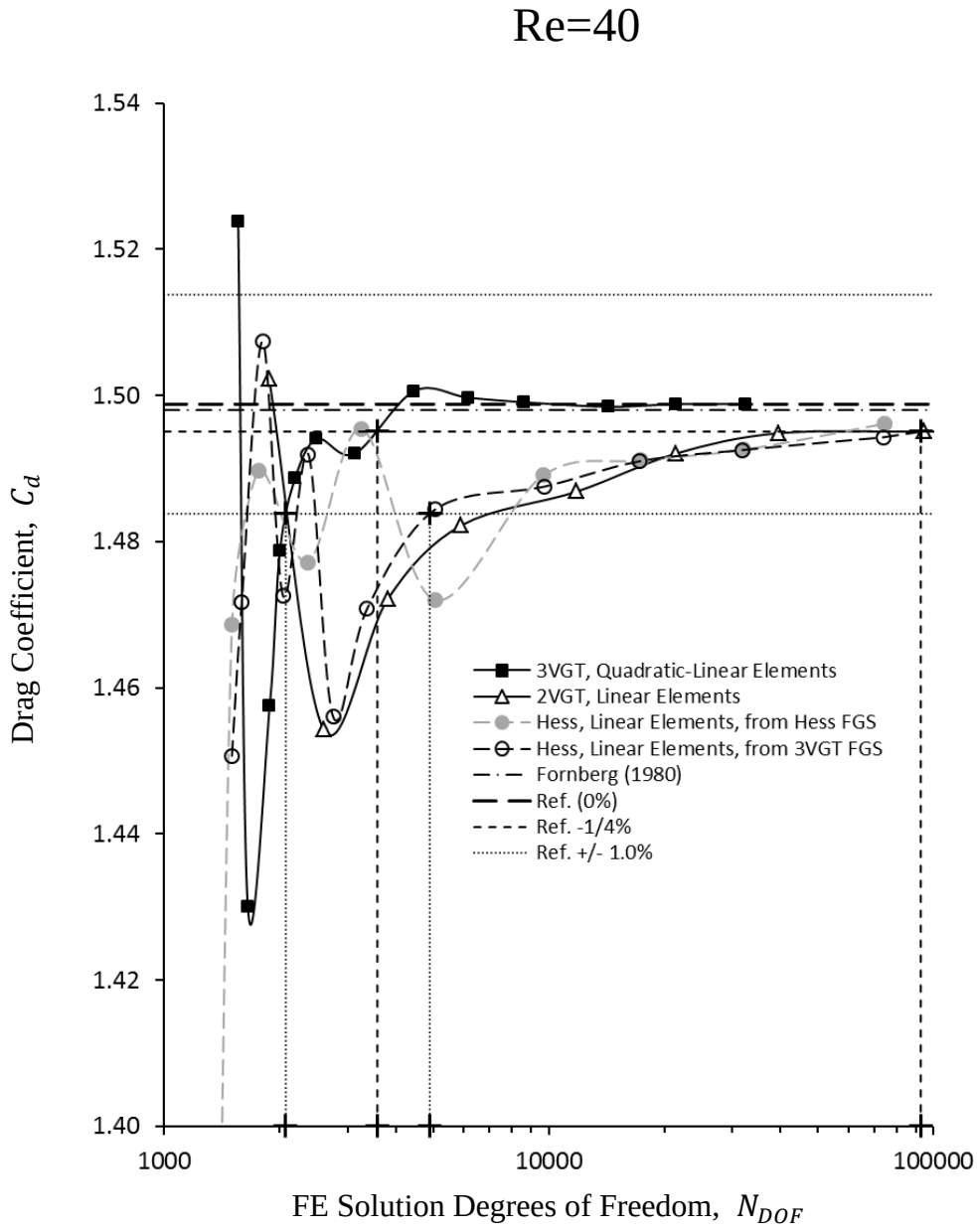


Figure 15. Computed total drag coefficient of a circular cylinder in a uniform flow at Reynolds number Re=40.

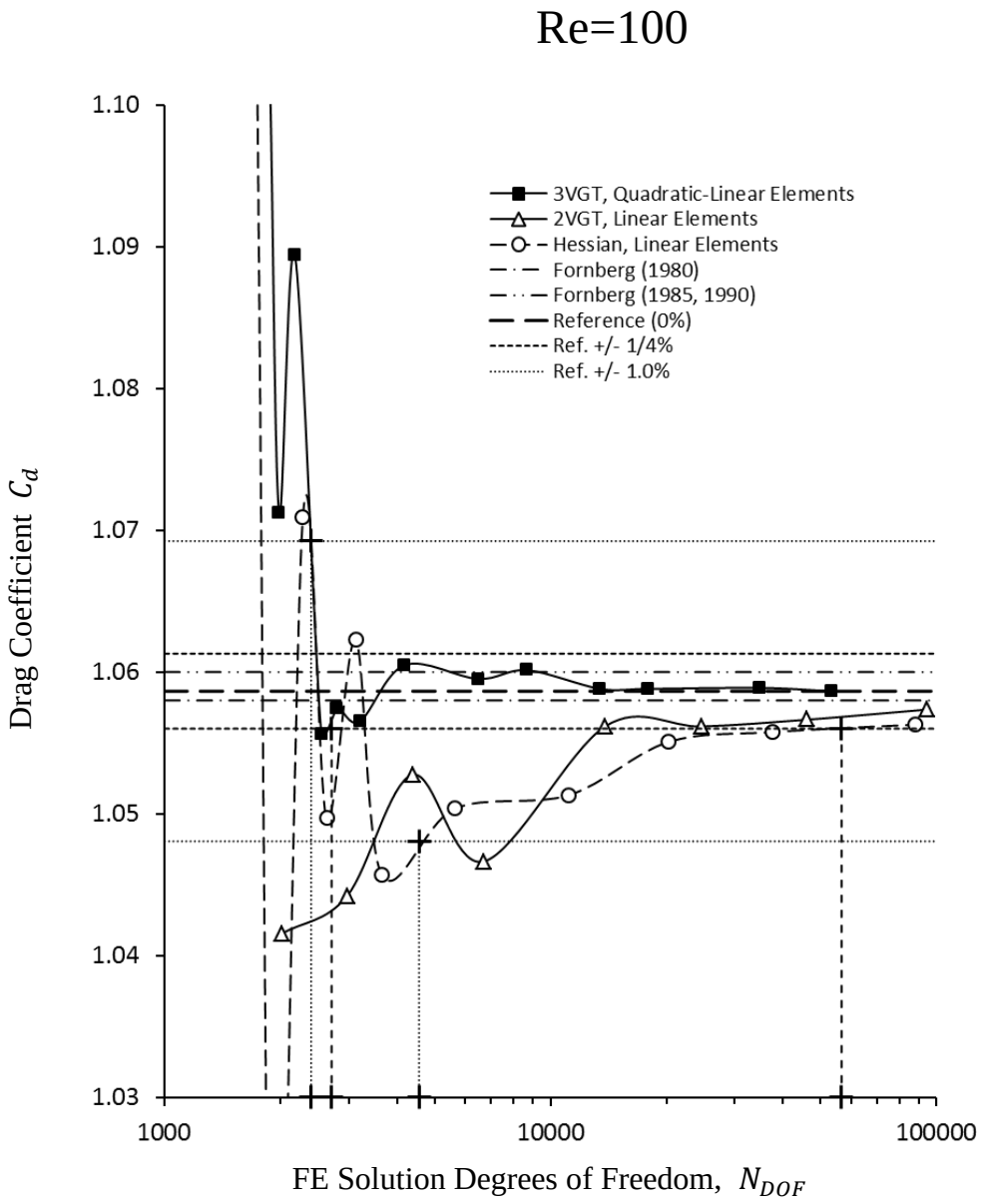
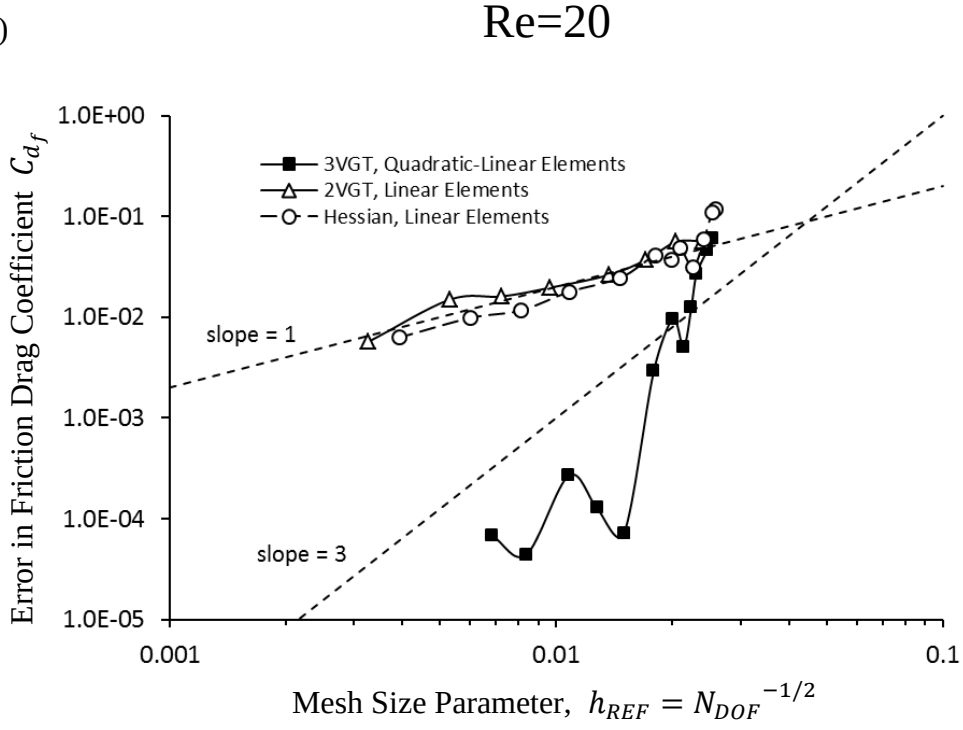


Figure 16. Computed total drag coefficient of a circular cylinder in a uniform flow at Reynolds number Re=100.

(a)



(b)

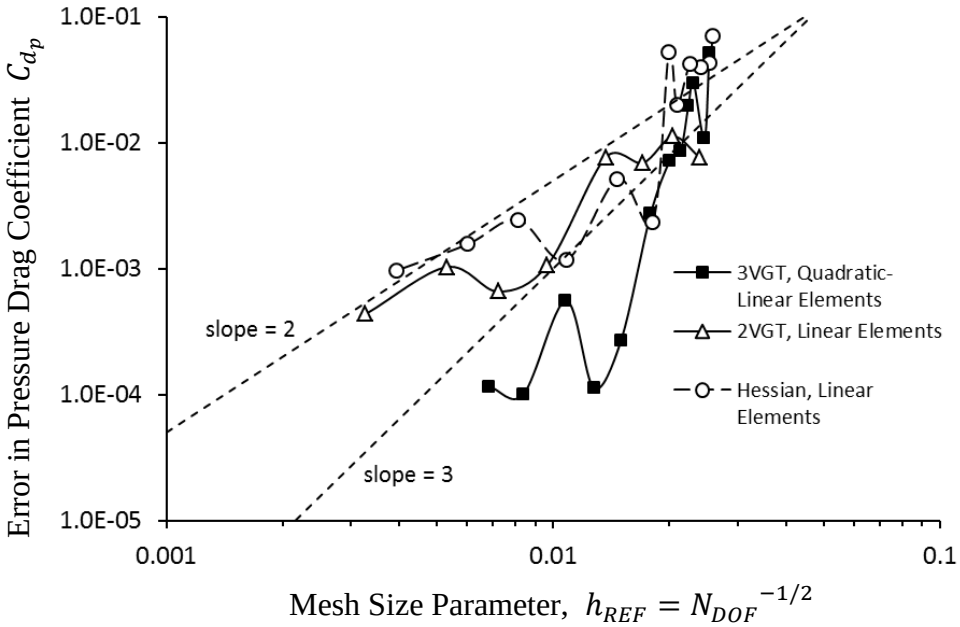
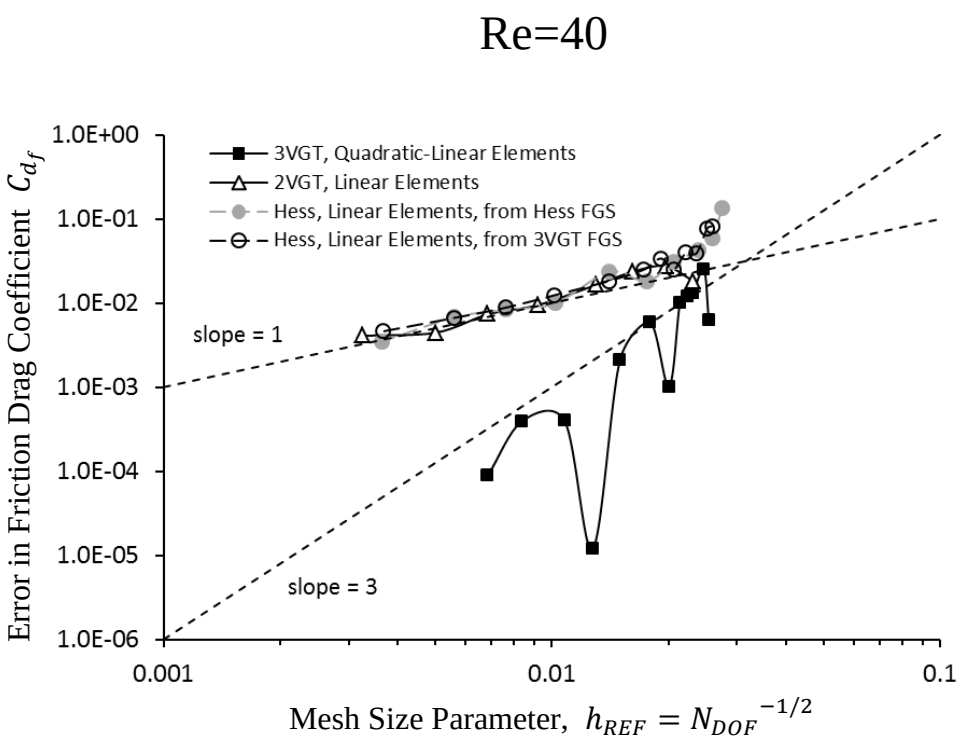


Figure 17. Estimated error of drag coefficients of circular cylinder in uniform flow at Reynolds number Re=20.

(a)



(b)

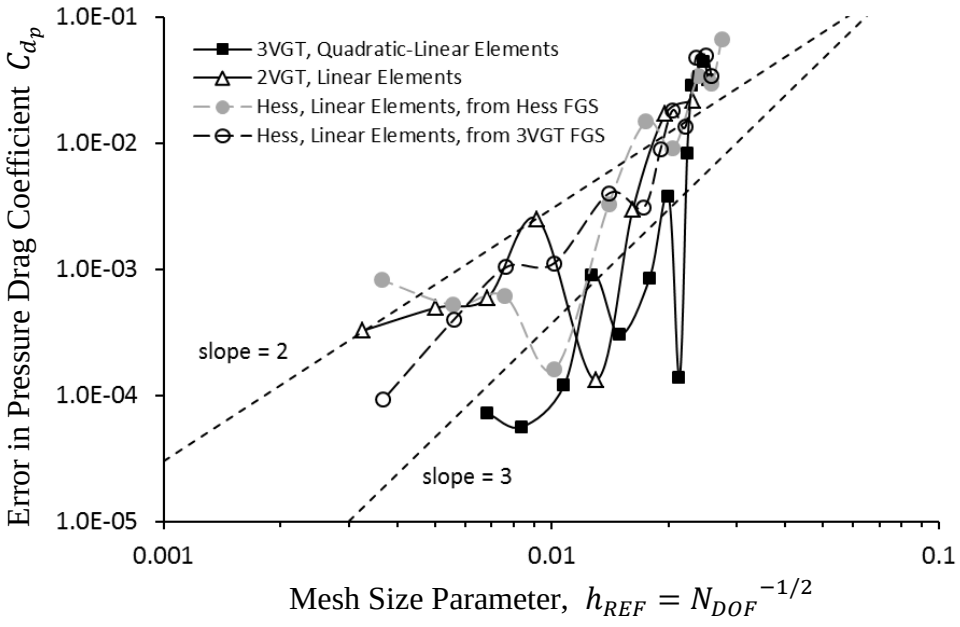
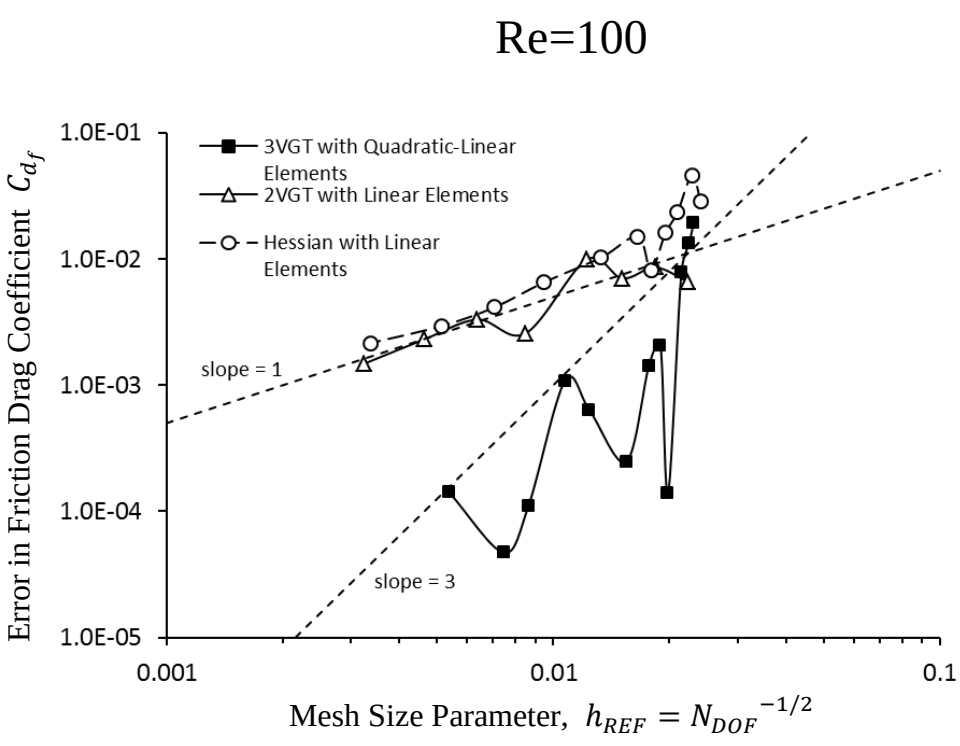


Figure 18. Estimated error of drag coefficients of circular cylinder in uniform flow at Reynolds number Re=40.

(a)



(b)

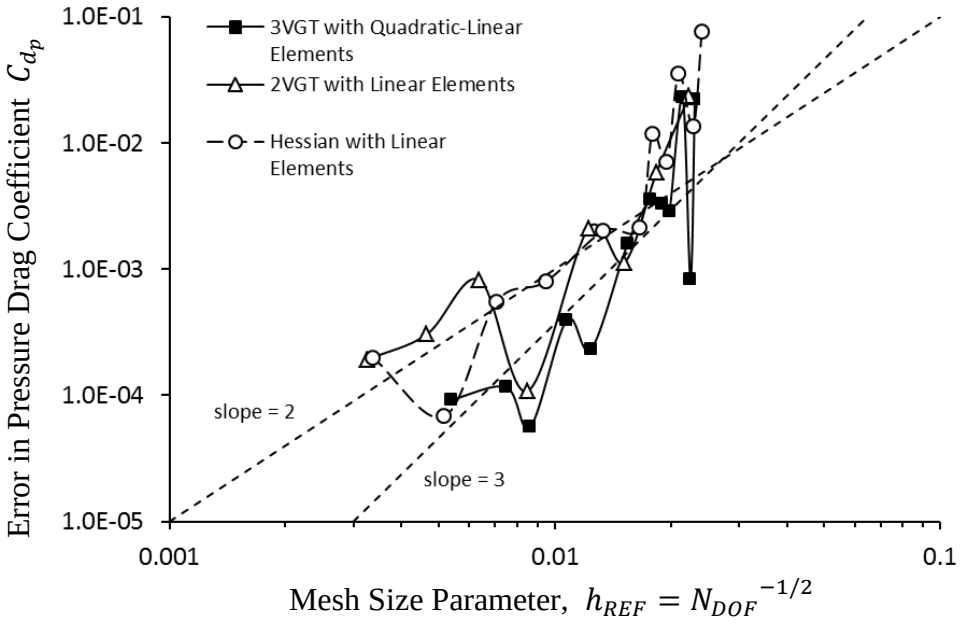
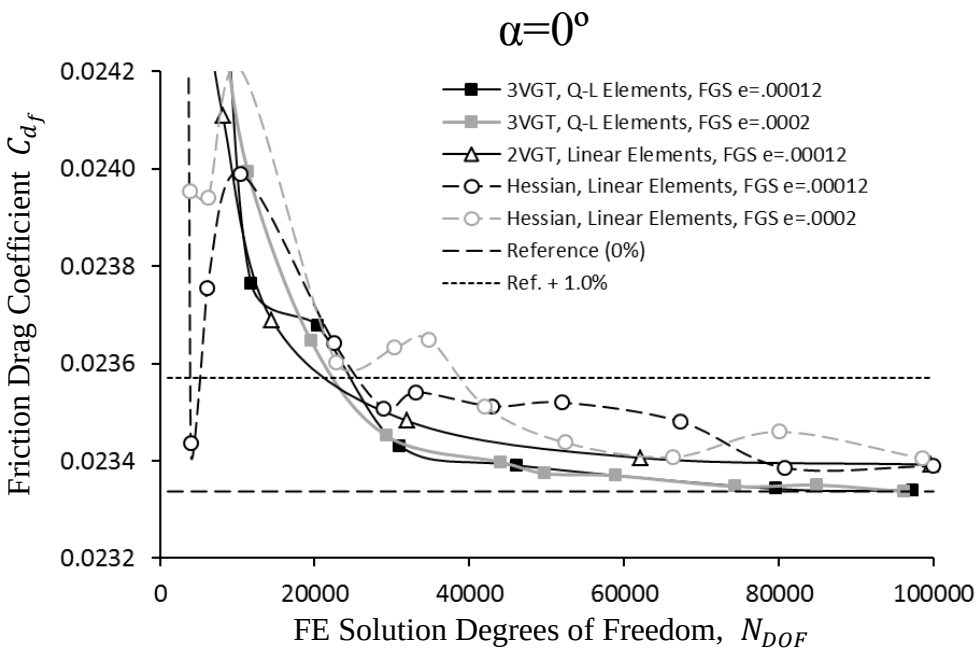


Figure 19. Estimated error of drag coefficients of circular cylinder in uniform flow at Reynolds number Re=100.

(a)



(b)

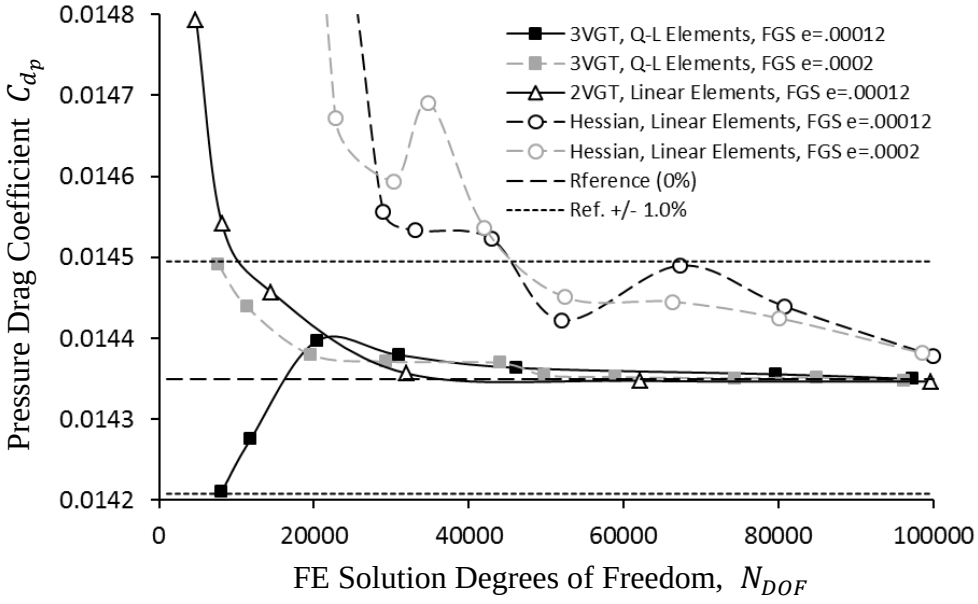
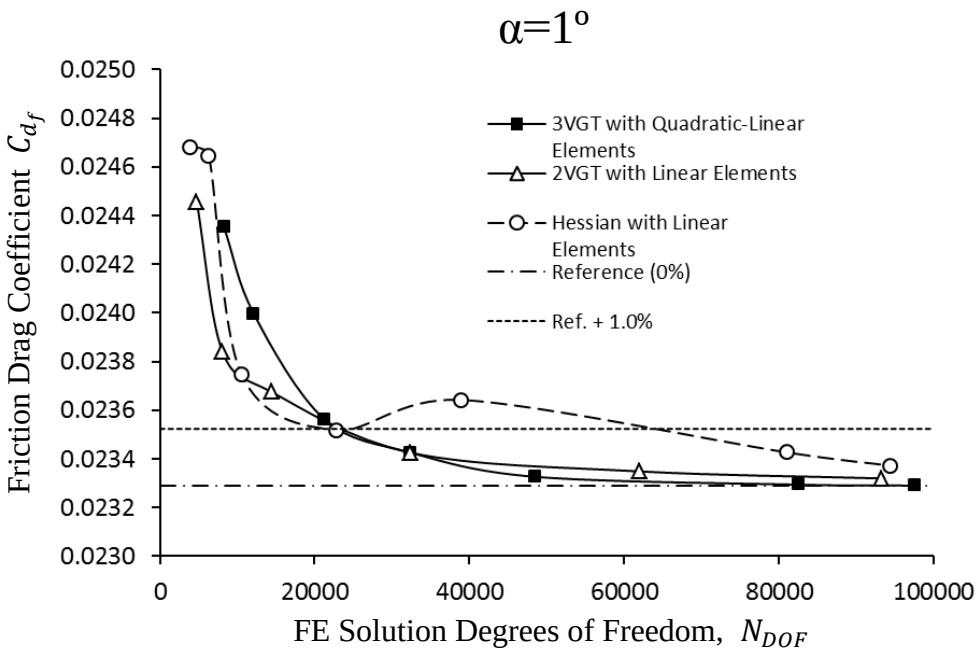


Figure 20. Friction and pressure drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=0º.

(a)



(b)

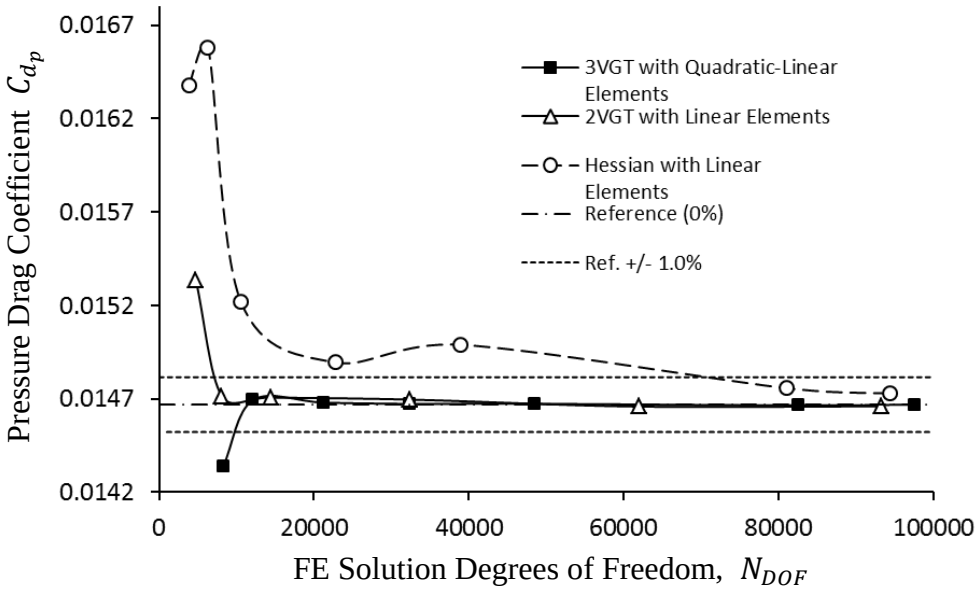
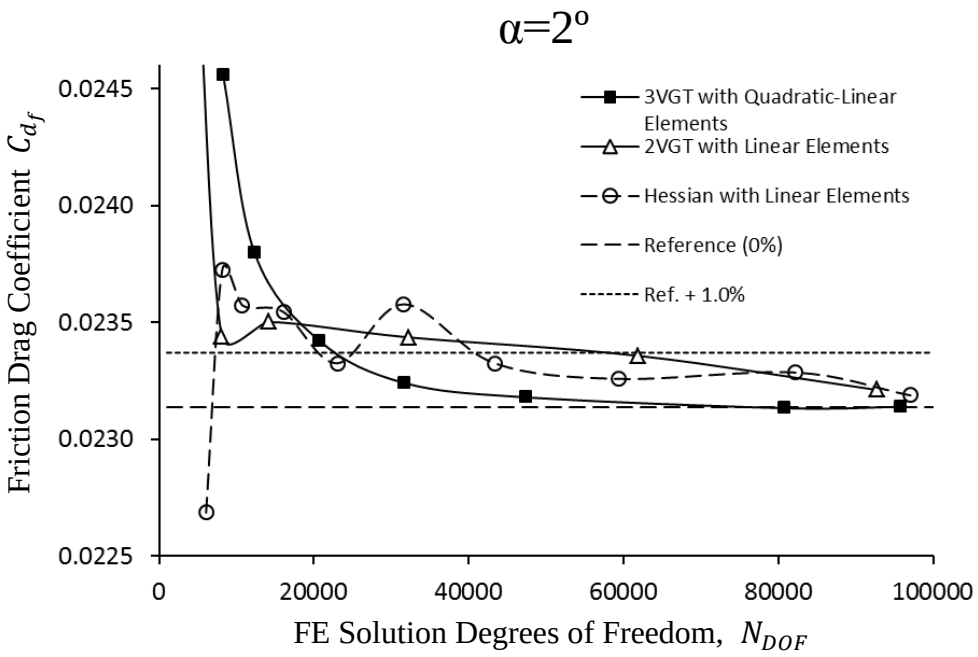


Figure 21. Friction and pressure drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=1°.

(a)



(b)

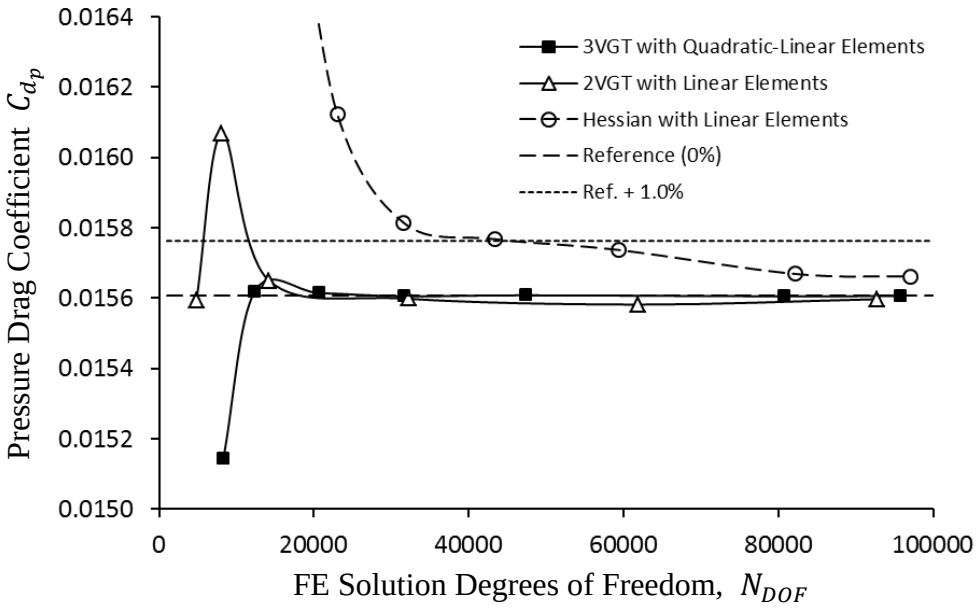
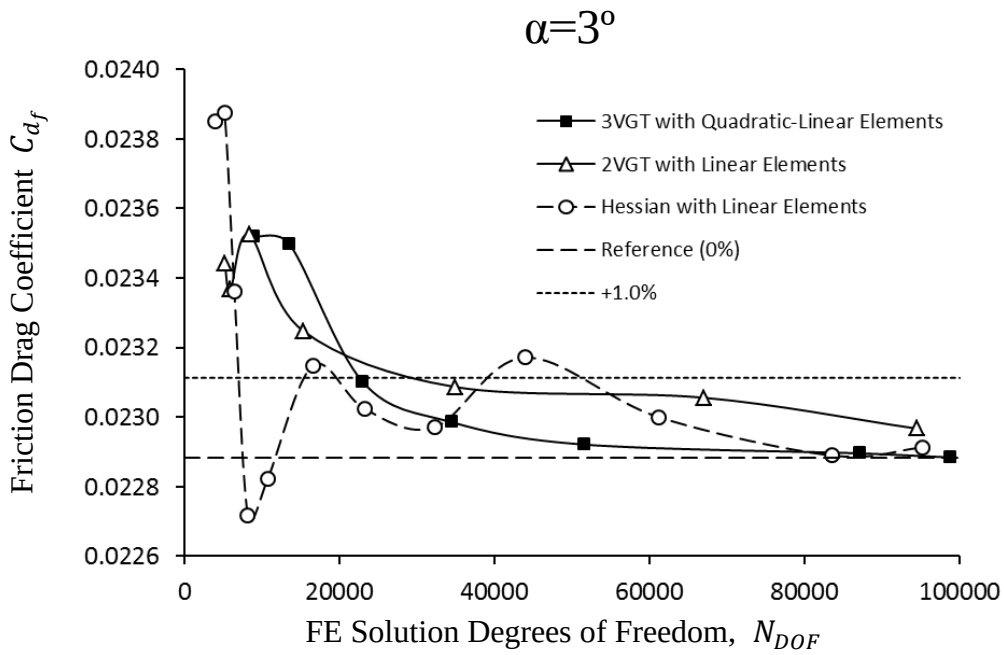


Figure 22. Friction and pressure drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=2°.

(a)



(b)

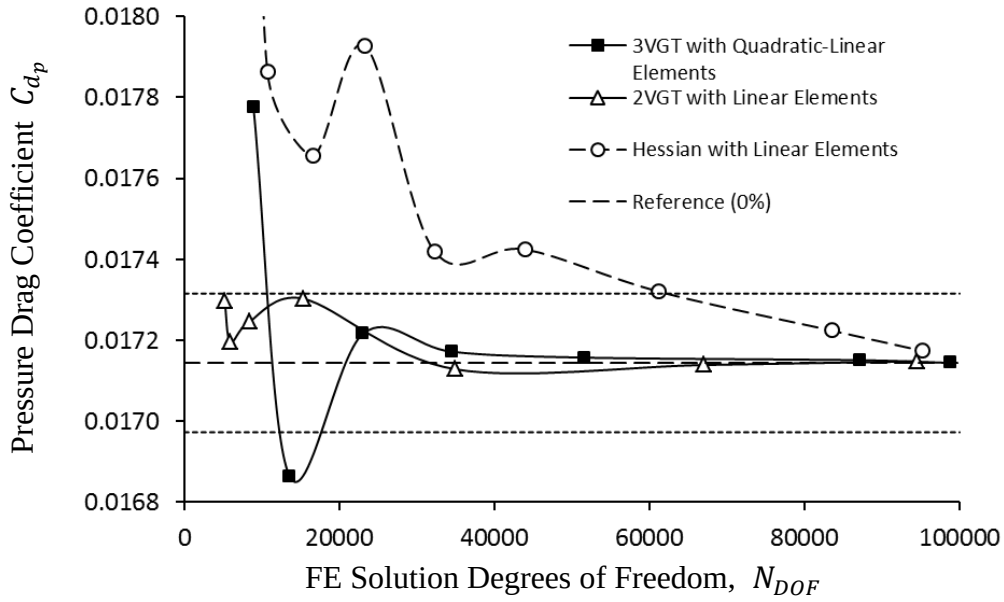
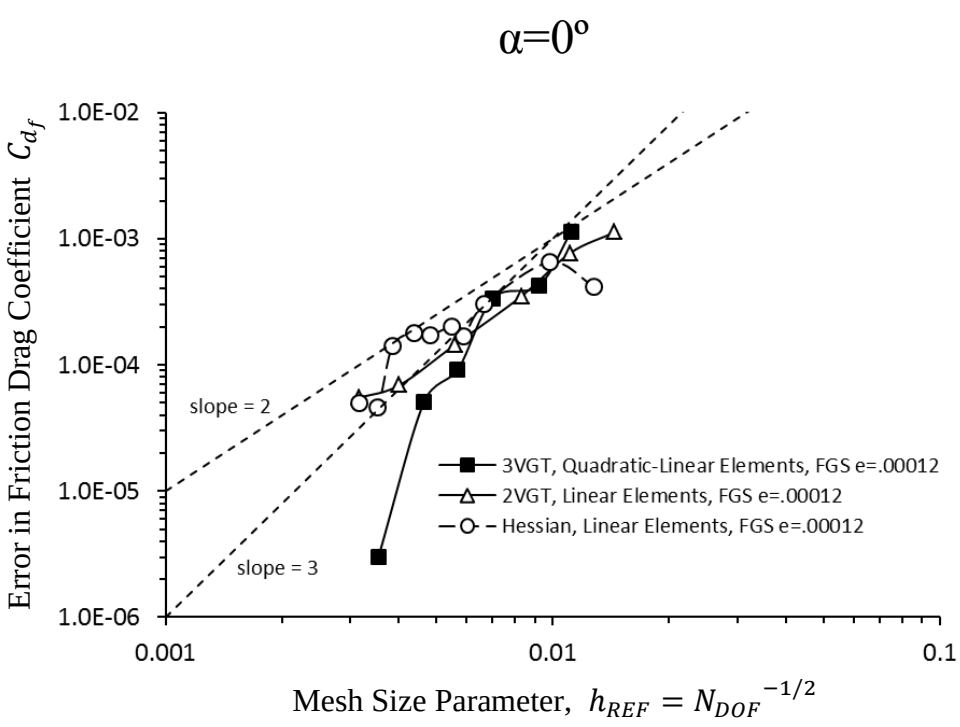


Figure 23. Friction and pressure drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=3º.

(a)



(b)

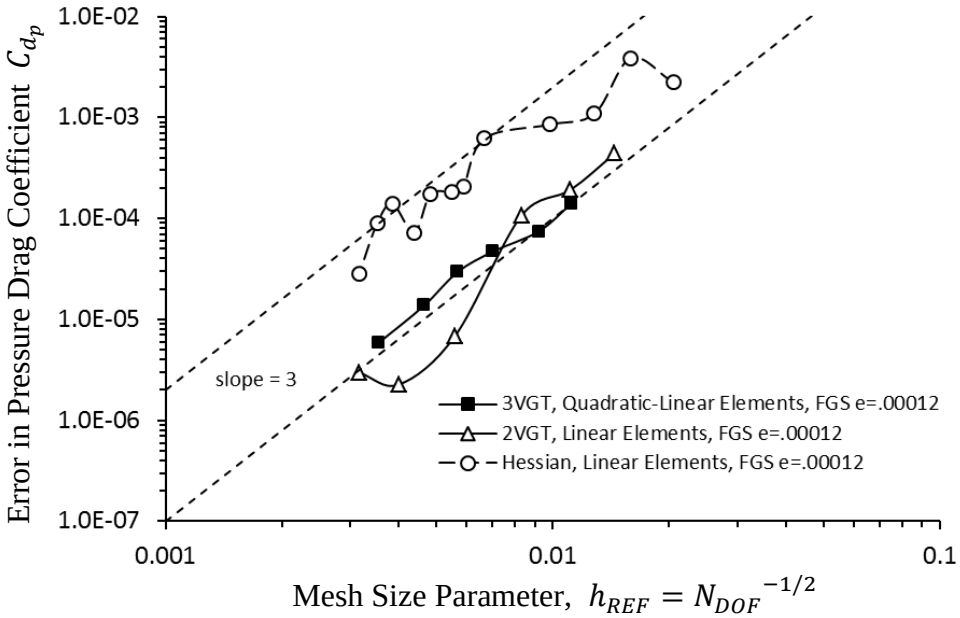
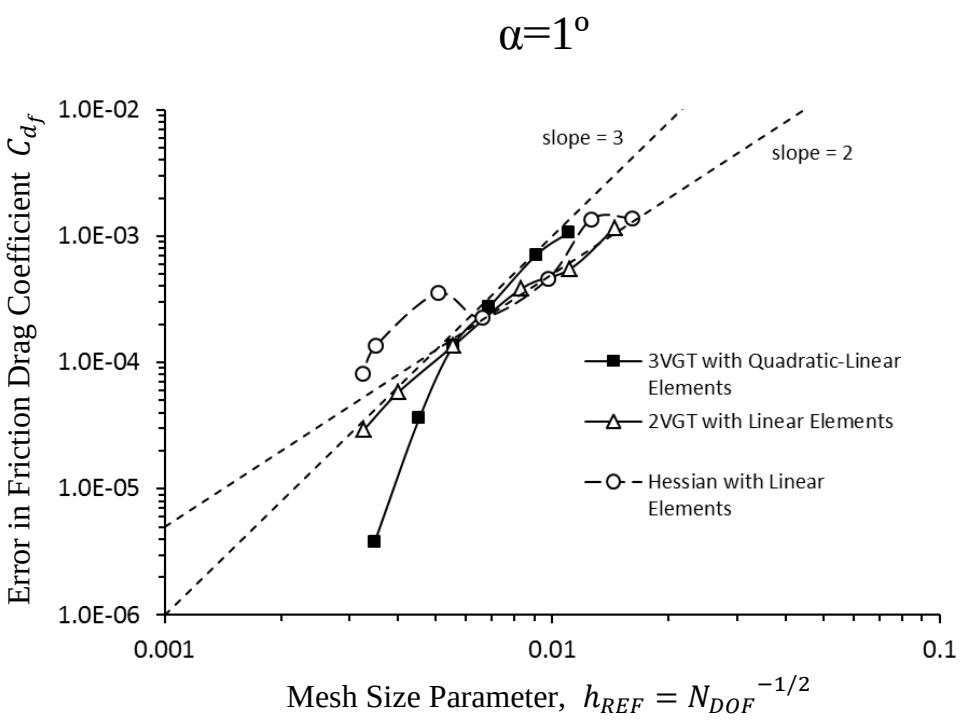


Figure 24. Estimated error of drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=0º.

(a)



(b)

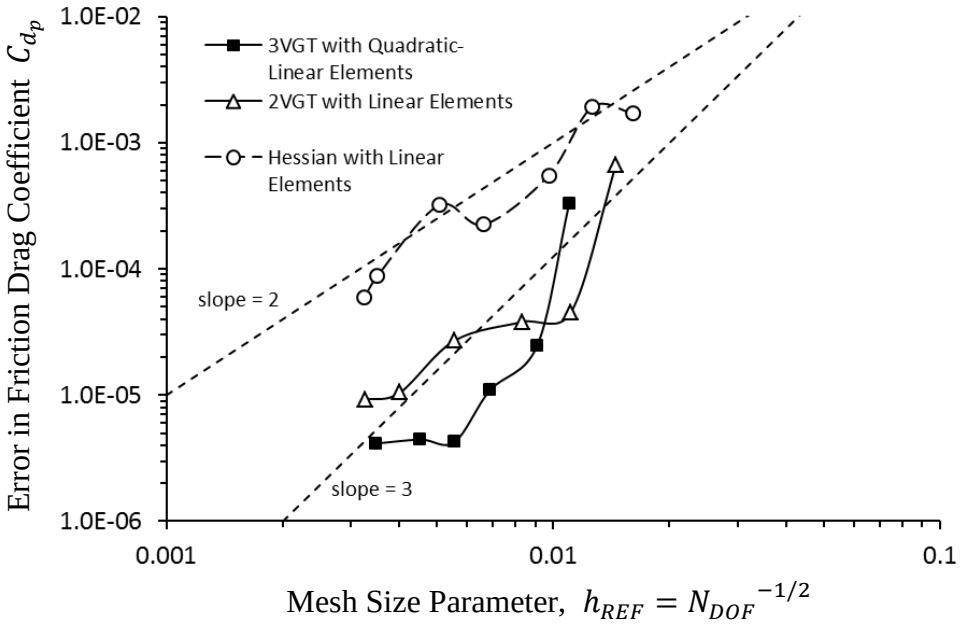
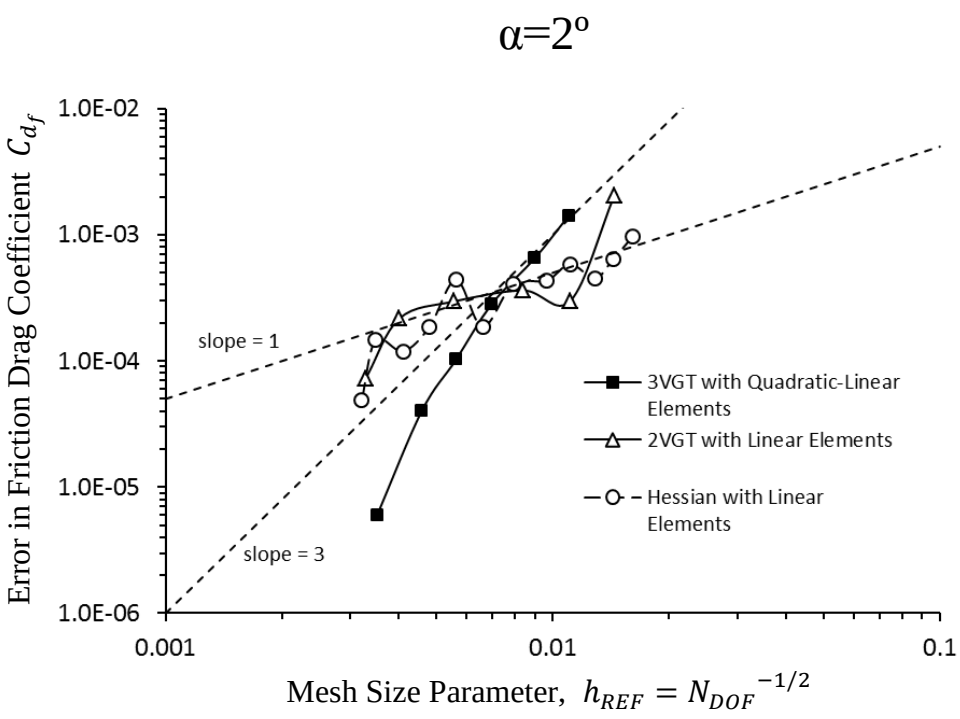


Figure 25. Estimated error of drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=1°.

(a)



(b)

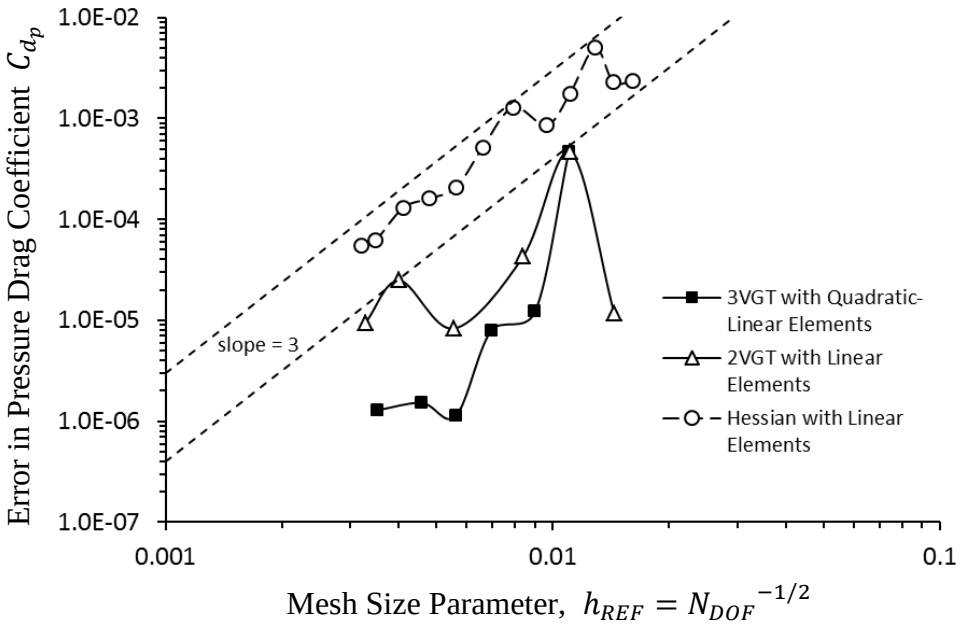
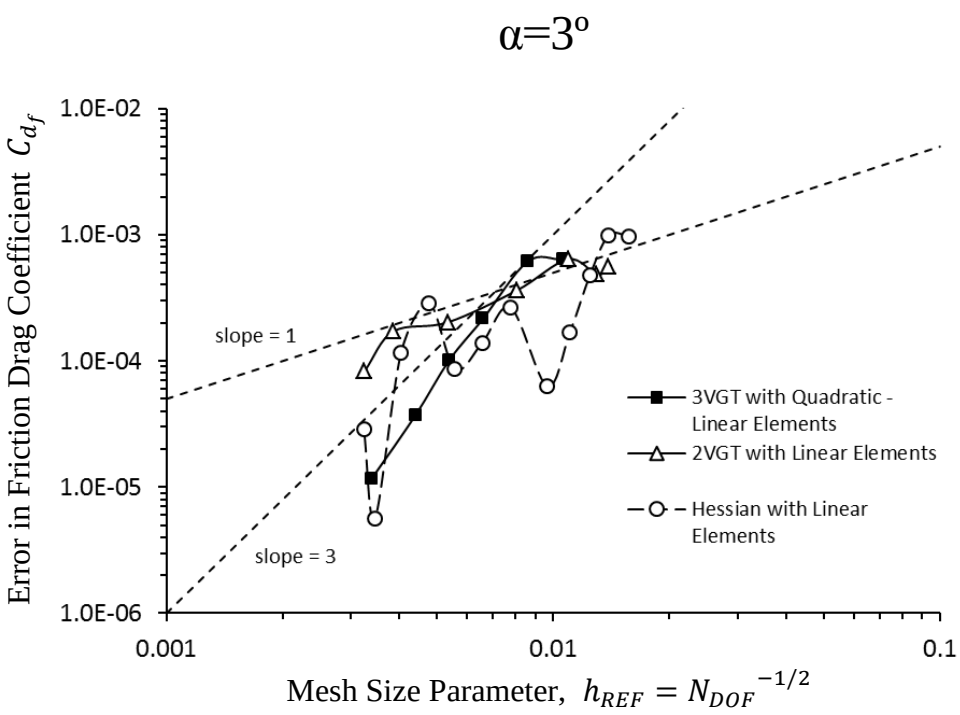


Figure 26. Estimated error of drag coefficients of NACA 0012 airfoil at Reynolds number Re=10,000 and angle of attack α=2º.

(a)



(b)

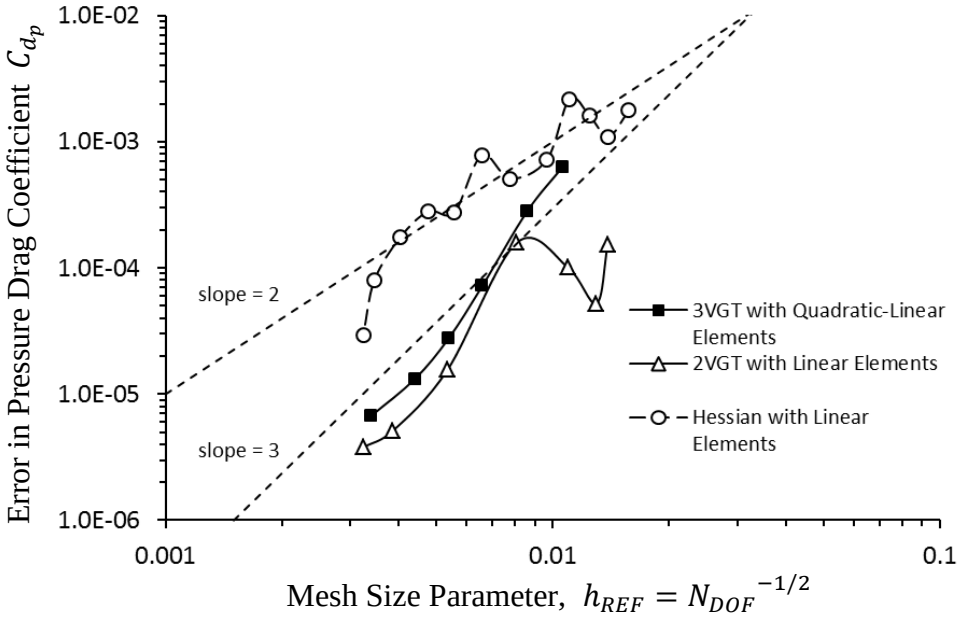


Figure 27. Estimated error of drag coefficients of NACA 0012 airfoil at Reynolds number Re-10,000 and angle of attack α=3º.

REFERENCES

AIT-ALI-YAHIA, D., BARUZZI, G., HABASHI, W. G., FORTIN, M., DOMPIERRE, J. &VALLET, M. G. 2002 Anisotropic Mesh Adaptation: Towards User-Independent, Mesh-Independent and Solver-Independent CFD. Part II: Structured Grids, *Intl. J. Numer. Methods Fluids*, **39**, 657-673.

AUBRY, R., LOHNER, R. 2009 Generation of Viscous Grids at Ridges and Corners, *Intl. J. for Numerical Methods in Engrg.* **77**, 1247-1289.

BABUSKA, I., RHEINBOLDT, W.C. 1979 Analysis of Optimal Finite-Element Meshes in R^1, *Mathematics of Computation*, **33** 435-463.

BOROUCHAKI, H., GEORGE, P.L., HECHT, F., LAUG P., SALTEL, E. 1997 Delaunay Mesh Generation Governed by Metric Specifications. Part I. Algorithms, *Finite Elements in Analysis and design* **25** 61-63

CASTRO-DIAZ, M. J., HECHT, F., MOHAMMADI, B. & PIRONNEAU, O. 1997 Anisotropic Unstructured Mesh Adaptation for Flow Simulations, *Intl. J. Numer. Methods Fluids*, **25**, 475-491.

D'AZEVEDO, E.F., SIMPSON, R.B. 1991 On Optimal Triangular Meshes for Minimizing the Gradient Error, *Numerische Mathematik*, **59** 321-348.

DE LATHAUWER, L., DE MOOR, B., VANDEWALLE, J. 2000 A Multilinear Singular Value decomposition. *SIAM J. Matrix Anlys. and Appl.* **21,** 1253-1278.

DENNIS S.C.R., CHANG, G.Z. 1970 Numerical Solutions for Steady Flow Past a Cylinder at Reynolds Numbers Up to 100, *J. Fluid Mech.* **42**, 471-489.

DENNIS, S.C.R. 1973 The Numerical Solution of the Vorticity Transport Equation, *Proc. 3rd Int. Conf. on Numerical Methods in Fluid Mech.*, vol. 2 (ed. H. Cabannes & R. Tewam), *Lecture notes in Physics*, **19**, 120.

DENNIS, S.C.R. 1976 A Numerical Method for Calculating Steady Flow Past a Cylinder, *Proc. 5th Int. Conf. on Numerical Methods in Fluid Mech.* (ed. A. I. Van De Vooren & P. J. Zandbergen), *Lecture notes in Physics*, **59**, 165.

DOMPIERRE, J., VALLET, M. G., BOURGAULT, Y., FORTIN, M. & HABASHI, W. G. 2002 Anisotropic Mesh Adaptation: Toward User-Independent, Mesh-Independent and Solver-Independent CFD. Part III: Unstructured Meshes, *Intl. J. Numer. Methods Fluids*, **39**, 675-702.

ELLIOTT, J. & PERAIRE, J. 1997 Practical Three-Dimensional Aerodynamic Design and Optimization Using Unstructured Meshes, *Am. Inst. Aeronaut. Astron.* **35,** 1479-1485.

FORNBERG, B. 1980 A Numerical Study of Steady Viscous Flow Past a Circular Cylinder Up to Reynolds Number 600, *J. Comput. Phys.* **98**, 819-855.

FORNBERG, B. 1985 Steady Viscous Flow Past a Circular Cylinder Up To Reynolds Number 60, *J. Comput. Phys.* **61**,297-320.

FORNBERG, B. 1991 Steady Incompressible Flow Past a Row of Circular Cylinders, *J. Fluid Mech.* **225**, 665-671.

FREY, P. J. & ALAUZET, F. 2005 Anisotropic Mesh Adaptation for CFD Computations, *Computer Methods in Applied Mechanics and Engrg.* **194**, 5068-5082.

FRINK, N. T., PIRZADEH, S. Z., ATKINS H. L., VIKEN, S. A.MORRISON, J. H. 2010 CFD Assessment of Aerodynamic Degradation of a Subsonic Transport Due to Airframe Damage, AIAA paper 2010-500.

GUNZBURGER, M. D. 1989 *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, Inc.

HOOD, P. 1976 Frontal Solution Program for Unsymmetric Matrices. , *Intl. J. Numer. Methods Engrg.* **10**, 379-399

HUGHES, T. J. R., FRANCA, L. P. & BALESTRA, M. 1986 A New Finite Element Formulation for Computational Fluid Dynamics: V, Circumventing the Babuška-Brezzi Condition: A Stable Petrov-Galerkin Formulation of the Stokes Problem Accommodating Equal-Order Interpolations, *Comput. Methods Appl. Mech. Engrg.* **59**, 85-99.

JIN, H. & WIBERG, N. E. 1990 Two-Dimensional Mesh Generation, Adaptive Remeshing and Refinement, *Intl J. Numer. Methods Engrg.*. **29**, 1501-1526.

KIM, H. J., CLEMENTEL, I. E. V., FIGUEROA, C. A., JANSEN, K. E. & TAYLOR, C. A. 2010 Developing Computational Methods for Three-Dimensional Finite Element Simulations of Coronary Blood Flow, *Finite Elements in Analysis and Design,* **46**, 514-525.

MAVRIPLIS, D.J., PIRZADEH, S. 1999 Large-Scale Parallel Unstructured Mesh Computations for Three-Dimensional High-Lift Analysis, *J.of Aircraft* **36**, 987-998.

MORTON, S. A., CUMMINGS, R. M. & KHOLODAR, D. B. 2007 High Resolution Turbulence Treatment of F/A-18 Tail Buffet, *J. Aircraft,* **6**, 1769-1775.

PERAIRE, J., JOAQUIN, J., FORMAGGIA, L., MORGAN, K. & ZIENKIEWICZ, O. C. 1988 Finite Element Euler Computations in Three Dimensions, *Intl J. Numer. Methods Engrg.* **26**, 2135-2159.

PERAIRE, J., VAHDATI, M., MORGAN, K. & ZIENKIEWICZ, O. C. 1987 Adaptive Remeshing for Compressible Flow Computations, *J. Comput. Phys.* **72**, 449-466.

PIRZADEH, S. 1994 Unstructured Viscous Grid Generation by the Advancing Layers Method, *AIAA J.* **32**, No. 8, 1735-1737.

PIRZADEH, S. 1996 Three-Dimensionl Unstructured Viscous Grids by the Advancing Layers Method, *AIAA J.* **34**, No. 1, 43-49.

PROBERT, J., HASSAN, J., PERAIRE, J. & MORGAN, K. 1991 An Adaptive Finite Element Method for Transient Compressible Flows, *Intl J. for Numer. Methods Engrg.* **32**, 1145-1159.

REMAKI, L. & HABASHI, W. G. 2009 A Posteriori Error Estimate Improvement in the Mesh Adaptation for Computer Fluid Dynamics Applications, *J. Mech. Engrg. Sci.* **223,** 1117-1126.

SAHNI, O., LUO, X. J.,JANSEN, K. E., SHEPHARD, M. S. 2010 Curved Boundary Layer Meshing for Adaptive Viscous Flow Simulations, *Finite Elements in Analysis and Design,* **46**, 132-139.

SAHNI, O., MÜLLER, J., JANSEN, K.E., SHEPHARD, M.S. & TAYLOR, C.A. 2005 Efficient Anisotropic Adaptive Discretization of the Cardiovascular System, *Computer Methods in Applied Mechanics and Engrg.* **195**, 5634-5655.

TAYLOR, C. A. & FIGUEROA, C. A. 2009 Patient-Specific Modeling of Cardiovascular Mechanics, *Ann. Rev. Biomedical Engr.* **11**, 109-134.

TRITTON, D.J. 1959 Experiments on the Flow Past a Circular Cylinder at Low Reynolds Numbers, *J. Fluid Mech.* **6**, 547-567.

VENDITTI, D. A. & DARMOFAL, D.L. 2003 Anisotropic Grid Adaptation for Functional Outputs: Application to two-dimensional Viscous Flows. *J. Comput. Phys.* **18**, 22-46.