REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0371

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
	Final	1 June 2000 - 30 September 2003

4. TITLE AND SUBTITLE

Adapative Navier-Stokes Flow Solvers for Aerospace Structures

5a. CONTRACT NUMBER

5b. GRANT NUMBER

F49620-00-1-0309

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

R. Panneer Selvam
Zu-Qing Qu

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

The University of Arkansas
Fayetteville, AR. 72701

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Office of Scientific Research
4015 Wilson Blvd
Mail Room 713
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A. Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Predicting wind induced aerodynamic forces on aerospace structures is usually computationally expensive even for powerful computational facilities. For these unsteady computations, adaptive finite clement technique may reduce the computer time and storage while keeping a desired accuracy. In this report, the p-version adaptive finite element method is implemented into a standard benchmark problem, the computational how around a circular cylinder, to compute aerodynamic forces. The error distribution for velocity is first estimated. Then, the polynomial order of the interpolation function is changed continuously in accordance with the changing of the error distribution. The second through fifth orders of polynomials are considered for the velocity in the adaptive method. One order less of polynomial is used for pressure. The benchmark problem of the flow around a circular cylinder with Reynolds number of 1000 is considered to study the performance.

15. SUBJECT TERMS

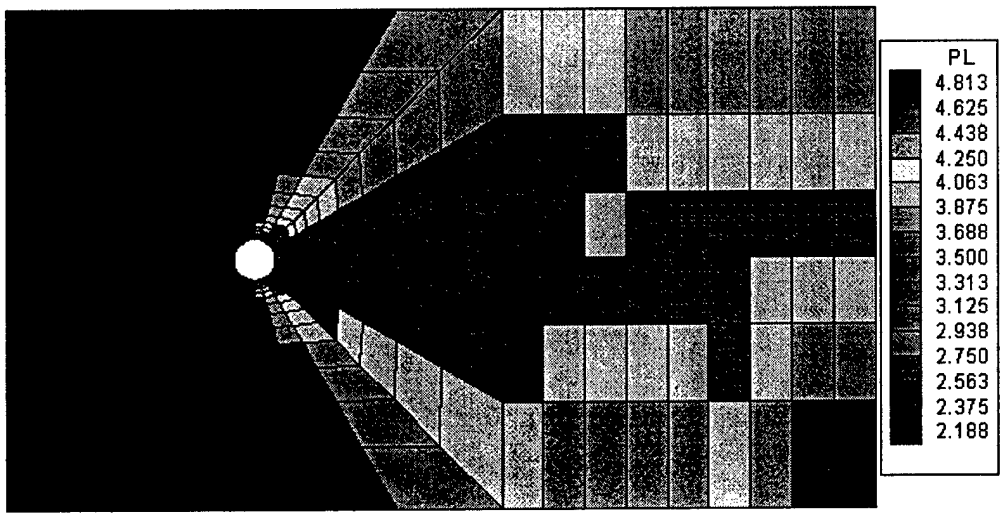
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			R. Panneer Selvam
					19b. TELEPHONE NUMBER *(Include area code)* 501-575-5356

ADAPTIVE NAVIER STOKES FLOW SOLVER FOR AEROSPACE STRUCTURES

(Final Report for 06/01/00-9/30/03, DOD/EPSCoR)

(Grant No: AFOSR; Grant No.: F49620-00-1-0309)

(May 2004)



R. Panneer Selvam

Zu-Qing Qu

Computational Mechanics Laboratory

4190 University of Arkansas, Fayetteville, Arkansas 72701

Content

Preface

Computer modeling is extensively applied in all areas of engineering and science. Especially for the design of economical and efficient aerospace structures computer modeling is a valuable tool. For flow around aerospace structures large computer memory and time is required due to complex geometry. If accurate solution is obtained with less number of grid point that is the ideal situation. To achieve this objective adaptive Finite Element Method (FEM) is tried here.

The adaptive FEM technique was proposed in early 1980s to reduce the discretized errors resulted from a FEM mesh. In this technique, the errors are first calculated to assess the accuracy of the solution. If the errors are large, the finite element model is then refined through redistributing the nodes (called R-version adaptive FEM), or reducing the size of elements (called H-version adaptive FEM), or increasing the order of the interpolation functions (called P-version adaptive FEM) [13], or using the combination ways. The new model is then re-analyzed and the errors in the new model are recalculated. The procedure is continued until the calculated errors fall below the specified permissible values. Hence, the adaptivity means that the FEM model refinement is based on the error distribution.

For standard problem with regular geometry several work has been done using adaptive FEM technique. However, there seems to be far less work being done directly for complex geometries. There are several challenges like selection of proper solvers when the condition number is very high and the advantage of using p-adaptive and h-adaptive. These issues are addressed in this research.

We would like to express our special thanks to the AFOSR, DOD for their sponsorship of this project. Especially the encouragement provided by Dr. Len Sakell, AFOSR in the initial stages is acknowledged.

R. Panneer Selvam & Zu-Qing Qu

Fayetteville, Arkansas, May 2004

ADAPTIVE NAVIER STOKES FLOW SOLVER FOR AEROSPACE STRUCTURES

Part I: P-Adaptive FEM for Flow Around Circular Cylinders

R. Panneer Selvam and Zu-Qing Qu

Department of Civil Engineering, University of Arkansas
4190 Bell Engineering Center, Fayetteville, Arkansas 72701

Final Report
AFOSR; Grant No.: F49620-00-1-0309

December 2002

Abstract

Predicting wind induced aerodynamic forces on aerospace structures is usually computationally expensive even for powerful computational facilities. For these unsteady computations, adaptive finite element technique may reduce the computer time and storage while keeping a desired accuracy. In this report, the p-version adaptive finite element method is implemented into a standard benchmark problem, the computational flow around a circular cylinder, to compute aerodynamic forces. The error distribution for velocity is first estimated. Then, the polynomial order of the interpolation function is changed continuously in accordance with the changing of the error distribution. The second through fifth orders of polynomials are considered for the velocity in the adaptive method. One order less of polynomial is used for pressure. The benchmark problem of the flow around a circular cylinder with Reynolds number of 1000 is considered to study the performance. The effects of the highest order of polynomials, error tolerance, and size of the element on the accuracy of the drag and lift coefficients are surveyed using this flow simulation. The results show that the accuracy of the velocity close to the cylinder affects the drag and lift coefficients and the error, 20% for example, far away from the cylinder does not have much effect on the accuracy of these coefficients.

1. Introduction

Prediction of wind-induced aerodynamic forces on aerospace structures has been mainly made by experimental methods involving the wind tunnel technique [1,2]. However, with the advent of supercomputers the tendency is gradually changing. Powerful computational facilities make it possible to deal with these physical quantities numerically. The Large Eddy Simulation (LES) based on the Smagorinsky's eddy viscosity model used by Murakami and his research group for wind engineering [3] is used in the modeling of drag crisis around a circular cylinder. Kakuda and Toskada [4], Kondo [5] and Tamura et al [6,7] used the no-

turbulence model to study the flow around a circular cylinder. Instead of using explicit methods [3-7], an implicit solution procedure to solve the Navier-Stokes (NS) equations using LES and finite element method was proposed by Selvam [8] to study flow around a cylinder. As we know, to solve the flow around a bluff structure is highly computationally expensive. For these unsteady computations, adaptive finite element method may reduce the computer time and storage.

The adaptive FEM technique was proposed in early 1980s to reduce the discretized errors resulted from a FEM mesh. In this technique, the errors are first calculated to assess the accuracy of the solution. If the errors are large, the finite element model is then refined through redistributing the nodes (called R-version adaptive FEM) [9,10], or reducing the size of elements (called H-version adaptive FEM) [11,12], or increasing the order of the interpolation functions (called P-version adaptive FEM) [13], or using the combination ways. The new model is then re-analyzed and the errors in the new model are recalculated. The procedure is continued until the calculated errors fall below the specified permissible values. Hence, the adaptivity means that the FEM model refinement is based on the error distribution.

The adaptive FEM technique has been discussed in detail during the past two decades. The results of such work are fruitful [14]. However, there seems to be far less work being done directly in the wind engineering which needs much more complicated procedures.

Choi and Yu [15,16] investigated the h-refinement for flows over a square cylinder. They used the penalty-function formulation to solve the NS equations. This procedure is not used commonly. Selvam [17, 18] applied the mesh enrichment technique (h-refinement) and p-refinement techniques to flows over a circular cylinder. He used primitive variable form to solve the NS equations.

In this report, the finite element modeling of the benchmark problem, the flow around a circular cylinder, is described in Section 2. Three-step advancement scheme for solving the N-S equations by LES is discussed. The error estimation

based on the velocity is proposed in Section 3. The details on the implementation of the p-adaptive technique are also discussed in this section. In Section 4, the simulation of a flow over a circular cylinder for Reynolds number of 1000 is considered. The effects of the highest order of polynomials, error tolerance, and size of the element on the accuracy of the drag and lift coefficients are surveyed using this flow simulation. Some useful conclusions will be given in that section. The computed drag and lift coefficients are also compared with the experimental and other computational results.

2. Finite Element Modeling of the Flow

In the following discussion Reynolds number R_e, drag coefficient C_d, lift coefficient C_l, and Strouhal number S_t are defined as

$$\begin{cases} R_e = VD/\nu \\ C_d = F_x/\left(0.5\rho V^2 D\right) \\ C_l = F_y/\left(0.5\rho V^2 D\right) \\ S_t = D/TV \end{cases} \tag{1}$$

where D is the diameter of the cylinder, V is the reference velocity, ν is the kinematic viscosity, F_x and F_y are the drag and lift force, T is the period of oscillation of the lift forces and ρ is the density.

The flow around a cylinder is represented by using the Navier-Stokes (NS) equations. The two and three-dimensional equations for an incompressible fluid in general tensor notation are as follows:

Continuity Equation: $$U_{i,i} = 0 \tag{2}$$

Momentum Equation: $$U_{i,t} + U_j U_{i,j} = -\left(p/\rho\right)_{,i} + \left[\nu\left(U_{i,j} + U_{j,i}\right)\right]_{,j} \tag{3}$$

where U_i and p are the velocity and pressure respectively. ρ is the fluid density. A comma represents one differentiation; t represent time. i =1, 2 and 3 mean variables in the x, y and z directions. To implement higher order approximation of the convection term the following expression is used instead of Equation (3)

$$U_{i,t} + U_j U_{i,j} - \theta\left(U_j U_k U_{i,j}\right)_{,k}/2 = -\left(p/\rho\right)_{,i} + \left[\nu\left(U_{i,j} + U_{j,i}\right)\right] \tag{4}$$

Depending upon the values of θ, different procedures can be implemented. For balance tensor diffusivity (BTD) scheme $\theta = \delta t$ is used; where δt is the time step size used in the integration. For streamline upwind procedure suggested by Brooks and Hughes [19], θ is considered as

$$\theta = \frac{1}{\max\left(\frac{|U_1|}{dx}, \frac{|U_2|}{dy}, \frac{|U_3|}{dz}\right)} \tag{5}$$

Here dx, dy, and dz are the control volume length in the x, y, and z directions; U_1, U_2, and U_3 are the velocities in the three directions. In this computation $\theta = \delta t$ is used. This has less numerical diffusion as compared with benchmark problems [20].

The NS equations are solved by using an implicit method, which is similar to Selvam [8], to eliminate the numerical stability restrictions. The three-step advancement scheme for Equations (2) and (4) is as follows:

Step 1: Solve for U_i from Equation (3). The diffusion and the higher order convection terms are considered implicitly to be in the current time and the first order convection terms are considered explicitly from the previous time step. The pressure is considered in the right hand side of the equation. This set of equations leads to a symmetric matrix and the preconditioned conjugate gradient (PCG) procedure is used to solve it.

Step 2: Solve for pressure correction from

$$\left(\delta p_{,i}\right)_{,i} = U_{i,i}/\delta t\,.$$

Step 3: Correct the velocity for incompressibility:

$$U_i = U_i - \delta t\left(\delta p_{,i}\right)$$

where U_i is not specified and update the pressure $p = p + \delta p$.

Suppose the element variables U_i and P are discretized using the shape functions N_U and N_p, i.e.

$$U_i^{'} = N_U u_i \,, \quad P^{'} = N_p p \tag{6}$$

where N_U and N_p are $1 \times n_e$ vectors; u_i and p are $n_e \times 1$ vector; n_e is the number of nodes for each element. Here, the velocity and pressure are approximated by using unequal order interpolation polynomial because if an equal order is used, the solution starts to diverge due to the violation of Babusk and Brezzi condition. Introducing Equation (6) into Equation (2) and using the regular Galerkin procedure the following matrix expressions are obtained:

$$m u_{i,t} + (d + v_1 + v_2) u_i + q p = f \tag{7}$$

where

$$m = \int_{\Omega_e} N_u^T N_u d\Omega \,, \quad d = \int_{\Omega_e} N_{u,j}^T (v + v_t) N_{u,j} d\Omega \,,$$

$$v_1 = \int_{\Omega_e} N_u^T u_j N_{u,j} d\Omega \,, \quad v_2 = \int_{\Omega_e} N_{u,i}^T (\theta U_i U_j / 2) N_{u,j} d\Omega \,, \quad q = \int_{\Omega_e} N_p^T N_{p,i} d\Omega \,. \tag{8}$$

In equation (7), m, d, v_1, v_2, and q are the mass, diffusion, convection due to the first and second order and pressure matrices of size $n_e \times n_e$. f is the $n_e \times 1$ vector which considers the given Neumann boundary conditions. After assembling all the element matrices, the equations of the finite element model becomes

$$M U_{i,t} + (D + V_1 + V_2) U_i + Q P = F \tag{9}$$

Equation (9) can be solved by using the aforementioned implicit method. The detailed solution procedure is similar to Selvam [8].

3. P-adaptive Finite Element Technique

The error estimate is one of the most important steps in the adaptive technique. It gives the error distribution in the present finite element grid, which can be used as indicator to refine the grid. The discretization errors of a finite element solution can be estimated by implicit or explicit method [21]. In structural engineering, stress recovery techniques are usually used to estimate the errors because they are much easier to implement in the programming. As the exact solution is generally not known, most of the approaches are concerned with posteriori error estimates.

In the present research, the error estimator based on the velocity is considered. The error is computed by considering the higher order solution to be exact and the lower order to be approximate. The difference between the two is defined as the error in the present grid, that is

$$\eta_i = \frac{\sqrt{\int_{\Omega_i} \left(v_p - w_{p-1}\right)^T \left(v_p - v_{p-1}\right) d\Omega_i}}{\sqrt{\int_{\Omega_i} v_p^T v_p d\Omega_i}} \tag{10}$$

where v_{p-1} and v_p are the velocity obtained from the finite element model using $p-1$ and p order interpolation functions, respectively. The order of polynomial to start with is quadratic (p=2) for velocity. Hence it is easy to compute from quadratic to linear, third order to quadratic, and fourth order to third order.

After the errors have been estimated and been found to be big, the next step is to refine the finite element model so as to reduce the errors. The accuracy of a finite element solution depends upon the shape and size of the elements and the order of the interpolation functions. Consequently, there are three methods to refine the finite element model. In this paper, p-version is implemented to the wind engineering.

P-refinement increases the order of the interpolation functions while keeping the mesh unchanged. Higher order elements generally provide a better description of the domain geometrically. They are particularly useful in regions where use of lower order elements would result in a mesh with poor aspect ratios in those elements [14]. From the point of view of solution accuracy, higher order elements are usually more accurate than the lower order elements.

The Hierarchical functions are applied here to increase of the interpolation function order because the method requires only the computation of the coefficients in rows and columns associated with the new enriched degrees of freedom, which together with the previously computed element matrices form a new stiffness matrix [22,23].

For one-dimensional elements, the linear shape functions are usually defined as

$$P_0(\xi)=\frac{1-\xi}{2},\quad P_1(\xi)=\frac{1+\xi}{2} \tag{11}$$

where ξ $(-1\le\xi\le1)$ is a non-dimensional coordinate. There are many ways to construct the higher order shape function [23]. The following shape functions are used in this report:

$$P_s(\xi)=\begin{cases}\xi^s-1 & s \text{ even}\\ \xi^s-\xi & s \text{ odd}\end{cases} \tag{12}$$

where s (≥2) is the degree of the introduced polynomial. The corresponding displacement function can be obtained as

$$u(\xi)=\sum_{i=0}^{n}a_iP_i(\xi) \tag{13}$$

Using these one-dimensional formulae, it is easy to derive two-dimensional shape functions [18]. These shape functions can be directly used in Equations (8) to compute the element matrices.

In the present work, four-node quadrilateral element is used to describe the geometry of the element. The polynomial order considered for velocity are from 2 to 5, which means in each direction interpolation function of the order of 2 through 5 are considered. The 1st through 4th orders of polynomial, which are one order less than that for velocity respectively, are considered for the pressure. The size of the element stiffness matrix for velocity varied from 9×9 to 36×36. The matrices are numerical integrated. The integration points considered at this time are 3×3 through 36×36.

4. Simulation of the Flow around a Circular Cylinder

The computational region of the wind around a circular cylinder is shown in Figure 1. Its length in the x and y directions are non-dimensionalized with respect to the diameter of the cylinder. The inlet velocities in these two directions are considered to be one and zero, respectively. On the top and bottom sides, the vertical velocities and the normal derivative of the velocities are set to be zero. The velocities are also

considered to be zero (no slip) on the surface of the cylinder. The flow around the circular cylinder for Reynolds number of 1000 will be considered in the following.

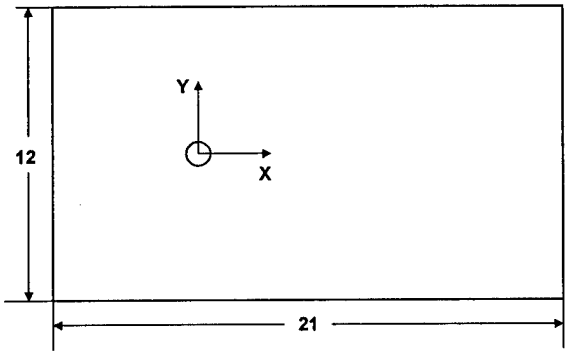
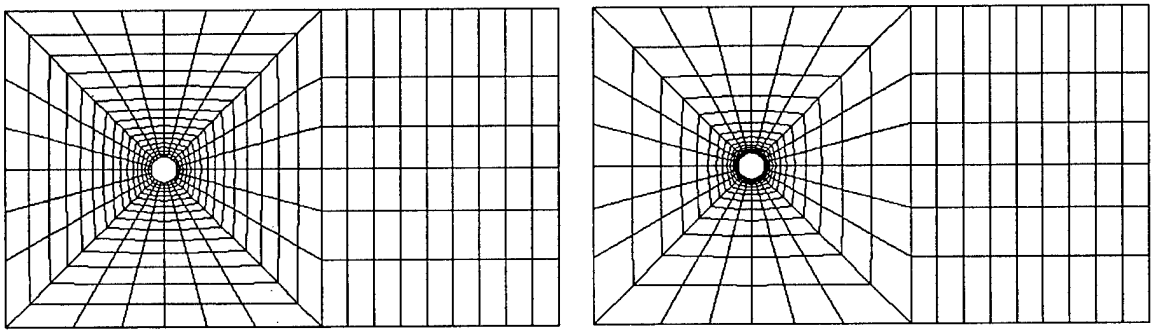


Figure 1. Computational region of the wind around the cylinder

4.1 Coarse Grid

Two coarse grids shown in Figure 2 are considered at first. Both grids have 414 $(24\times15+9\times6)$ elements and 447 nodes. The minimum spaces in the radial direction around the cylinder are, respectively, 0.098 and 0.020 in Grid A and Grid B. The highest orders of the interpolation polynomials considered are $P=2$ and $P=5$. Here and in the following, P denotes the highest polynomial order used in the adaptive refinement. P=2, for example, means that no refinement is applied even though the error is larger than the prescribed one. The flow is run 60 seconds. After 3 seconds, the error distribution will be evaluated every 0.2 second. The finite element model will be refined if the estimated error is higher than prescribed which is set as 10% at this time.



(a) (b)

Figure 2. Two coarse grids: (a) Grid-A; (b) Grid-B

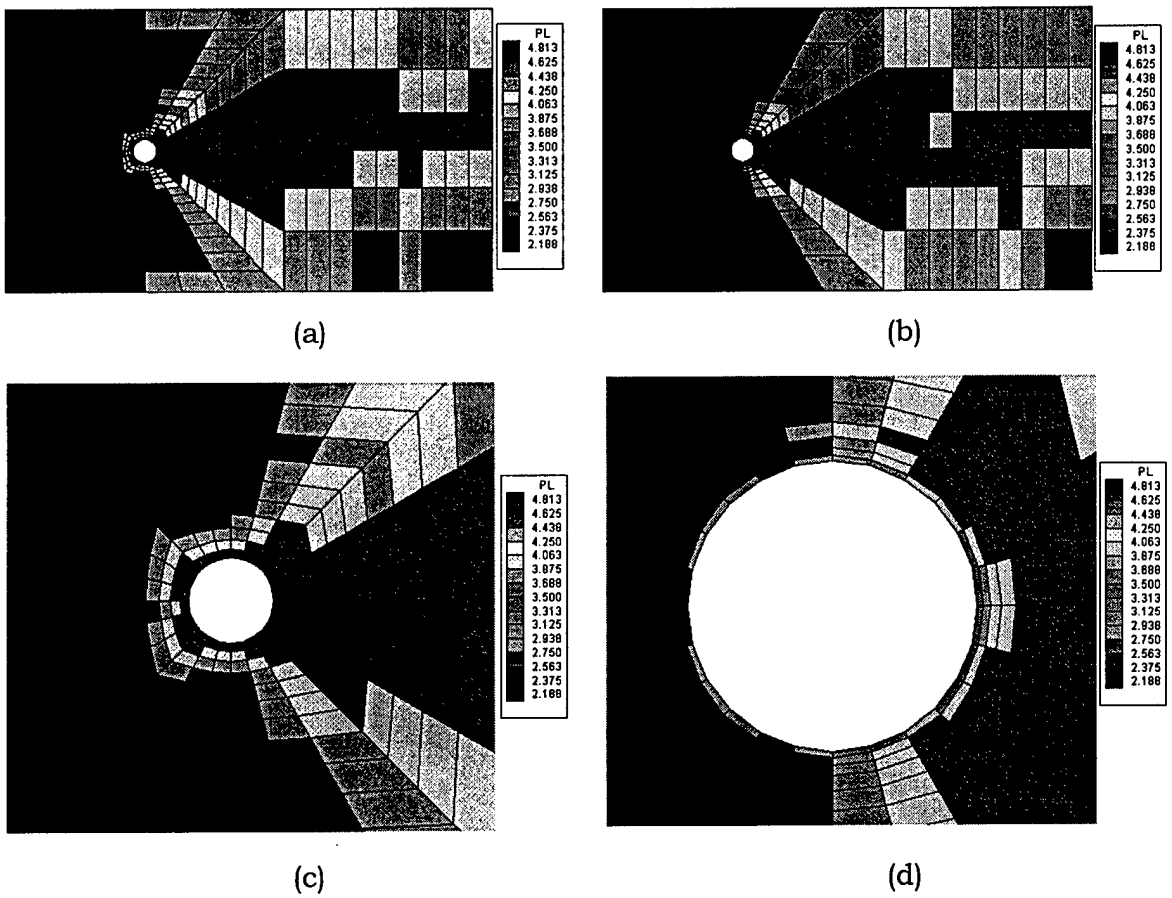


Figure 3. Distribution of the polynomial order: (a) Grid-A; (b) Grid-B; (c) Zoom of Grid-A; (d) Zoom of Grid-B

The distributions of the polynomial order at the end of 60-second are plotted in Figures 3(a) and 3(b). The corresponding error distributions are shown in Figure 4. The estimated errors are, respectively, 24% and 19% for the two cases. For most of the elements around the cylinder and on its right side, their orders have been increased by 3, 4, and 5 respectively. The 5th order of polynomial are only located on the vorticity shade area. For clarity purpose, the areas around the cylinder are zoomed in Figures 3(c) and 3(d) respectively. Since the size of the element in the radial direction used in Grid-A is much bigger than the Grid-B, the orders of many elements around the cylinder in the former case are increased by 5, while they are 2 through 4 in the latter case. As we know, if the orders of all the element are increased from 2 to 5, the number of the unknowns will be increased by 4 times, while they are 2.95 and 2.64 times for the Grid-A and Grid-B respectively.

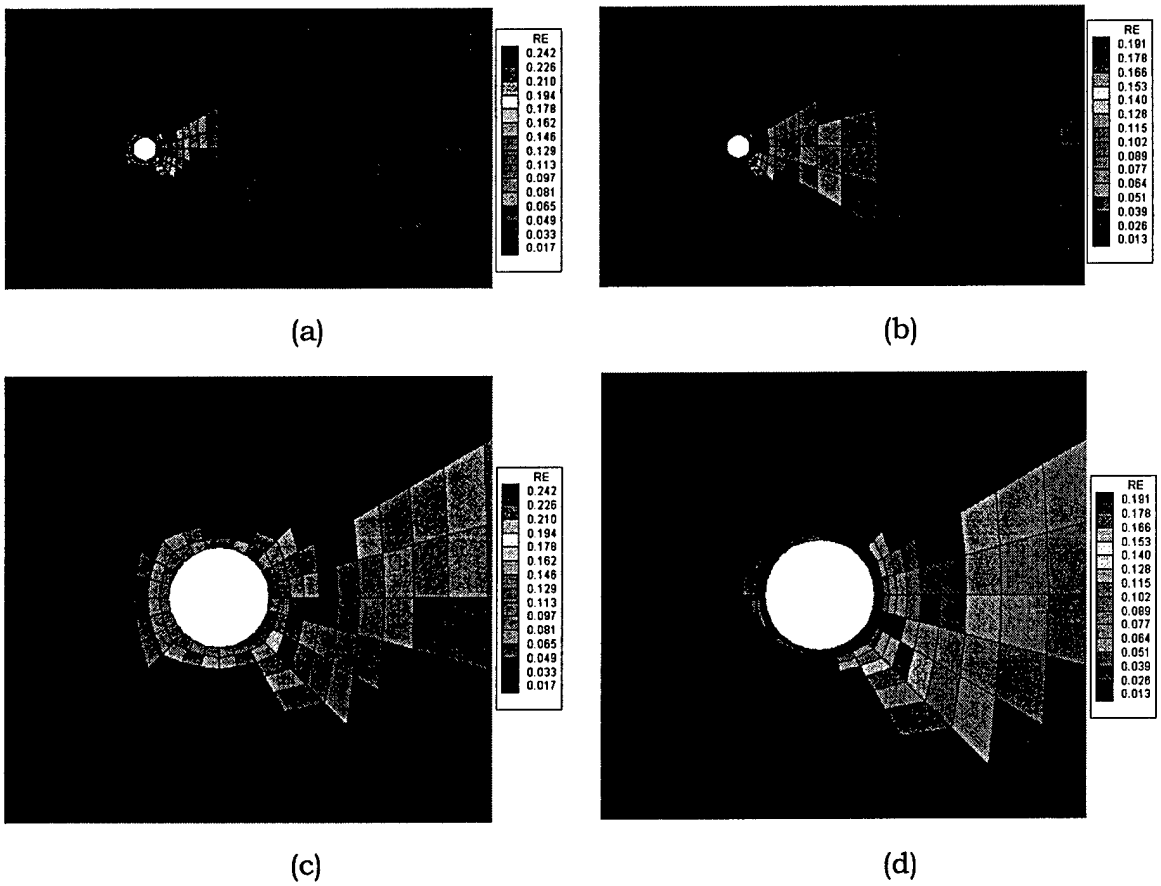
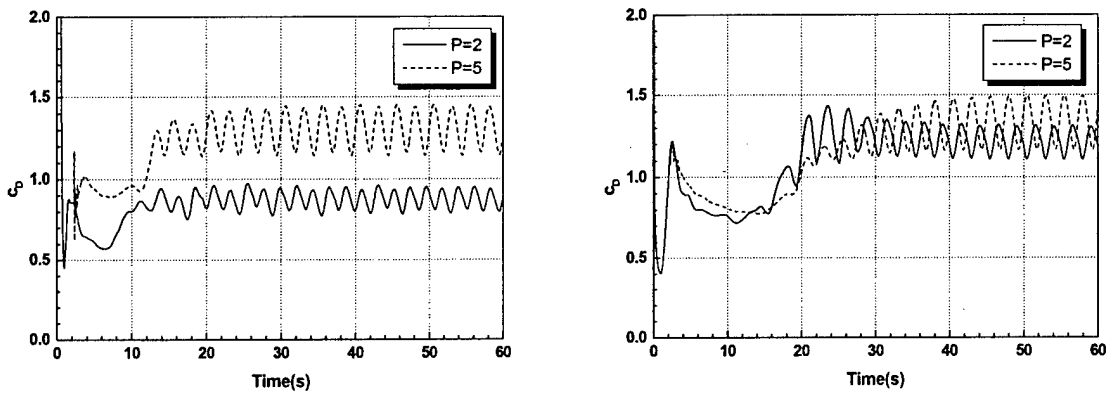


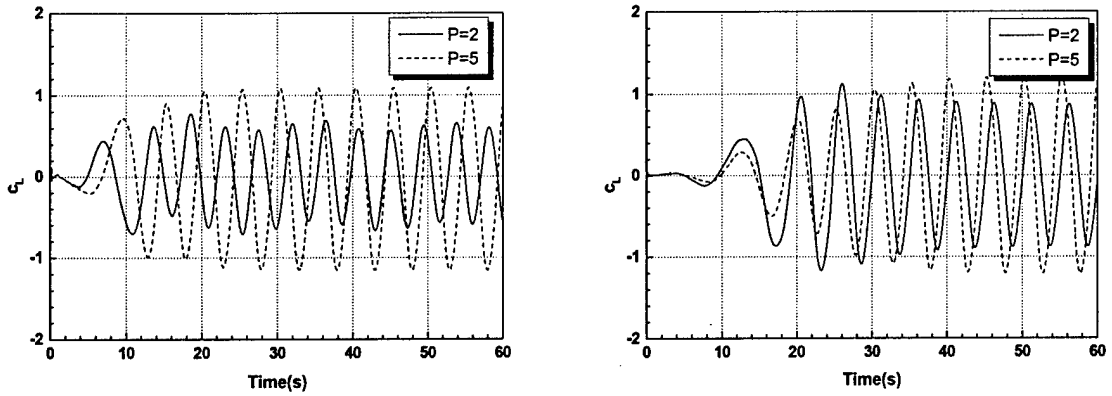
Figure 4. Distribution of the error: (a) Grid-A; (b) Grid-B;
(c) Zoom of Grid-A; (d) Zoom of Grid-B

The drag and lift force coefficients for both cases are shown in Figures 5 and 6 respectively. The results obtained from both $P=2$ and $P=5$ are included. The difference of the coefficients is significant especially for the Grid-A. The average, amplitude of the drag coefficients and the period of lift coefficient for the two cases are listed in Table 1. The averaged drag coefficients, for example, in Figure 5 (a) are 0.872 and 1.298 for $P=2$ and $P=5$ respectively. The former is much smaller than the reasonable value 1.42 which will be provided later. This means Grid-A is unreasonable to be used to perform the force coefficient analysis. After the p-adaptive technique is implemented, the averaged drag coefficient increases a lot. Its error reduces from 38.6% to 8.5%. The amplitudes of the drag coefficient have similar phenomenon.



(a) (b)

Figure 5. Drag coefficient for both cases: (a) Grid-A; (b) Grid-B



(a) (b)

Figure 6. Lift coefficient for both cases: (a) Grid-A; (b) Grid-B

Table 1. Drag coefficients and period for Grid-A and Grid-B

Grid	P	Average (c_D)	Amplitude (c_D)	Period (c_L)
Grid-A	2	0.872	0.074	4.35
	5	1.298	0.146	5.00
Grid-B	2	1.208	0.100	5.00
	5	1.329	0.165	5.00

Since the minimum space around the cylinder in the radial direction is reduced from 0.098 in Grid-A to 0.020 in Grid-B, the computed drag coefficient increases to 1.208 for $P=2$. After the refinement is considered, the coefficient becomes 1.329.

The difference between $P=2$ and $P=5$ becomes 9.1% while it is 32.8% in Grid-A. One of the reasons may be that the orders of the elements around the cylinder for Grid-B are not increased as much as that in Grid-A. Totally, the estimated errors are still a little high, around 20%, even though $P=5$ is considered in Grid-B. Therefore, the value of the drag coefficient 1.329 is a little smaller than 1.42. Consequently, more refine grid is necessary for accuracy purpose. The sizes of the element around the cylinder will be reduced both in the radical direction and its tangential direction.

4.2 Refined Grid

Three finite element grids shown in Figure 7 are considered. The properties of them are listed in Table 2. The flow is also run 60 seconds. After 3 seconds, the finite element model will be refined every one second.

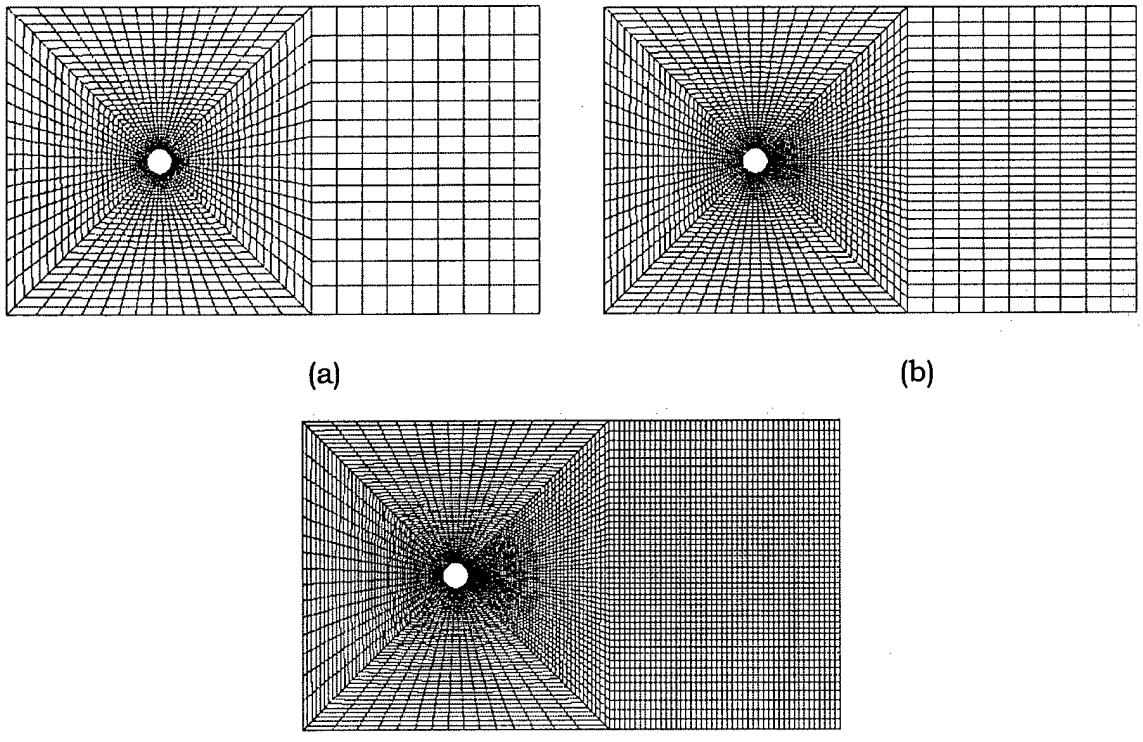


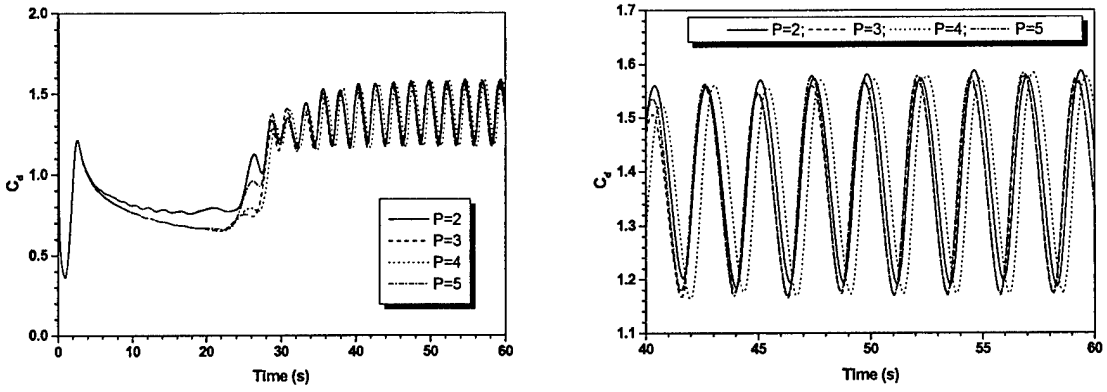
Figure 7. Three finite element grids: (a) Grid-I; (b) Grid-II; (c) Grid-III

Table 2. Properties of the three finite element grids

Grid No.	No. of Elements	No. of Nodes	Eqs. of V/P	Min. Space
Grid-I	1995(60×31+15×9)	2064	8118/2064	0.020
Grid-II	3195(75×39+30×9)	3279	12948/3279	0.015
Grid-III	7335(90×39+45×45)	7470	29610/7470	0.012

Effects of the Highest Polynomial Order P

The drag and lift force coefficients computed using Grid-I are plotted in Figures 8 and 9. The drag coefficients between 40-second and 60-second are zoomed in the plot 7(b) for clear purpose. The prescribed error limitation is set as 10% at first. After the simulation runs 40 seconds, both the drag and the lift force coefficients become stable. It can be seen from Figures 8 and 9 that there is no much difference among the amplitudes of the drag and lift coefficients for different Ps. For accuracy purpose, only the last four periods of drag plots and three periods of lift plots are considered for comparison in the following.



(a) (b)

Figure 8. Drag coefficients for P=2, 3, 4, and 5 with Grid-I: (a) Full plot; (b) Locally zoomed plot

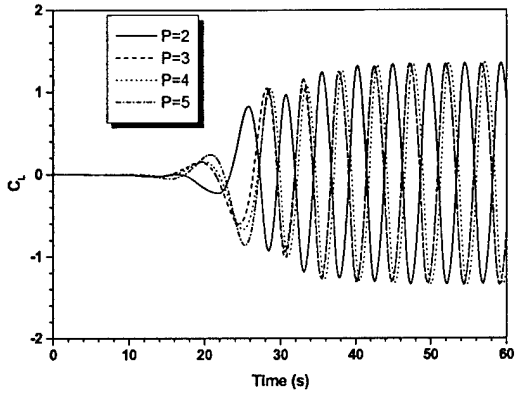
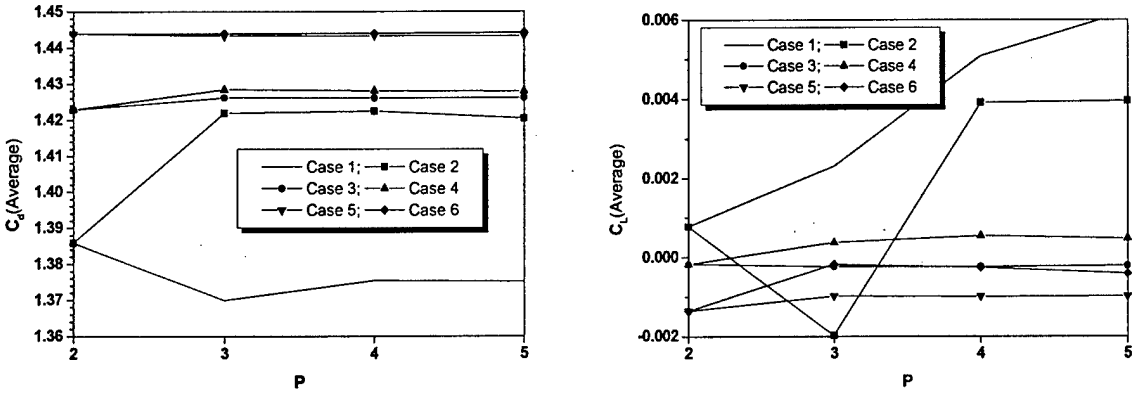


Figure 9. Lift coefficients for P=2, 3, 4, and 5 with Grid-I

The averages, amplitudes, and periods of the drag and lift force coefficients computed from the Grid-I are listed in Table 3. When the polynomial orders of all the elements are two, the average, amplitude, and period of the drag coefficient are 1.386, 0.195, and 2.389 respectively. When the highest polynomial order P is set as 3, the average and period decrease to 1.370 and 2.369 while the amplitude of the drag coefficient increase a little. After that, even though the highest polynomial P has been increased to 4 and 5, the average, amplitude, and period change very slightly. As for the lift force coefficient, it changes a little with the increase of the highest polynomial order P. When the prescribed error is 5%, these results change a lot from P=2 to P=3. Then, there is no much difference for the P=3, 4, and 5.

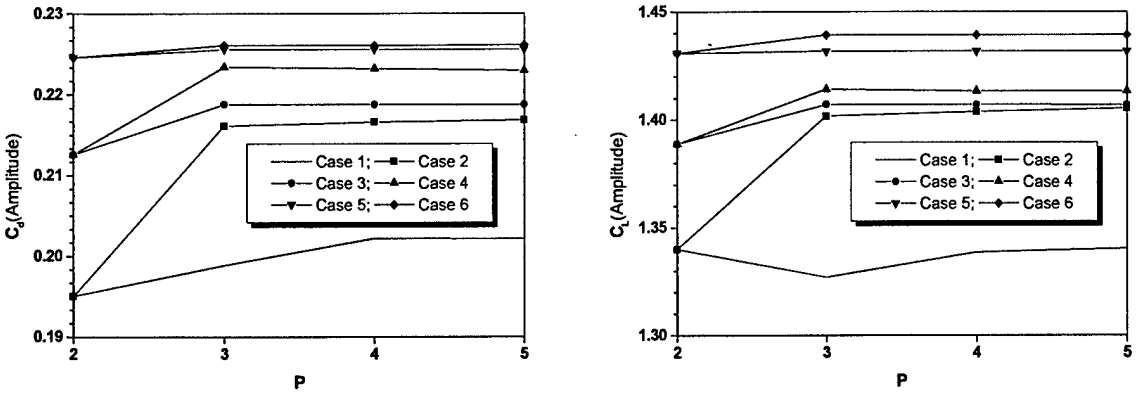
Table 3. Drag and lift coefficients for Grid-I

Prescribed Error	P	Drag Coefficient			Lift Coefficient		
		Average	Amplitude	Period (s)	Average	Amplitude	Period (s)
10%	2	1.386	0.195	2.389	0.001	1.340	4.768
10%	3	1.370	0.199	2.369	0.002	1.327	4.743
10%	4	1.375	0.202	2.368	0.005	1.339	4.743
10%	5	1.375	0.202	2.372	0.006	1.340	4.741
5%	3	1.422	0.216	2.352	-0.002	1.402	4.723
5%	4	1.422	0.217	2.357	0.004	1.404	4.720
5%	5	1.420	0.217	2.359	0.004	1.405	4.727



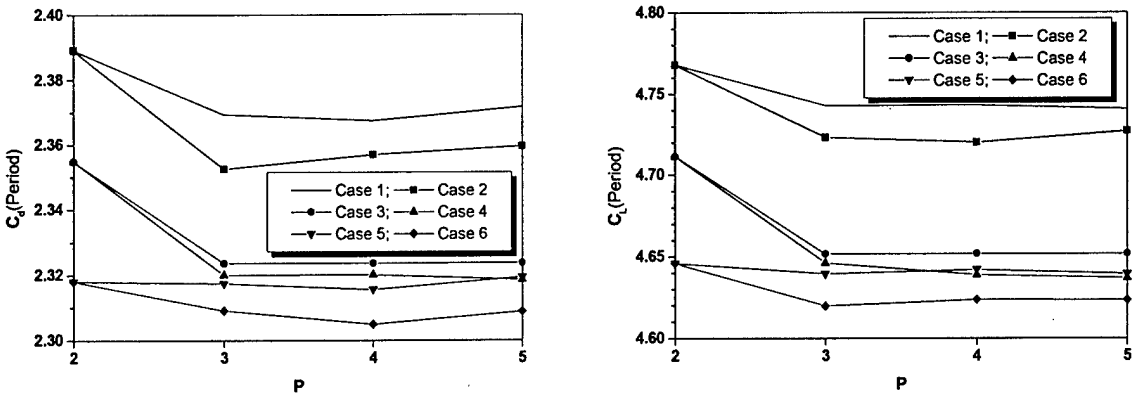
(a) (b)

Figure 10. Average of the drag coefficients for different cases: (a) drag coefficients; (b) Lift coefficients



(a) (b)

Figure 11. Amplitudes of the drag and lift coefficients for different cases: (a) drag coefficients; (b) Lift coefficients



(a) (b)

Figure 12. Periods of the drag and lift coefficients for different cases: (a) drag coefficients; (b) Lift coefficients

The averages, amplitudes, and periods of the drag and lift force coefficients computed from the Grid-II and Gird-III are plotted in Figures 10, 11, and 12. The results obtained from Grid-I are also plotted in these figures for comparison purpose. In these figures, the notations of the six cases are listed in Table 4. Obviously, when the highest polynomial order P is set as 3, the accuracy of the results obtained from Grid-II and Grid-III can improved some. However, the changes of the accuracy of the drag and lift coefficients are very insignificant for the P=4 and 5. Consequently, P=4 for the present case is usually enough.

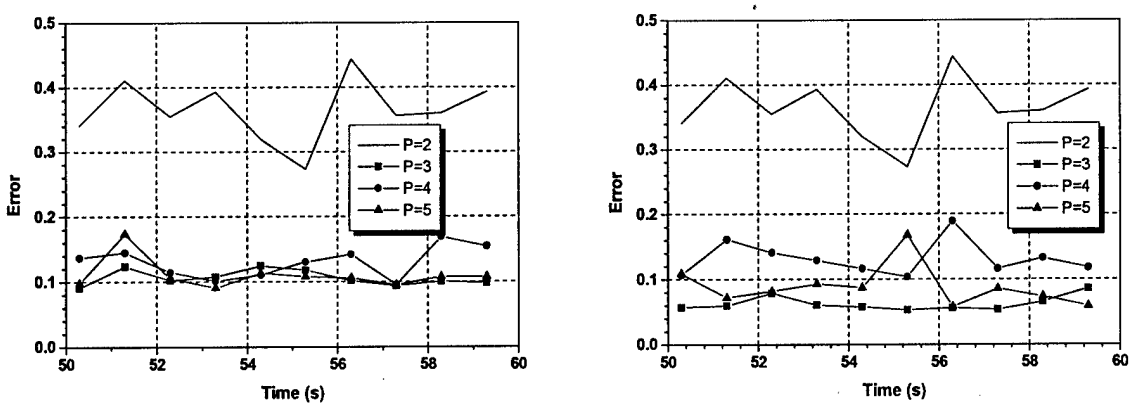
Table 4. Notation of the six cases

Case	Prescribed Error	Grid	Case	Prescribed Error	Grid
1	10%	Grid-I	4	5%	Grid-II
2	5%	Grid-I	5	10%	Grid-III
3	10%	Grid-II	6	5%	Grid-III

The maximum errors of the velocity computed between 50-second and 60-second are plotted in Figure 13. Obviously, the errors for the P=2 are very big and most of them are higher than 30% for Grid-I. When P is set as three, the accuracy of the velocity improves a lot. Unfortunately, the errors do not change much when P is

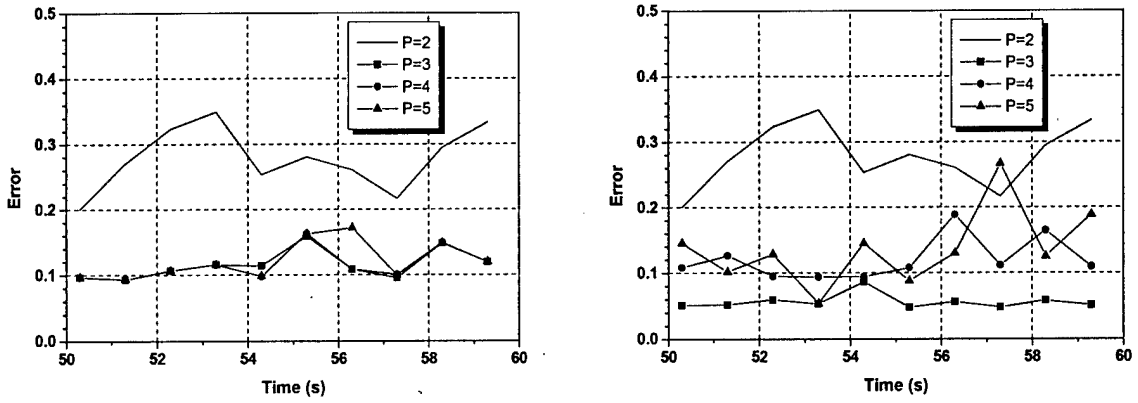
increase from three to four and then five. There are two possibilities. One is that the error estimation defined in this paper is unreasonable. The other is that higher order polynomials, 4th and 5th for example, do not have much effect on the accuracy of velocity. Sometimes, the errors reduce with the increase of the highest polynomial order P. The errors in plot 13(b) for P=4 and 5, for example, are higher than those resulted from P=3. One possible reason is that the orders of the interpolation polynomials are set as 2 for all elements.

It can be seen from plot 13(b) that the errors of the velocity obtained from P=4 are much higher those from P=3. However, the drag and lift force coefficients for both cases are very close that are indicated in Case 2 of plots 10(a) and 11. The same phenomenon happened for Grid-II. The accuracy of the velocity for P=4 and 5 is much lower than that for P=3 while the aerodynamic force coefficients are very close. If the error estimation defined in this report is reasonable, the aerodynamic force coefficients are insensitive to the accuracy of the velocity. This means coarse grids and low order interpolation polynomial can produce similar results with respect to the refined grids and high order polynomial.



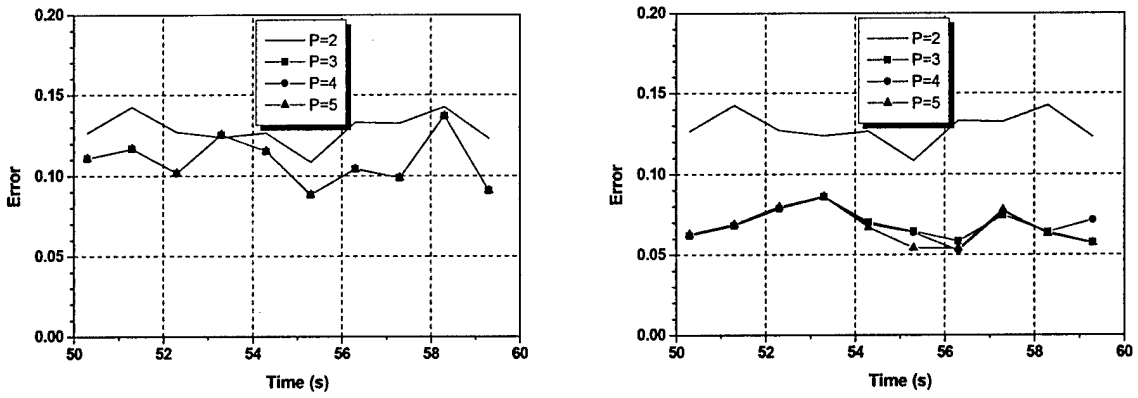
(a) (b)

Figure 13. Maximum errors between 50-second and 60-second for Grid-I:
(a) Error tolerance 10%; Error tolerance 5%



(a) (b)

Figure 14. Maximum errors between 50-second and 60-second for Grid-II: (a) Error tolerance 10%; Error tolerance 5%



(a) (b)

Figure 15. Maximum errors between 50-second and 60-second for Grid-III: (a) Error tolerance 10%; Error tolerance 5%

Together with the numbers of equations of velocity and pressure at the time 60-second, the run times for the three grids are listed in Tables 5, 6, and 7 for difference Ps. The run times mentioned here are not the exact and can only be used qualitatively.

Let us look at the results for Grid-I with the error tolerance is 10% that are listed in the up part of Table 5. When the P increases from 2 to 3, the number of equations

for velocity and pressure increase by 16.9% and 292% respectively. Unfortunately, the drag force coefficient reduces a lot which leads to a much higher error that P=2 because the "exact" coefficient for this grid is looked as 1.42. After that, even though the coefficient increase when P increases to 4 and 5, they are still lower than that for P=2. Hence, for this case the p-adaptive technique is absolutely unnecessary. For the case with error tolerance 5%, the result seems a little better. When P increases from 2 to 3, the number of equations of velocity and pressure increase by 28.3% and 293% respectively. The accuracy of the corresponding aerodynamic coefficients also increases significantly. Unfortunately, even though the number of equations increases a lot, the drag and lift coefficients change very insignificant. The similar phenomenon happens in the Grid-II and Grid-III.

Table 5. Number of equations and run time for Grid-I

Error	Polynomial Order	Eqs. Of Velocity At 60s	Eqs. of Pressure At 60s	Run Time (min)
10%	2	8118	2064	209
10%	3	9492	8094	421
10%	4	9758	9613	459
10%	5	10524	10447	670
5%	3	10414	8118	554
5%	4	10972	10076	661
5%	5	14579	14446	1855

Table 6. Number of equations and run time for Grid-II

Error	Polynomial Order	Eqs. Of Velocity At 60s	Eqs. of Pressure At 60s	Run Time (min)
10%	2	12948	3279	365
10%	3	14291	12909	630
10%	4	14334	14126	638
10%	5	14398	14236	646
5%	3	15994	12948	899
5%	4	16220	15527	881
5%	5	17693	17522	1368

Table 7. Number of equations and run time for Grid-III

Error	Polynomial Order	Eqs. Of Velocity At 60s	Eqs. of Pressure At 60s	Run Time (min)
10%	2	29610	7470	1125
10%	3	30735	29511	1346
10%	4	30742	30511	1405
10%	5	30781	30556	1338
5%	3	33698	29610	1889
5%	4	33770	33356	2052
5%	5	34084	33801	1948

Effects of the Error Tolerance

The program is run again for the Gird-I when the error tolerances are set as 2% and 1% respectively. The maximum errors computed between 50-second and 60-second with P=5 are plotted in Figure 16. The percentages of the elements with 2nd, 3rd, 4th, and 5th orders interpolation polynomial are listed in Table 8. They are resulted from P=5 and error tolerances are 10%, 5%, 2%, and 1% respectively. The corresponding averages, amplitudes, and periods for the error tolerances 2% and 1% are 1.421, 0.217, 2.351 and 1.420, 0.215, 2.359 respectively.

The averaged maximum errors between 50-second and 60-second are, respectively, 0.1102, 0.0885, 0.0702, and 0.0760 for η =10%, 5%, 2%, and 1%. The accuracy of the results improves a lot when the prescribed error changes from 10% to 5%. There is no much change when the error tolerance reduces from 5% to 2% and then to 1% even though the percentage of the elements with 5th order polynomial increases from 10.93% to 20.40% and then 27.07%. Again, it seems higher order polynomial does not have much effect on the accuracy of the force coefficients.

As for the aerodynamic forces, there is no much difference for the Grid-II and Grid-III when the error tolerance reduces. One reason is that the results obtained from Grid-II and Grid-III are already very close to the exact. Again, the higher order polynomial seems insignificant to the accuracy of the drag and lift force coefficients

for the present problem. Consequently, although the low error tolerance can increase the number of equations of velocity and pressure and the accuracy of the velocity somewhat, it can not increase the accuracy of the aerodynamic forces significantly.

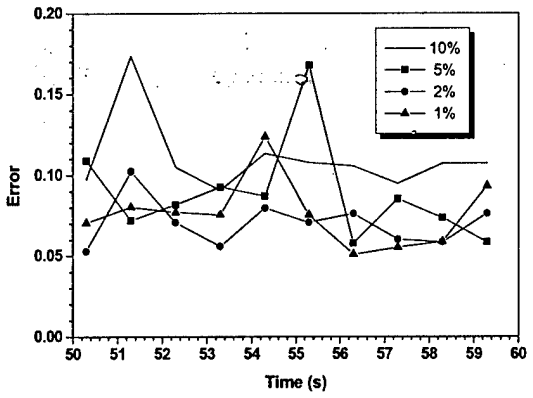
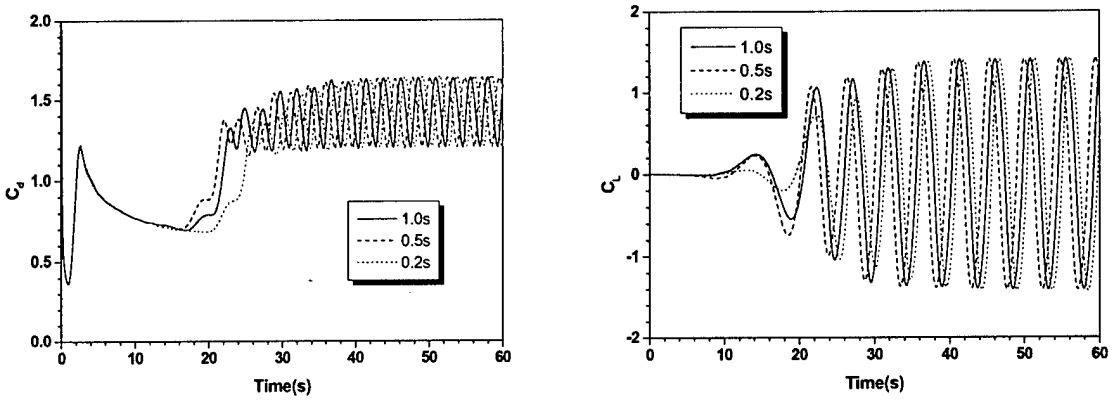


Figure 16. Maximum errors between 50-second and 60-second for different error tolerances with P=5

Table 8. Percentages of the elements with 2nd, 3rd, 4th, and 5th orders of polynomial

Error Tolerance	Order 2	Order 3	Order 4	Order 5
10%	83.26	11.78	3.21	1.75
5%	71.88	11.33	5.86	10.93
2%	62.01	12.58	5.01	20.40
1%	59.00	8.07	5.86	27.07

In the above discussion, the error distribution is evaluated every one second. To make sure that the refinement has been done completely within the 60 seconds, 0.5 second and 0.2 second are used for the time interval. The computed drag and lift coefficients are plotted in Figure 17. Except some time shift, the amplitudes and averages of the drag and lift coefficients are very close. This means that results provided above are reasonable.



(a) (b)

Figure 17. Drag and lift coefficients for different time interval of refinement: (a) Drag coefficients; (b) List Coefficients

Effects of the Grid

It can be seen from Figures 13(b), 14(b), and 15(b) that the cases of P=3, P=3, and P=3 or 4 or 5 have the highest accuracy of velocity for Grid-I, Grid-II, and Grid-III respectively. Hence, the aerodynamic forces of these three cases, which are listed in Table 9, are considered as the exact for each grid. For the Grid-I with P=3, the number of equations of velocity and pressure are 10414 and 8118. They are 15994, 12948 and 33698, 29610 for Grid-II and Grid-III with P=3 respectively. The differences of the aerodynamic forces among the three grids are very minor while the differences of the number of equations are very significant. Consequently, it is unnecessary to use highly refined grid to evaluate the drag and lift forces.

Table 9. The exact aerodynamic forces for the three grids

Grid	Average C_d	Amplitude C_d	Period (s) C_d	Amplitude C_L	Period (s) C_L
I	1.422	0.216	2.352	1.402	4.723
II	1.428	0.223	2.320	1.414	4.646
III	1.444	0.226	2.309	1.439	4.620

5.Conclusions

The p-version adaptive finite element method has been implemented into the computational flow around a circular cylinder to compute aerodynamic forces. The second through fifth orders of polynomials are considered for the velocity in the adaptive method. One order less of polynomial is used for pressure. Velocity is selected as the error estimator. A flow around a circular cylinder with Reynolds number of 1000 was simulated using this technique. The effects of the highest order of polynomials, error tolerance, and size of the element on the accuracy of the drag and lift coefficients are surveyed using this flow simulation.

When the grid is coarse, the p-adaptive technique is very efficient. It can improve accuracy significantly while the computed time does not increase much. For the refined grid, the results of the present simulation show that the higher orders of interpolation polynomials, P=4 and 5 for example, do not have much effect on the accuracy of the velocity as well as the drag and lift coefficients. The error of the velocity does not have much effect on the accuracy of the drag and lift force coefficients. If the error estimation defined in this report is reasonable, the aerodynamic force coefficients are insensitive to the accuracy of the velocity. The accuracy of the results improves a lot when the prescribed error changes from 10% to 5%. There is no much change when the error tolerance reduces from 5% to 2% and then to 1% even though the percentage of the elements with 5th order polynomial increases. The differences of the aerodynamic forces among the three refined grids are very minor while the differences of the number of equations are very significant.

The code will be applied to the practical aerospace problems, especially the AFOSR/DEPSCoR funded work on fluid-structure interaction of aerospace structures.

6. Acknowledgements

This study has been partially funded by Department of Defence, AFOSR under DEPSCOR program and Federal Highway Administration through Lendis Corporation, McLean, Virginia, University of Arkansas.

7.References

[1] A. Roshko, Experiments on the flow past a circular cylinder at very high Reynolds number, Journal of Fluid Mechanics, 10(1961) 345-356.

[2] H.J. Lugt, Vortex Flow in Nature and Technology, Wiley, New York, 1983.

[3] S. Murakami, A. Mochida, On turbulent vortex-shedding flow past 2D square cylinder predicted by CFD, Journal of Wind Engineering and Industrial Aerodynamics, 54/55(1995) 191-211.

[4] K.Kakuda, N.Tosaka, Numerical simulation of high Reynolds number flows by Petrov-Galerkin finite element method, Journal of Wind Engineering and Industrial Aerodynamics, 46&47(1993) 339-356.

[5] N.Kondo, Direct third-order upwind finite element simulation of high Reynolds number flows around a circular cylinder, Journal of Wind Engineering and Industrial Aerodynamics, 46&47(1993) 349-356.

[6] T.Tamura and K.Kuwahara, Numerical study of aerodynamic behavior of a square cylinder, Journal of Wind Engineering and Industrial Aerodynamics, 33(1990) 161-170.

[7] T.Tamura, I.Ohta, and K.Kuwahara, On the reliability of two-dimensional simulation for unsteady flows around a cylinder-type structure, Journal of Wind Engineering and Industrial Aerodynamics, 35(1990) 275-298.

[8] R.P.Selvam, Finite element modeling of flow around a circular cylinder using LES, Journal of Wind Engineering and Industrial Aerodynamics, 67&68(1997) 129-139.

[9] A.R.Diaz, N.Kikuchi, and J.E.Taylor,, Method of grid optimization for finite element methods, Computer Methods in Applied Mechanics and Engineering, 41(1983), 29-45.

[10] M.S.Shephard,. Approaches to the automatic generation and control of finite element meshes, Applied Mechanics Reviews, 41(1988), 169-185.

[11] L.Demkowicz, and T.Strouboulis, Adaptive finite elements for flow problems with moving boundaries, part I: variational principles and a posteriori estimates, Computer Methods in Applied Mechanics and Engineering, 46(1984), 217-251.

[12] P.Devloo, J.T.Oden, and T.Strouboulis, Implementation of an adaptive refinement technique for the SUPG algorithm, Computer Methods in Applied Mechanics and Engineering, 61(1987), 339-358.

[13] B.A.Szabo, Computation of stress field parameters in areas of steep stress gradients, Communications in Applied Numerical Methods, 2(1986), 133-137.

[14] L.-Y.Li,, Adaptive finite element methods: a review, Applied Mechanics Reviews, 50(1997), 581-591.

[15] C.K.Choi, and W.J.Yu,, Adaptive finite element wind analysis with mesh refinement and recovery, Wind and Structures, 1(1998), 111-125.

[16] C.K.Choi,and W.J.Yu, Adaptive refinement/recovery for analysis of wind around structure, Journal of Aerospace Engineering, 12(1999), 168-175.

[17] R.P.Selvam, Computation of flow over circular cylinder using p-adaptive FEM, International Symposium on Wind and Structures for the 21st Century, Chejudo, Korea, January 26-28, (2000).

[18] R. P. Selvam and Z.-Q. Qu. Adaptive p-finite element method for wind engineering, Wind and Structures, 5 (2002), 301-316.

[19] A.N.Brooks, T.J.R.Hughes, Streamline unwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations, Computer Methods in Applied Mechanics and Engineering, 32(1982), 199-259.

[20] R.P.Selvam, and H.Bosch, Finite element modelling of flow around bridges, Wind Engineering into the 21st Century, Rotterdam (1999), 1321-1327.

[21] I.Babuska, T.Strouboulis, C.S.Upadhyay, S.K.Gangaraj, and K.Copps, Validation of a posteriori error estimators by numerical approach, International Journal for Numerical Methods in Engineering, 37(1994), 1073-1123.

[22] A.Peano, A.Pasini, R.Riccioni, and L.Sardella, Adaptive approximations in finite element structural analysis, Computers and Structures, 10(1979), 333-342.

[23] O.C.Zienkiewicz, O.C., The Finite Element Method, McGraw-Hill Book Company, England, Vol.1, (1989).

ADAPTIVE NAVIER STOKES FLOW SOLVER FOR AEROSPACE STRUCTURES

Part II: H-Adaptive Advancing Grid Generation

R. Panneer Selvam and Zu-Qing Qu

Department of Civil Engineering, University of Arkansas
4190 Bell Engineering Center, Fayetteville, Arkansas 72701

Final Report
AFOSR; Grant No.: F49620-00-1-0309

February 2003

1. Introduction

The development of the advancing front mesh generation method is described in a series of papers by Peraire et al [1-4]. Important contributions to the development of the method have also been given by Lohner et al [5-7]. Although, two dimensional mesh generation methods, similar in concept to the advancing front method, have been used since at least the early seventies, the generalizations which make the advancing front method qualify as an automatic three dimensional mesh generation method were not devised until the mid-eighties by the previously referenced works by Peraire et al [3,4].

The characteristic feature of the advancing front method is that [7] elements and nodes are created simultaneously, with all elements residing behind a front which sweeps over the domain, starting from external as well as internal boundaries. The front consists of those mesh line segments that separates the discretized and undiscretized parts of the domain from each other. Hence, the initial front consists of the mesh line segments that make up the boundary discretization. A line segment is selected from the front and a new node is constructed so that a new triangular element is formed. As this procedure is repeated, the front advances over the domain and changes continuously in that old line segments are deleted and new added as triangles are created. The mesh is complete when the front is empty.

2. Generation of Initial Mesh

2.1. Background Grid

The problem of generating a mesh over a two dimensional region of arbitrary shape is considerably simplified if unstructured triangular meshes are employed. For the method to be described here, this process is started by constructing by hand a coarse background grid of 3-node triangular elements which completely covers the solution domain of interest [2]. The background grid is used to provide a piecewise linear spatial distribution for these parameters over the grid to be generated. Thus, at each node on the background grid, the stretching: node spacing δ , value of stretching parameter s and the direction of stretching α must be

specified. During the generation process the local values of these quantities will be obtained by linear interpolation, over the triangles of the background grid, between the specified nodal values. If δ is required to be uniform initially and no stretching is to be specified, then the background grid need only consist of a single element which covers the solution domain.

2.2. Generation of Boundary Nodes

Define the boundaries of the domain to be gridded. This is typically accomplished by spines in 2-D and surface patches in 3-D. The boundary of the solution domain is represented by the union of closed loops of curved segments and boundary nodes are placed at the points of intersection of these segments. For simply connected regions there is only one closed loop, whereas for multi-connected regions there will be as many internal loops as the number of openings inside the domain. The segments of exterior boundary are defined in an anti-clockwise manner while the segments of the interior boundaries are specified in a clockwise fashion. This means that, as the boundary curve is traversed, the region to be triangulated always lies to the left. Before beginning the process of generating triangles within the region of interest, the positioning of additional nodes on the boundaries of the region has to be performed. Each boundary segment is considered in turn and nodal points are generated on the boundary segments, with the spacing of the points being determined by interpolated values of δ, s and α [2]. This yields the initial front.

2.3. Triangle Generation

At the start of the process the front consists of the sequence of straight line segments which connect consecutive boundary nodes. During the generation process, any straight line segment which is available to form an element side is termed active, whereas any segment which is no longer active is removed from the front. Thus, while the domain boundary will always remain the same, the generation front will change continuously and has to be updated whenever a new element is formed. The following steps are involved in the process of generation a new triangle in the mesh.

2.3.1 Prepare the creation the new element

2.3.1.1 Select the base edge on which the element is to be constructed. If large variations in δ are present in the background grid then it is advantageous to look for the smallest active side, but if δ is constant or varying slowly the last active side in the front is used.

2.3.1.2 Determine the element size, i.e. the length of the two new edges of the so-called ideal element. Suppose the chosen side joins nodes A and B. Determine the local mesh parameters δ_M, s_M and a_M at the mid-point of AB by interpolating over the background grid.

$$\delta_M = \frac{\delta_E \cdot S_{MFG} + \delta_F \cdot S_{MGE} + \delta_G \cdot S_{MEF}}{S_{EFG}} \tag{1}$$

where S_{MFG}, S_{MGE}, S_{MEF} and S_{EFG} are the areas of the triangles MFG,MGE, MEF and EFG and they are defined in Figure 1. Make a local rotation of coordinates so that a_M lies along the x_1 axis and scale the x_1 coordinate by a factor s_M.

2.3.1.3 Define the spatial neighborhood, i.e. the region close to the base edge where the configuration of the front must be known to make the validation of the new element possible.

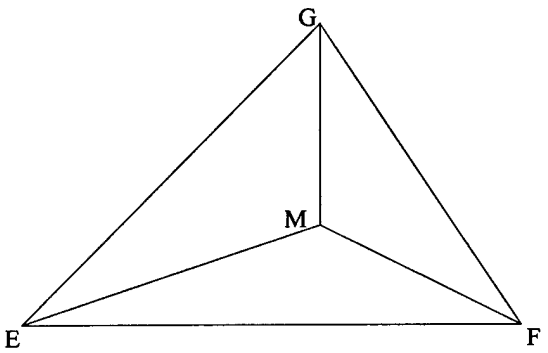


Figure 1. Area distribution

2.3.2 Create the new element

2.3.2.1 Determine the position of the ideal node so that the resulting element is as equilateral as possible while respecting the element size as defined by

background grid. In the new coordinate system, a triangle which is as regular as possible will be generated. Determine δ_1 according to

$$\delta_1 = \begin{cases} 0.55AB & \delta_M \le 0.55AB \\ \delta_M & 0.55AB \le \delta_M \le 2AB \\ 2AB & 2AB \le \delta_M \end{cases} \quad (2)$$

The inequalities used here are necessary to obtain ecometrical compatibility and to ensure that the elements with essive distoration are not generated. Different inequalities can be devised out values shown have worked well in practice. Note that not only the element size, but also the configuration of the front is considered when determining the position of the ideal node. Also note that the configuration of the front might prevent creation of the ideal node. If no node created go to 2.3.2.3.

2.3.2.2 Check the validity and the suitability of the ideal element. The validity check consists in ensuring that none of the newly created edges intersects with the front in the neighborhood. The suitability check consists in ensuring that the node and the edges of the element are sufficient far from nodes and edges in the front. If the ideal element satisfies the validity and suitability requirements go to 2.3.3.

2.3.2.3 Check if any existing active node in the neighborhood can be used to form the new element. A node should be positioned sufficiently close to the base edge to be considered a candidate node. Usually, the candidate nodes are defined as those lie within the circle with centre at C and radius nAB. (There is no unique choice for the value of n which should be adpoted, but the value 2 has been used for the program.) These nodes are odered according to their distance from C with the first node in the list being the closest to C. The validity and the suitability of the elements formed with the candidate nodes are checked in accordance with step 2.3.2.2. If several elements pass both these checks, select the best element as determined by the suitability check and then go to 2.3.3.

2.3.2.4 Create a set of try nodes on positions which are predetermined relative the base edge. The validity and the suitability of the elements formed with the try nodes are checked in accordance with step 2.3.2.2. If several elements

pass both these checks, select the best element as determined by the suitability check and then go to 2.3.3.

2.3.2.5 Check the validity and the suitability of the elements formed with the existing nodes in the neighborhood as well as with the try nodes in step 2.3.2.4. The The validity and the suitability of the elements formed with the try nodes nodes are checked in accordance with step 2.3.2.2. Among those elements that pass the validity check, select the best element as determined by the suitability check and then go to 2.3.3.

2.3.3 Update the front data

First, delete edges and nodes that no longer are members of the front. Second, add a new node and new edges to the front data structures. The, if the front is not empty go to 2.3.2.1 otherwise finish.

2.4 Implementations

2.4.1 Normalized Space

The construction of the ideal triangle is done in a transformed, so-called normalized space as proposed by Peraire et al [2]. This is not absolutely necessary but convenient in case of anisotropic mesh control, i.e. when the size of the triangle should be different in mutually orthogonal directions. Anisotropic mesh control is most easily specified by a node spacing δ , a value of stretching parameter s and a direction of stretching α . Note that we assume α to be a unit vector. The variation over the domain of the mesh characteristics is given by a mesh size function, for instance, a background mesh.

The first step of the coordinate transformation from model space to normalized space is a translation of the origin of the model space coordinate system to the midpoint of the base edge, x_M . In the second step, the coordinate system is rotated so that the x_1 axis is aligned with the stretch direction α . The third step is to scale with $1/s$ along the α direction by which a normalized space with isotropic mesh control is obtained. Hence, the ideal triangle is equilateral and the element size is δ in the normalized space. The coordinate transformation from model space to normalized space is most easily expressed as

$$\hat{N} = SRT = \begin{bmatrix} 1/s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{m,1} \\ 0 & 1 & -x_{m,2} \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where the transformation matrix $\hat{N}$ is composed of the translation matrix T, the rotation matrix R and the scaling matrix S. The mesh parameters s, δ and a defining the transformation are, for instance, evaluated at the midpoint of the base edge. θ is defined the angle from x_1 to $\hat{x}_1$ and

$$\theta = \begin{cases} arc\cos\left(\dfrac{x_{B,1} - x_{A,1}}{AB}\right) & \left(x_{B,2} - x_{B,1} \geq 0\right) \\ 2\pi - arc\cos\left(\dfrac{x_{B,1} - x_{A,1}}{AB}\right) & \left(x_{B,2} - x_{B,1} \geq 0\right) \end{cases} \tag{4}$$

Note that the transformation matrix defined by equation () assumes homogeneous coordinates which iin this context simply means that a two dimensional point $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$ is expressed as the triple $\boldsymbol{x}_h = \begin{bmatrix} x_1 & x_2 & 1 \end{bmatrix}^T$. Furthermore, note that the row vectors of R rotate into the new coordinate axes. The inverse transformation from the normalized space to model space is

$$\hat{M} = T^{-1}R^{-1}S^{-1} = \begin{bmatrix} 1 & 0 & x_{m,1} \\ 0 & 1 & x_{m,2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

2.4.2 Validity Check

Candidate nodes and triangles must satisfy certain so-called validity criteria to be accepted as member in the mesh. The criterion whether to accept a candidate node is that it should be positioned sufficiently far from existing nodes on the front. Here sufficiently far is $0.67\delta_{ab}$ which is the element size. Hence, the node and triangle validity criteria are as follows:

- A new node is valid and can be accepted as a member of the mesh if the distance from it to the closest node is larger than $0.67\delta_{ab}$.
- A triangle is vald and can be accepted as a member of the mesh if the two edges do not intersect the advancing front.

The node validity criterion is simply that a new candidate node should be at least the distance $0.67\delta_{ab}$ away from existing nodes on the front. To check the validity of a node is straightforward since it only consists of distance calculations. For efficiency reasons square root evaluations should be avoided. Another optimization is to check the distance in each coordinate direction separately in a filtering step by which nodes out of bounds are quickly discarded. Note that the neighborhood is large enough to contain all nodes of interest for the node validity check.

The element validity criterion is what a new element is valid if it does not intersect the front. To ensure the validity of an element is thus equivalent with to check if any front edge intersects any of the edges of the candidate element. Here we assume that the elements are straight sided triangles, and the validity check then reduce to determine whether bounded line segment intersects. Even though this is rather straightforward, the great number of intersections calculations that needs to be done for the construction of a mesh motivates a carefully designed algorithm. The core of the algorithm we propose is based on a line clipping algorithm described by Blinn [3]. Before we proceed with description of the algorithm for determining whether a front edge intersects a triangle in two dimensional space, we introduce some useful notation. Each triangle edge, E_i, I=1,2,3 define a line that divides the Euclidian plane into two half spaces which we call the positive half space, E_{i+} and the negative half space, E_{i-}. The triangle is oriented so that its interior is in E_{i+}, and the normal vector , n, to E_i is directed towards the interior of the triangle. The implicit equation of the line E_i is

$$b_x^{E_i} = n^T\left(x - x_{E_i}\right) \tag{6}$$

where x_{E_i} is a point on the line, x is an arbitray point, and $b_x^{E_i}$ is the so called boundary coordinate of the point x with respect to the line E_i. The following relations holds for the boundary coordinates

$$b_x^{E_i} \begin{cases} > 0 & (x \in E_{i+}) \\ = 0 & (x \in E_i) \\ < 0 & (x \in E_{i-}) \end{cases} \tag{7}$$

Note that $\left|b_x^{E_i}\right|$ equals the perpendicular distance between the point x and the line E_i if n is a unit vector. Also note that we use the term line and edge to distinguish between the unbounded and the bounded line, respectively.

A three layered filtering algorithm for checking whether a candidate triangle, t, intersects a front edge, e, is defined as follows:

1. Check if both endpoints of e are contained in the one and same negative half space as defined by any of the triangle edges E_i. If that is no intersection between e and t, otherwise go to 2.
2. Check if one or both endpoints of e are in the interior of t. If that is the case, there is an intersection between e and t, otherwise go to 3.
3. Check for intersection between e and E_i, where I=1,2. If there is no intersection, it can be finally concluded that e and t does not intersect.

Step 1 is based on the observation that intersection can be ruled out if both endpoints of the front edge are in the negative half-space of at least one of the triangle edges. Note that the boundary coordinates for the front edge are saved as they are calculated since they also are used in steps 2 and 3. Thus, the front edge and the triangle do not intersect if the boundary coordinates for both endpoints of the edge are contained in the triangle. A point is contained in the triangle if and only if the point is in the positive half-space for each one of the triangle edges. This corresponds to that all three boundary coordinates for a front edge (one for each E_i) are positive. Step 3 is reached only if the front edge intersects at least one of the straight lines defined by the triangle edges E_i (not necessarily within the boundaries of the triangle edge). The check in step 3 is based on that there is an intersection between a line segment in parametric form

$$x = x_0 + s(x_1 - x_0) \tag{8}$$

and a triangle edge E_i if and only if the boundary coordinates $b_{x_0}^{E_i}$ and $b_{x_1}^{E_i}$ have opposite signs, i.e. the points x_0 and x_1 are on different sides of the line E_i. The point of intersection between the line segment and the line is then obtained from equation (6) and (7), viz.,

$$s = \frac{b_{x_0}^{E_i}}{b_{x_0}^{E_i} - b_{x_1}^{E_i}} \tag{9}$$

An intersection s is calculated for each line E_i for which the boundary coordinates $b_{x_0}^{E_i}$ and $b_{x_1}^{E_i}$ have opposite signs. What remains, is to determine whether any of the intersections is within the boundaries of a triangle edge. This is accomplished by associating the parameters t_0 and t_1 with the points x_0 and x_1, respectively. The parameters t_0 and t_1 are initialized to zero and one, respectively. For each intersection calculated from equation (9), the parameter whose associated boundary coordinate is negative is updated, viz.,

$$\begin{cases} t_0 = \max(t_0, s) & \left(b_{x_0}^{E_i} < 0\right) \\ t_1 = \min(t_1, s) & \left(b_{x_1}^{E_i} < 0\right) \end{cases} \tag{10}$$

If t_0 ever becomes greater than t_1 $(t_0 > t_1)$ the line segment and the triangle does not intersect. Consequently, the line segment crosses the triangle if t_0 still less than t_1 $(t_0 < t_1)$ when e has been checked against all three E_i. Finally, note that the second and third step can be done simultaneously since the computations done in step two also must be done in step three (check for boundary coordinate with opposite signs). This is usually efficient since the inclusion of front nodes (step two) in a triangle is not so common.

2.4.3 Suitability Criteria

The satisfaction of the validity criteria ensures the creation of a topologically compatible mesh. However, the validity criterion does not ensure the creation of well shaped elements. To avoid the construction of ill shaped elements, it is necessary to control the distance between nodes and edges as the front advances over the domain. Based on the considerations above we try to create well behaved meshes without having to resort to global smoothing. To achieve this goal it is necessary to control the distance between nodes and edges before accepting any new node or edge. In fact, the suitability criteria are expressed as an angle criterion which is as follows:

- A new node is accepted as a member of the mesh if the smallest angle formed by the node and any front edge is larger than $\geq 30^{\circ}$.
- A new edge is accepted as a member of the mesh if the smallest angle formed by the edge and any front is $\geq 30^{\circ}$.

2.4.4 Control Line Discretization

Curve discretization is the least covered topic of mesh generation. Perhaps because the subject is viewed as trivial. Although, curve discretization algorithms have been proposed in References [3,6,9,10] and discussed by Frykestig in detail [8].

2.4.4.1 Straight Linear Interpolation

Straight line interpolation is the simplest case for the discretization of boundary lines. It is used for uniformed element size. The number of nodes or elements that should be created on the line is usually specified in this case. The nodal locations are determined through interpolation. Usually linear interpolation which results in an equal subdivision of the line is used.

When the number of elements, N_e, is defined, the element length is

$$l = \frac{s_L}{N_e} \tag{11}$$

where s_L is the total length of the boundary line. If the length of elements, l^*, is specified, the number of elements is

$$N_e = \left[\frac{s_L}{l^*} + 0.5\right]_{Int} \tag{12}$$

Then equation (11) is used to calculate the actual element length. The locations of node n_i is

$$s_i = \sum_{j=1}^{i} l, \quad i = 1,2,\cdots,N_e \tag{13}$$

2.4.4.2 Arithmetic series expansion

The use of arithmetic [11] and geometric [10] series expansions is quite common for line discretization. In these approaches, it is required to specify the element size

at one or both endpoints of the line, and, possibly, the number of elements to be created on the line. These methods are usually used in the case that the element size changes slowly.

Consider a line with prescribed element lengths l_1^* and l_2^* at the two endpoints. They can be obtained from the background definition. An arithmetic series line discretization is defined as

$$l_i = l_{i-1} + d = l_1^* + (i-1)d \tag{14}$$

$$s_L = \sum_{i=1}^{N_e} l_i = \frac{N_e(l_1^* + l_2^*)}{2} = \frac{N_e\left[2l_1^* + (N_e - 1)d\right]}{2} \tag{15}$$

where d is a constant for the line that determines the mesh grading. The number of elements on the line is solved from equation (15) as

$$N_e = \left[\frac{2s_L}{l_1^* + l_2^*} + 0.5\right]_{Int} \tag{16}$$

The lengths of the elements are then obtained as

$$l_i = l_1^* + \frac{2(i-1)}{N_e - 1}\left(\frac{s_L}{N_e} - l_1^*\right), \quad i = 1,2,\cdots,N_e \tag{17}$$

We note that l_{Ne}, calculated from equation (17), is likely to differ from the specified element size l_2^*. The residual is $r = l_2^* - l_{Ne}$. It will satisfy $l_{Ne} = l_2^*$ by distributing r to the interior elements, i.e.

$$l_i = l_i\left(1 - \frac{r}{s_L - l_1^* - l_2^*}\right), \quad 2 \le i \le Ne - 1 \tag{18}$$

2.4.4.3 Meshing size function approach

In this method, the spacing function $\delta(s)$ is defined over the line segment. It defines the element size, or the distance between two neighboring nodes at the point given by $x(s)$ equivalently. Since $\delta(s)$ can be thought on as having the unit *lenght/node*, the reciprocal $1/\delta(s)$ can be interpreted as a node density. The number of element which need to be created along the boundary line is calculated by direct integration [3,9]

$$A_e = \int_0^{s_L} \frac{1}{\delta(s)} ds \tag{19}$$

Taking N_e equal to the nearest integer to A_e. The nodal positions, s_i, are calculated from

$$i = \frac{N_e}{A_e} \int_{s_0}^{s_i} \frac{1}{\delta(s)} ds \tag{20}$$

Suppose the element sizes at both endpoints are δ_1^* and δ_2^* respectively. The element size is then assumed to vary linearly between the two endpoints. The spacing function $\delta(s)$ is

$$\delta(s) = \delta_1^* + \frac{\delta_2^* - \delta_1^*}{s_L} s \tag{21}$$

Substituting equation (21) into equations (19) and (20), one has

$$A_e = \frac{s_L}{\delta_2^* - \delta_1^*} \ln\left(\frac{\delta_2^*}{\delta_1^*}\right) \tag{22}$$

$$s_i = s_0 + \frac{\delta_1^* s_L}{\delta_2^* - \delta_1^*} \left(e^{i \ln(\delta_2^*/\delta_1^*)/N_e} - 1\right), \quad i = 1,2,\cdots,N_e \tag{23}$$

2.4.5 Locating Query Point M

In the h-version adaptive finite element method, the finite element gird is generated adaptively according to the error in the computation. Usually, the adaptive mesh is generated by considering the current computational gird as a background gird [2]. Because the number of elements in the background mesh is large, some consideration must be given to the search problem of finding the element in the background mesh in which the query point M is located.

2.4.5.1 Searching Algorithms

Algorithm I

This algorithm requires, for each element e of the background grid, the knowledge of the three surrounding elements which have sides in common with element e. Given the coordinates of M and a starting element of the background grid, the three area-coordinates of M are determined. If each area coordinate lies between zero and one, then the element contains the point M. If not, the node for

which the area coordinate is a minimum is found and this indicate the nest element to be checked. As shown in Figure 2, if the area coordinates L_1, L_2 and L_3 of element e are evaluated at M and $L_1 < L_2 < L_3$, the next element to be checked is element e_1. In this manner, the necessary of searching over all the elements in the background grid is avoided.

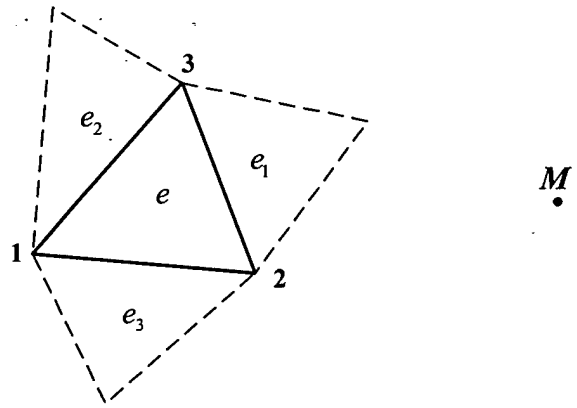


Figure 2. Searching algorithm to locate point M on the background grid

Algorithm II

Another possibility is to store the nodes of the background mesh in an actree as proposed by Lohner [5]. Every time the mesh size is queried at some point, the node of the background mesh closest to the query point is found by searching the octree. Once the closest point has been found, it is straightforward to find the element containing the query point assuming that the adjacency relationships between nodes and elements in the background mesh are available. The details of the octree may be found in References 5,8.

2.4.5.2 Locating the Point M

Algorithm I - Peraire: [2]

If the three area-coordinates of M with respect to a triangle element in the background grid lie between zero and one, the element contains the point. This condition is equivalent to that the smallest area coordinate is non-negative. It can be proven simply. Suppose coordinate L_i is the smallest among the three

coordinates. Because $L_i \geq 0$ and $L_i + L_j + L_k = 1$, we have $L_j + L_k \leq 1$. With considering $L_j \geq L_i \geq 0$ and $L_k \geq L_i \geq 0$, the conclusion may be obtained.

Algorithm II - Frykestig [8: 151-152]

Each triangle edge, E_i, I=1,2,3 define a line that divides the Euclidian plane into two half spaces which we call the positive half space, E_{i+} and the negative half space, E_{i-}. The triangle is oriented so that its interior is in E_{i+}, and the normal vector, n, to E_i is directed towards the interior of the triangle. The implicit equation of the line E_i is

$$b_x^{E_i} = n^T\left(x - x_{E_i}\right) \tag{24}$$

where x_{E_i} is a point on the line, x is an arbitrary point, and $b_x^{E_i}$ is the so called boundary coordinate of the point x with respect to the line E_i. The following relation holds for the boundary coordinates

$$b_x^{E_i} \begin{cases} > 0 & (x \in E_{i+}) \\ = 0 & (x \in E_i) \\ < 0 & (x \in E_{i-}) \end{cases} \tag{25}$$

Note that $\left|b_x^{E_i}\right|$ equals the perpendicular distance between the point x and the line E_i if n is a unit vector. Also note that we use the term line and edge to distinguish between the unbounded and the bounded line, respectively.

For the line segment AB, one of its normal vectors $\boldsymbol{n}$ through the original may be expressed as

$$\boldsymbol{n}: (0,0) \rightarrow (y_A - y_B, x_B - x_A) \tag{26}$$

Therefore, equation (24) becomes

$$b_x^{E_i} = (y_A - y_B)(x - x_E) + (x_B - x_A)(y - y_E) \tag{27}$$

3. Adaptive Remeshing

The procedure outlined above enable an initial approximation to the steady state solution to be obtained for a given problem. The solution quality can be improved by

adaptively refining the mesh. This mesh adaptation is achieved by using the computed solution to determine "optimum" nodal values for δ, s, and a. The mesh is then regenerated with the initial computational mesh now acting as a background grid.

4. Numerical Examples

Example I

As shown in Figure 3, the size of the rectangle ABCD is $1\times2\times1\times2$. Four triangular elements are used for the background grid. At first, the uniformed grid size is used. Therefore, the input data are given by

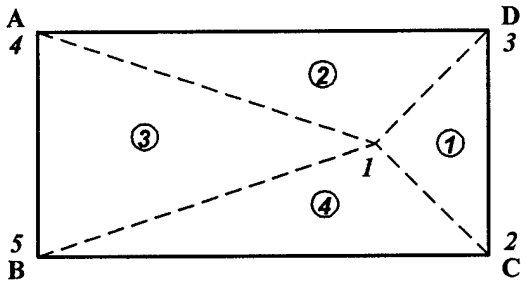


Figure 3. Rectangle to be meshed

INODE	NISEG	NBACK	NPOIN
4	4	4	5

Coordinates of the boundary nodes:

X	Y
0	1
0	0
2	0
2	1

Line segments defining the boundary:

1	2

2	3
3	4
4	1

Definition of the grid size through background element:

X	Y	Size
1.5	0.5	0.1
2	0	0.1
2	1	0.1
0	1	0.1
0	0	0.1

Connectivity of the background grid:

1	2	3
1	3	4
1	4	5
1	5	2

Finally, the grid is shown in Figure 4. If we let the size at point 1 be 0.01. The final gird is shown in Figure 5.

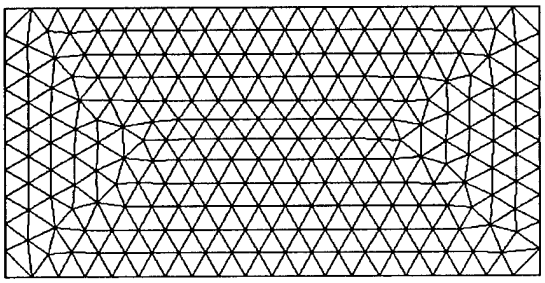


Figure 4. Grid of the Rectangle with uniform size required

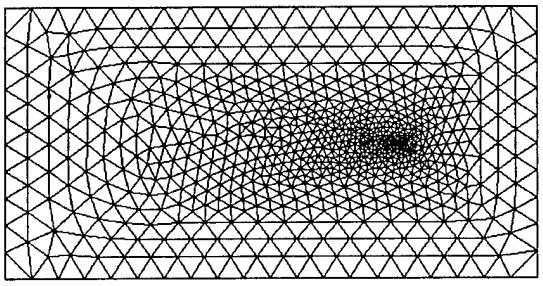


Figure 5. Grid of the Rectangle with non-uniform size required

Example II

As shown in Figure 6, the L-shaped domain is to be meshed. Four triangles are used to define the background. Two cases are considered. For both cases, the element sizes at nodes 1, 2, 3, and 4 are 10. The sizes at node 5 are 10 and 0.1 respectively. The input data can be given similarly. Finally, the grids are shown in Figures 7 and 8.

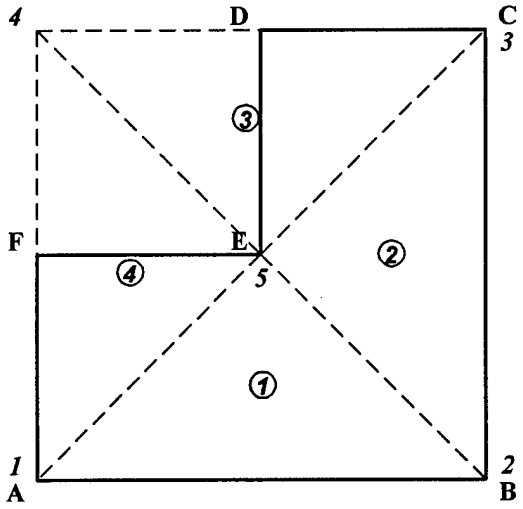


Figure 6. L-shaped domain

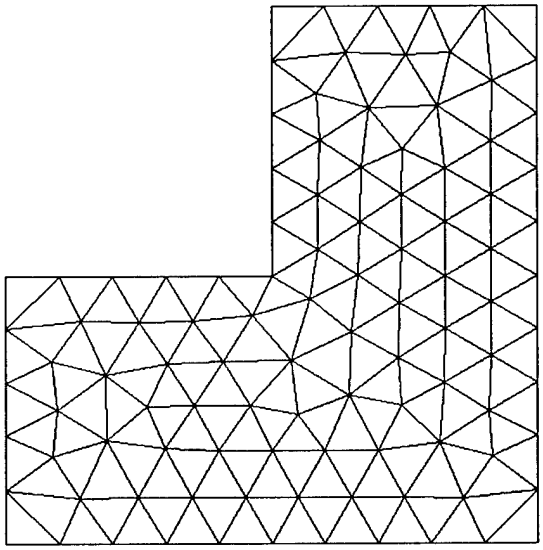


Figure 7. Grid of the L-shaped domain with uniform size required

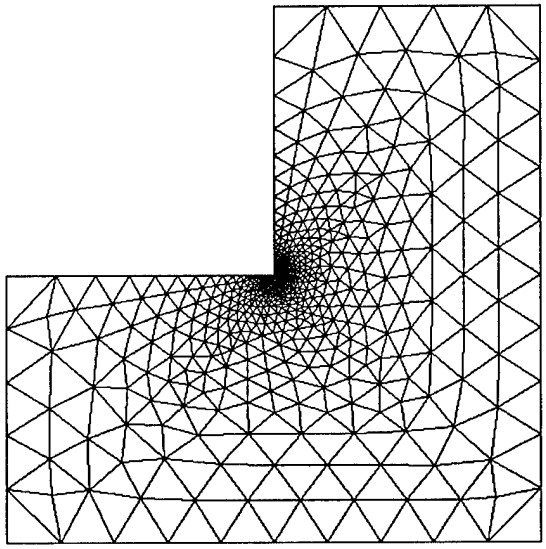


Figure 8. Grid of the L-shaped domain with uniform size required

Example III

Finally, a rectangle with one hole, as shown in Figure 9, is considered. The boundary of the hole is approximated by 16 line segments. Two cases for the background sizes are considered and listed in the following two tables. The finally grids are shown in Figures 10 and 11.

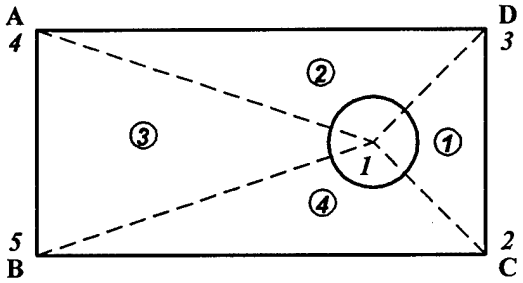


Figure 9. Rectangle with one hole

Definition of the grid size through background element (I):

X	Y	Size
1.5	0.5	0.01
2	0	0.1
2	1	0.1
0	1	0.1
0	0	0.1

Definition of the grid size through background element (II):

X	Y	Size
1.5	0.5	0.002
2	0	0.05
2	1	0.05
0	1	0.1
0	0	0.1

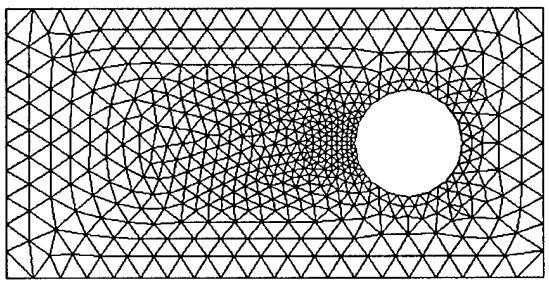


Figure 10. Grid for case 1 of the rectangle with hole

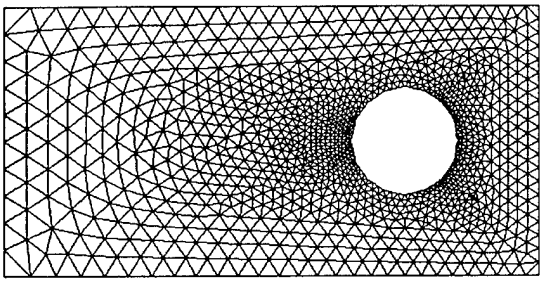


Figure 11. Grid for case 2 of the rectangle with hole

Example IV

In this example, the adaptive remeshing scheme provided in Section 3 is utilized. The initial grid is shown in Figure 12 which is uniform. After some time, the grid is regenerated depending on the error at each element. During the remeshing, the initial grid is looked as the background grid. The refined grid is shown in Figure 13.

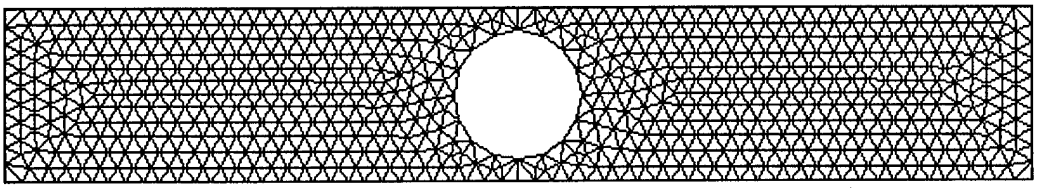


Figure 12. Initial Grid

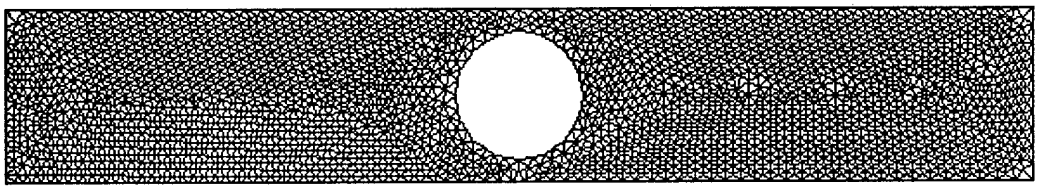


Figure 13. Refined Grid

5. Conclusions

The details for the generation of two-dimensional advancing front grid was provided. The schemes for the implementation of the generation were also described. Two efficient searching algorithms for locating the query point have been

presented. In the adaptive remeshing the current grid obtained from the advancing front algorithm is usually looked as the background grid. Several examples were provided to demonstrate the schemes in this report.

6. Acknowledgements

This study has been partially funded by Department of Defence, AFOSR under DEPSCOR program and Federal Highway Administration through Lendis Corporation, McLean, Virginia, University of Arkansas.

7. References

[1] Bonet, J. and Peraire, J., "An Alternating Digit Tree (ADT) Algorithm for 3D Geometric Searching and Intersection Problems," International Journal for Numerical Methods in Engineering, Vol.31, No.1, 1991, pp.1-17.

[2] Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations," Journal of Computational Physics, Vol.72, No., 1987, pp.449-466.

[3] Peraire, J., Peiro, J., Formaggia, L., Morgan, K. and Zienkiewicz, O. C., "Finite Element Euler Computations in Three Dimensions," International Journal for Numerical Methods in Engineering, Vol.26, No., 1988, pp.2135-2159.

[4] Peraire, J., Peiro, J. and Morgan, K., "Adaptive Remeshing for Three-Dimensional Compressible Flow Computations," Journal of Computational Physics, Vol.103, No., 1992, pp.269-285.

[5] Lohner, R., "Some Useful Data Structures for the Generation of Unstructured Grids," Communications in Applied Numerical Methods, Vol.4, No.1, 1988, pp.123-135.

[6] Lohner, R. and Parikh, P., "Generation of Three-Dimensional Unstructured Grids by the Advancing-Front Method," International Journal for Numerical Methods in Fluids, Vol.8, No., 1988, pp.1135-1149.

[7] Lohner, R., "Generation and Adaption of 3-D Unstructured Grids for Transient Problems," AGRAD Conference Proceedings No. 464, Application of Mesh Generation to Complex 3-D Configurations, 19.1-19.17, 1989.

[8] Frykestig, J., Advancing Front Mesh Generation Techniques with Application to the Finite Element Method. Ph.D. Dissertation, Department of Structural Mechanics, Chalmers University of Technology, 1994.

[9] Frey, W. H., "Selective Refinement: A New Strategy for Automatic Node Placement in Graded Triangular Meshes," International Journal for Numerical Methods in Engineering, Vol.24, 1987, pp.2183-2200.

[10] Jin, H., and Wiberg, N.-E., "Two-Dimensional Mesh Generation, Adaptive Remeshing and Refinement," International Journal for Numerical Methods in Engineering, Vol.29, 1990, pp.1501-1526.

[11] Buell, W. R. and Bush, B. A., "Mesh Generation - A Survey," Journal of Engineering Industrial, Vol.95, 1973, pp.332-338.

CONCLUSIONS

In the first part P-adaptive FEM has been developed and applied for flow around circular cylinder. The second through fifth orders of polynomials are considered for the velocity in the adaptive method. One order less of polynomial is used for pressure. Velocity is selected as the error estimator. A flow around a circular cylinder with Reynolds number of 1000 was simulated using this technique. The effects of the highest order of polynomials, error tolerance, and size of the element on the accuracy of the drag and lift coefficients are surveyed using this flow simulation.

When the grid is coarse, the p-adaptive technique is very efficient. It can improve accuracy significantly while the computed time does not increase much. For the refined grid, the results of the present simulation show that the higher orders of interpolation polynomials, P=4 and 5 for example, do not have much effect on the accuracy of the velocity as well as the drag and lift coefficients. The error of the velocity does not have much effect on the accuracy of the drag and lift force coefficients. If the error estimation defined in this report is reasonable, the aerodynamic force coefficients are insensitive to the accuracy of the velocity. The accuracy of the results improves a lot when the prescribed error changes from 10% to 5%. There is no much change when the error tolerance reduces from 5% to 2% and then to 1% even though the percentage of the elements with 5th order polynomial increases. The differences of the aerodynamic forces among the three refined grids are very minor while the differences of the number of equations are very significant.

In the second part h-adaptive is investigated by first developing grid generation program. The details for the generation of two-dimensional advancing front grid was provided. The schemes for the implementation of the generation were also described. Two efficient searching algorithms for locating the query point have been presented. In the adaptive remeshing the current grid obtained from the advancing front algorithm is usually looked as the background grid. Several examples were provided to demonstrate the schemes in this report. Currently work is in progress

to use the grid generator to solve the flow around circular cylinder and study the issues in using h-adaptive comparing to p-adptive. Also work is under progress to develop robust solvers. Some work is published in the following publication:

Selvam, R.P., Computational issues in solving the incompressible NS equations (invited paper), Proceedings of the 2nd International Conference on Fluid Mechanics & Fluid Power, P.C. Jain et. al. (Ed.), Ajay Printers & Publishers, IIT Roorkee, India, Dec. 12-14, 2002 , Volume 1, pp. 1-6