46th AIAA Aerospace Sciences Meeting and Exhibit
7 - 10 January 2008, Reno, Nevada

AIAA 2008-532

An Embedded Cartesian Grid Euler Solver with Radial Basis Function for Boundary Condition Implementation

L. Carolina[*] and H. M. Tsai [†]

National University of Singapore, Singapore 119260, Republic of Singapore

F. Liu[‡]

University of California, Irvine, Irvine, California 92697-3975

A Cartesian grid approach for the solution of the Euler equations within the framework of a patched, embedded Cartesian field mesh is described. As Cartesian grids are not necessarily body-aligned, an accurate representation for the surface boundary is important. In this paper a gridless boundary treatment using a cloud of nodes in the vicinity of the body combined with the multiquadric radial basis function (RBF) for the conserved flux variables for boundary implementation is proposed. In the present work, the RBF is applied only at the boundary interface, while a standard structured Cartesian grid approach is used everywhere else. Flow variables for solid cell centers for boundary condition implementation are determined via the use of reflected node involving a local RBF fit for a cloud of grid points. RBF is well suited to approximate multidimensional scattered data without any mesh accurately. Compared to the least-square method, RBF offers greater flexibility in regions where point selection may be very limited since the resulting matrix will be non-singular regardless of the sampling point's location. This is particularly important in the context of computations involving complex geometries where eligible points selected may be very close to one another. It is also shown that it provides similar accuracy with less cloud of points. The use of a Cartesian field mesh for the non boundary regions allows for effective implementation of multigrid methods, and issues associated with global conservation are greatly mitigated. Several two and three-dimensional problems are presented to show the efficiency and robustness of the method.

Nomenclature

$\vec{n}$	Normal vector
ρ	Density
E	Total energy
H	Total enthalphy
P	Pressure
C_p	Pressure coefficient
N	Number of supporting nodes
λ	RBF coeffient
ϕ	RBF function
c	Shape parameter
R	Radius of curvature

[*]Associate Scientist, Temasek Laboratories, Kent Ridge Crescent.
[†]Senior Research Scientist, Temasek Laboratories, Kent Ridge Crescent. Member AIAA.
[‡]Professor, Department of Mechanical and Aerospace Engineering. Associate Fellow AIAA.

I. Introduction

Mesh generation is one of the most problematic parts in Computational Fluid Dynamics (CFD) computation involving complex geometry. Existing methods to discretize the flowfield can be categorized in three groups, i.e. unstructured meshes, body-fitted curvilinear meshes, and Cartesian meshes. The traditional unstructured grid is normally constructed from triangles in 2D or tetrahedrals in 3D.[2,27] This method is suitable for complex configurations since the cells can be oriented in arbitrary direction to adjust the geometry. However, the computational cost and time for larger-scale problems are generally higher than the structured mesh approach. The body-fitted curvilinear grid is nowadays widely used for flow computation.[1,24] Its main benefit is that the boundary conditions can be implemented easily because of the body-aligned nature of mesh. Unfortunately, it is often a challenge to generate a mesh around complex configuration.

Recently, there is renewed interest in using Cartesian grid method.[5,7,8] Compared to body-fitted grid approach, its main advantage is the ease of grid generation when dealing with complex geometries. Additionally, terms involving grid transformations are not required in the computation. When compared to unstructured grid, grid connectivity information is obvious and thus it needs no additional storage requirements. The convergence properties can also be maintained because the computation will not involve skewed or distorted cells. Furthermore, the higher-order schemes are easy to implement in the Cartesian grid method. However, the boundary condition implementation is not obvious since the meshes are not body aligned and the cells near the body may intersect with surfaces of solid components. Hence, the success of Cartesian method depends greatly on the accuracy and efficiency of boundary treatment.

In the context of using Cartesian grids, various methods have been proposed to solve the boundary conditions either using structured or unstructured mesh approaches. Several approaches make use of cut cells for a finite volume construction and the rest involve grid points for a finite difference technique. A cut-cells method assures conservation since it uses the same flux computation procedure as for interior cells. However, the volume and fluxes calculation for irregularly shaped cut cells can be problematic. Issues related to numerical stability may arise due to very small or skewed cells near the boundary. To overcome this problem, cell-merging techniques must be used to combine cut cells that are too small with neighboring cells as proposed by Clarke et al.[7] and Ye et al.[28] Such cell-merging formation is not very straightforward in three-dimensional domain. In contrast to the finite volume approach, a finite difference scheme requires no computation of cut cell volumes and fluxes. However, this method is not conservative and may need an extrapolation procedure by making use of ghost points. Moreover, higher order representation for the boundary is required to improve solution accuracy and minimize the lack of conservation as seen in the work of Forrer and Jeltsch.[11]

An alternative method proposed in the literature to reduce the mesh generation effort is to use a gridless method instead of relying on well ordered cells. Batina pioneered a pure gridless method for compressible flows with the Euler equation[3,4] for the entire computational domain though the point distribution was made using an unstructured grid generator. A similar approach was reported by Liu and Su,[20] who also used this to solve for viscous flow problems. Most of the gridless methods usually employ a least-square fitting to solve the unknown flow variables. A gridless method approximates the flow equation by making use of a cloud of points. The cloud points can be chosen without any requirements of connectivity among the points. Additionally, this approach does not require the computation of cut-cells volume near the surface boundary. However the method does not ensure local conservation of the flow properties and instability can arise in cases where shock waves occur. Furthermore, such approaches tend to be less efficient since they require additional effort to construct the least-squares fitting over groups of grid points in solving the flow equations. Moreover, the implementation of solution acceleration techniques such as the multigrid method is not straightforward.

One way to combine the benefits of the conventional Cartesian grid approach and the gridless approach while reducing their disadvantages is to use a Cartesian method to solve the flow equations for the grid points in the interior domain where finite volume or finite difference stencils are complete and employ the gridless method to solve the boundary conditions near the surface. This technique was demonstrated by Kirshman and Liu,[18] who employed a finite difference scheme with Van Leer flux-splitting method. Koh et al.[19] applied this approach by employing least-square fitting to compute the flow for a layer of cells around the object and tested with several 2D problems. A recent work by Luo et al.[21] also adopted similar approach to analyze compressible flows.

The present paper combines the Cartesian multigrid method and the meshless method for Euler problems in three-dimensional domain. In contrast to earlier work by Kirshman and Liu,[18] and Koh et al.,[19] the current work does not use a least square approach nor solve the flow equations using the gridless method near or on the boundary. Instead, it approximates the solution inside the bodies via the use of reflected node using the radial basis function (RBF) to approximate the value from a selected cloud of points.

RBF is a recent tool in fluid mechanics for interpolating data which has been shown to be very successful in functional approximation. The RBF approach is also applicable for Navier-Stokes problems as seen in for example the work of Vanka and Ploplys.,[26] and Shu et al.,[23] who use the RBF method throughout the flow domain. However in our present study, the use of RBF is only applied to solve the surface boundary conditions. Like our previous effort reported in Koh et al.,[19] the method adopted solves the flow outside solid bodies using traditional finite volume scheme so that issues related to global conservation are greatly mitigated. Moreover with this framework of using structured grids, implementation of multigrid schemes is greatly facilitated.

The history of RBF approximations may be traced back to 1968, when multiquadric RBFs were first used by Hardy[14,15] to represent topographical surfaces given sets of sparse scattered measurements. The main strength of RBF lies in its ability to elegantly and accurately approximate multidimensional scattered data without any mesh. A highly useful property of RBF approximations is that the non-singularity of matrix is maintained without restrictions to the location of interpolation points. This is a great advantage over least-squares approximation where an ill-conditioned matrix may arise if appropriate points are not selected. Moreover, RBF does not require a minimum number of sampling points as least-square fitting does. Hence, the RBF approach offers greater flexibility for the boundary implementation especially in regions where point selections may be limited. Such regions are present in the context of complex geometries which can be a great challenge in points selections in implementing the boundary conditions.

In following sections, the approach that combines a three-dimensional Cartesian method for Euler equations is presented. The discretized equations are solved using the modified four-stage Runge-Kutta scheme developed by Jameson et al.[16] The basic principle of the multigrid implementation is also discussed. Furthermore, the proposed RBF method for boundary condition treatment and determination of cloud points are discussed. The method is then used to analyze the flowfield over NACA 0012 airfoil, ONERA M6 wing, and RAE Wing-Body. Solutions are compared with results using a traditional Euler solver with body-fitted curvilinear grids and/or experimental results where applicable.

II. Numerical Method

A. Governing Equations

The three-dimensional Euler equations representing the mass, momentum, and energy conservation that govern the motion of an unsteady inviscid flow can be expressed in integral form as

$$\frac{\partial}{\partial t}\int_V U \, \mathrm{d}V + \int_S F \cdot \vec{n} \, \mathrm{d}S = 0, \tag{1}$$

where V denotes the volume with closed boundary surface S and $\vec{n}$ is the outward normal vector on the surface. The conservative variable U and the flux vector F are given by

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix} ; \; F = a\,\mathrm{e}_x + b\,\mathrm{e}_y + c\,\mathrm{e}_z, \tag{2}$$

where

$$a = \begin{bmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uw \\ \rho uH \end{bmatrix} ; \; b = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + P \\ \rho vw \\ \rho vH \end{bmatrix} ; c = \begin{bmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + P \\ \rho wH \end{bmatrix} . \tag{3}$$

In the above equations P, ρ, E, and H denote the pressure, density, total energy, and total enthalpy, respectively, while u, v, and w are the Cartesian velocity components. Vectors a, b, and c are the x, y, and

z components of the flux vector F. For a perfect gas, total energy E, and total enthalpy$H$ are given by

$$E = \frac{P}{(\gamma - 1)\rho} + \frac{1}{2}(u^2 + v^2 + w^2);\; H = E + \frac{P}{\rho}, \tag{4}$$

where γ is the ratio of specific heats.

B. Spatial and Temporal Discretization

The governing equations above are solved using a 2nd order cell-centered finite volume scheme. In each cell (i, j, k) in the solution domain, the equation is approximated by

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(U_{i,j,k} V_{i,j,k}\right) + R_{i,j,k} = 0, \tag{5}$$

where $V_{i,j,k}$ is the cell volume, while the residual is defined as

$$R_{i,j,k} = Q_{i,j,k} + D_{i,j,k}. \tag{6}$$

$Q_{i,j,k}$ is the flux entering the cell through six faces, whereas $D_{i,j,k}$ is the artificial dissipative terms made up of a blend of second-order and fourth-order differences which was proposed by Jameson et al.[16] to provide first-order dissipation around shocks and third-order dissipation in smooth flow region.

Next, the multi-stage Runge-Kutta scheme is applied to march the solution to convergence in pseudo time. The integration is given as follows:

$$\begin{aligned}
U^{(0)}_{i,j,k} &= U^{m}_{i,j,k}\\
U^{(1)}_{i,j,k} &= U^{(0)}_{i,j,k} - \frac{1}{4}\frac{\Delta t}{V_{i,j,k}} R^{(0)}_{i,j,k}\\
U^{(2)}_{i,j,k} &= U^{(0)}_{i,j,k} - \frac{1}{3}\frac{\Delta t}{V_{i,j,k}} R^{(1)}_{i,j,k}\\
U^{(3)}_{i,j,k} &= U^{(0)}_{i,j,k} - \frac{1}{2}\frac{\Delta t}{V_{i,j,k}} R^{(2)}_{i,j,k}\\
U^{(4)}_{i,j,k} &= U^{(0)}_{i,j,k} - \frac{\Delta t}{V_{i,j,k}} R^{(3)}_{i,j,k}\\
U^{m+1}_{i,j,k} &= U^{(4)}_{i,j,k}
\end{aligned}$$

The scheme is explicit with a Courant-Friedrichs-Lewy (CFL) condition of CFL $\leq 2\sqrt{2}$ and has the second-order accuracy in time for a nonlinear equation. For steady state solutions, local time stepping can be used to accelerate convergence. Hence, the time step for each cell may vary and is based on applying the CFL criterion to the local flow condition.

C. Multigrid Strategy

Multigrid method is an effective technique to accelerate solution convergence. The main idea of the multigrid strategy is to use coarser grids in order to drive the solution on the finest grid faster to steady state. A more rapid convergence and a reduction of numerical effort can be achieved since the computation for finding a new solution is distributed mainly over the coarser grids and larger time steps can be used on these grids.

The multigrid technique used here is based on the traditional multigrid approach. In this scheme the finest computational grid spans the entire domain and successively coarser grids are formed by omitting every other grid point of the finer grid. Additionally, all the grids in each level have to be fully embedded in the previous level of coarser grids. The finer grids are responsible for near body solution while the coarser grids are responsible for calculating solution on the remaining parts of the flowfield.

The efficiency of the Cartesian flow solver can be greatly enhanced with a multigrid implementation. A good multigrid strategy helps accelerate solution convergence and reduce computational time significantly. By exploiting the use of successively coarser sets of grids where each grid level is responsible for a particular bandwidth of errors, the solution is accelerated by time stepping via a combinational sequence of fine and coarse grids. The grids in one level have to be fully embedded in the previous coarser level to ensure proper nesting of grids. This is important for the multigrid process where transfer of information takes place from one grid to another. This approach provides a mean of strongly-coupled communication in the hierarchy of Cartesian grids.

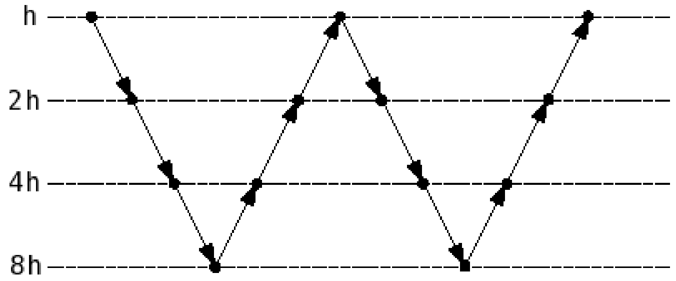


Figure 1. V-Cycle multigrid.

For this work, the multigrid procedure is based on the V-cycle multigrid, where the computation starts from the finest grid as shown in Figure 1. The solution is then interpolated to the next coarser grid, several cycles are executed on intermediate levels, and finally the coarsest grid is reached. The transfer of information between coarse and fine grids implemented here is based on the Full Approximation Storage (FAS) method which can be applied directly to nonlinear equations.[6] The procedure is outlined as follows:

1. The solution on the fine grid is transferred to the coarse grid by means of interpolation

$$U_{2h}^{(0)} = \hat{I}_h^{2h} \, U_h^{n+1}, \tag{7}$$

   where $\hat{I}_h^{2h}$ is the interpolation operator, which transfers the value from the center of fine grid cells to the center of coarse grid cells via trilinear interpolation.

2. Additionally, the residuals have to be transferred from the fine to the coarse grid in order to smooth the low-frequency order components and retain the solution accuracy of the fine grid on the coarse grid. To do this, the so-called forcing function is calculated as the difference between the residual transferred from the fine grid and the residual computed using the initial solution $U_{2h}^{(0)}$ on the coarse grid, i.e.,

$$(Q_F)_{2h} = I_h^{2h} \, R_h^{n+1} - R_{2h}^{(0)}. \tag{8}$$

   Here, I_h^{2h} denotes the restriction operator which is defined as a sum of the residuals from all cells which are contained in one-coarse grid control volume.

3. Then, the solution is reevaluated in the coarse grid again by using the modified residual which has been added by the forcing function, i.e.,

$$(R_F)_{2h} = R_{2h} + (Q_F)_{2h} \, . \tag{9}$$

4. After the coarse grid solution is obtained, we compute the coarse grid correction

$$\delta U_{2h} = U_{2h}^{n+1} - U_{2h}^{(0)}. \tag{10}$$

   The coarse grid correction is interpolated to the fine grid in order to improve the solution accuracy there. Thus, the new solution on the fine grid is

$$U_h' = U_h^{n+1} + I_{2h}^h \delta \, U_{2h}, \tag{11}$$

   where I_{2h}^h denotes the prolongation operator, which is again implemented by trilinear interpolation.

5. The previous steps are repeated through the whole multigrid cycles until the convergence is obtained.

III. Boundary Condition Implementation

To ensure the success of the Cartesian grid method, the boundary condition should be implemented accurately. For this reason the radial basis function (RBF) is used for the interpolation, since it has several advantages compared to the least-square fitting. One of the main benefits of RBF is that it does not require a minimum number of supporting points because it is a function of distance between points, not polynomial terms. Additionally, RBF can give excellent approximations even when the number of interpolation points is small. Moreover, RBF is applicable not only for interpolation problems, but also for extrapolations. The resulting matrix is also non-singular regardless of the irregular position of cloud points. Thus, the RBF can perform better in cases where point selection is limited.

A. Nodes Classification

Before we discuss about how to apply the boundary conditions on a surface, we need to classify the nodes near a surface boundary into three types as depicted in Fig. 2. The intersection points (type 1) are points on which the boundary condition will be imposed. Cell centers which are located inside an object are denoted as solid-cell centers (type 2) and centers which are located outside a surface and have solid-cell center as neighbor are called cut-cell centers (type 3). In this code, the flow variables on a cut-cell center are obtained from the finite volume computation, while the quantities on a solid-cell center are calculated via the use of reflected node, which is the reflection of solid-cell center with the tangent plane on intersection as a mirror plane. Alternatively, the true reflected point can be found by calculating the nearest distance between the solid-cell center and the surface at first. Nevertheless, this approach shall increase the computational cost, and therefore, the first method to find the reflected node is more preferable. In addition, the errors due to using an approximated reflected node can be reduced by refining the Cartesian mesh. Finally, having computed the flow variables on solid cell center, the values on intersection can now be computed by employing a linear interpolation between values on the cut-cell center and the solid-cell center.

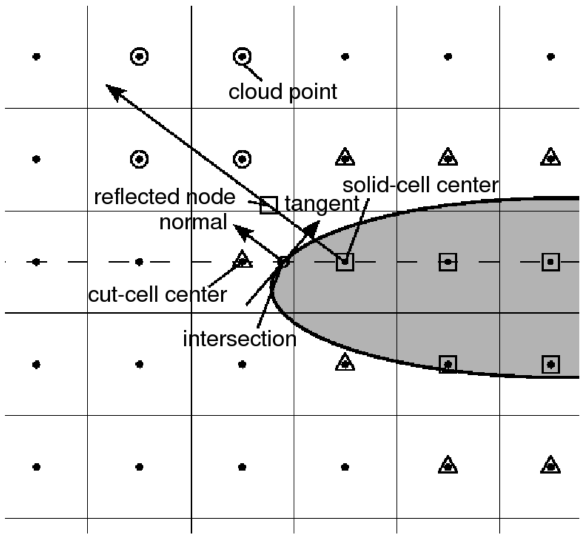


Figure 2. Classification of nodes.

The procedure for nodes determination is given as follows:

1. Intersection points are obtained by performing a ray tracing procedure which starts from the face centre of the most-outer cells of all Cartesian grids following the three main directions. The program then determines whether an intersection is an entering or exiting node by computing the scalar product between the normal vector on the intersection and the ray direction

2. After finding the intersections, the cut-cell centres and solid-cell centres can now be determined by calculating the scalar product between the normal vector on the intersection and the vector which connects the intersection and the cell-centre

3. The reflected node are obtained by mirroring the solid-cell centre with the tangent plane on the intersection as the mirror plane

4. A cloud of points is chosen to be the sampling points for interpolating values on the reflected node. Points which are located on the same side as the outward normal direction on the intersection and have the nearest distance to the reflected node are considered as the cloud points. The RBF method shall be applied to these points.

This method of boundary implementation is direct and simple. In addition, the missing solid nodes' value needed for Euler fluxes computation can be provided at the same time. The information pertaining to the surface nodes are associated with the cut-cell and solid-cell centers and are provided in the preprocessing stage prior to flow computation. A solid cell can have more than one associated surface nodes depending on the number of intersections it has with the object surface. Multiple intersections associated with a single solid cell are more likely to be found for profiles with thin surface. However, the current method can handle this special case since the surface nodes together with the corresponding cut cell and solid cell are managed using an unstructured data approach. As such, the boundary implementation is performed in a separate routine which makes it flexible and can be easily implemented into other schemes.

B. Interpolation Using Radial Basis Function

To approximate the variables on a reflected node, the RBF is applied to a cloud of points which surround the reflected node. For scattered data interpolation problem, the approximation of a function $f(x)$ takes the form

$$f(x) \approx s(x) = \sum_{j=1}^{N} \lambda_j \, \phi\left(\|x - x_j\|\right), \tag{12}$$

where N is the number of sampling points, λ is the coefficient vector to be determined, and ϕ is the radial basis function. Several commonly used radial basis functions are

$$\text{multiquadric} \quad : \phi(r) = \sqrt{r^2 + c^2},\ c > 0 \tag{13}$$

$$\text{inverse multiquadric} \quad : \phi(r) = \frac{1}{\sqrt{r^2 + c^2}},\ c > 0 \tag{14}$$

$$\text{Gaussian} \quad : \phi(r) = e^{-cr^2},\ c > 0, \tag{15}$$

where $r = \|x - x_j\|_2$ and c^2 is the shape parameter which must be chosen carefully as described later. The coefficient λ is obtained by solving the system of equations

$$\Phi \, \lambda = f, \tag{16}$$

where Φ is the $N \times N$ matrix with elements

$$\Phi_{ij} = \phi\left(\|x_i - x_j\|\right),\ 1 \le i, j \le N, \tag{17}$$

while f is a vector consisting of N functional values.

Many investigations have shown that the multiquadric function (MQ) provides most accurate results and therefore, it is implemented in our calculation.[10, 13, 17] However, the accuracy of this approach depends heavily on the value chosen for c^2. The optimal value of c is problem-dependent, and most-formulas for choosing c^2 involve the number of sampling points and their locations. One of those formulas was proposed by Franke[12] which assumes that $c^2 = (1.25D/\sqrt{N})^2$, where D is the diameter of a sphere which surrounds the xyz data point set. For present computation, it is replaced by

$$c^2 = (1.25d)^2, \tag{18}$$

where d is the mean distance from each data point (x_i, y_i, z_i) to its nearest neighbor.

In this work, the MQ RBF is implemented in a slightly different manner as the common one. According to Shu et al.,[23] an extra term may be added to Eq. (12). This term shall improve the matrix condition number and stabilize the computation. Additionally, the solution obtained shall be more accurate due to this extra controlling parameter which mainly depends on the functional values, not the point coordinates. In three-dimensional problems the MQ RBF can then be written as

$$f(x,y,z) = \sum_{j=1}^{N} \lambda_j \sqrt{(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2 + c^2} + \lambda_{N+1}. \tag{19}$$

To make the problem be well-posed, another equation is needed

$$\sum_{j=1}^{N} \lambda_j = 0 \Rightarrow \lambda_i = - \sum_{j=1,\, j\neq i}^{N} \lambda_j. \tag{20}$$

Substituting Eq. (20) into Eq. (19) yields

$$f(x,y,z) = \sum_{j=1,\, j\neq i}^{N} \lambda_j \, g_j(x,y,z) + \lambda_{N+1}, \tag{21}$$

where

$$g_j(x,y,z) = \sqrt{(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2 + c^2} - \sqrt{(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 + c^2}. \tag{22}$$

To avoid confusion, λ_{N+1} can be replaced by λ_i, and Eq. (21) can be expressed as

$$f(x,y,z) = \sum_{j=1,\, j\neq i}^{N} \lambda_j \, g_j(x,y,z) + \lambda_i. \tag{23}$$

Then, the system of equations for N selected points can be written as $\Phi \, \lambda = f$, where

$$\Phi = \begin{bmatrix} g_1(x_1,y_1,z_1) & \cdots & 1 & \cdots & g_N(x_1,y_1,z_1) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ g_1(x_i,y_i,z_i) & \cdots & 1 & \cdots & g_N(x_i,y_i,z_i) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ g_1(x_N,y_N,z_N) & \cdots & 1 & \cdots & g_N(x_N,y_N,z_N) \end{bmatrix} ; \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \vdots \\ \lambda_N \end{bmatrix} ; f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_N \end{bmatrix}. \tag{24}$$

After solving for the coefficients λ, the variables on the reflected node can be easily found using Eq. (19).

The RBF approximation explained above is then employed for calculating several CFD benchmarks and the results shall be compared with those obtained by the same Cartesian method but combined with the least-square fitting. Therefore, a brief illustration about the least-square approach is presented subsequently. For a group of N cloud points in 3D domain, the least-square approximation of a function $f(x,y,z)$ can be written as:

$$f(x,y,z) = a_0 + a_1 x + a_2 y + a_3 z + a_4 xy + a_5 yz + a_6 xz + \ldots + a_m x^i y^j z^k, \tag{25}$$

which consists of $m+1$ polynomial terms where $a_0, a_1, \ldots, a_m$ are the least-square coefficients and i, j, and k are the highest order of x, y, and z, respectively. To obtain accurate and smooth solutions while minimizing the computational cost, the least-square approximation can be truncated into a 2nd order polynomial

$$f(x,y,z) = a_0 + a_1 x + a_2 y + a_3 z + a_4 xy + a_5 yz + a_6 xz + a_7 x^2 + a_8 y^2 + a_9 z^2, \tag{26}$$

which includes 10 polynomial terms. By applying this equation to each cloud point, we obtain a system of equations for N selected points which can be expressed as $\Phi \, \lambda = f$, where

$$\Phi = \begin{bmatrix} 1 & x_1 & \cdots & y_1^2 & z_1^2 \\ 1 & x_2 & \cdots & y_2^2 & z_2^2 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_N & \cdots & y_N^2 & z_N^2 \end{bmatrix} ; \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{10} \end{bmatrix} ; f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{bmatrix}. \tag{27}$$

Since λ has 10 unknowns, at least 10 cloud points are required for solving the 2nd order least-square approximation in 3D problems properly. This is one of the disadvantages of using least-square approach instead of RBF method.

C. Surface Boundary Condition

The boundary condition is implemented on the surface by applying a linear interpolation between the solution on cut-cell centers and the solution on solid-cell centers. Hence, the variables on solid-cell centers need to be recovered at first by making use of the values on reflected nodes.

For inviscid flow cases, the variables on solid-cell centers are set as follows:

$$\begin{aligned}(V_n)_S &= -(V_n)_R \\ (V_{t1})_S &= (V_{t1})_R \\ (V_{t2})_S &= (V_{t2})_R \\ (\rho)_S &= (\rho)_R \\ (P)_S &= (P)_R + d\frac{\partial P}{\partial \vec{n}}\,,\end{aligned} \tag{28}$$

where $\vec{n}$ is the normal direction, while t_1 and t_2 are the two orthogonal directions on the tangential plane. V, ρ, and P denote the velocity, density, and pressure, respectively, while d is the distance between the reflected node (R) and the solid-cell center (S). The normal pressure gradient used for pressure interpolation can be computed as

$$\frac{\partial P}{\partial \vec{n}} = -\frac{\rho V_t^2}{R} \tag{29}$$

where R is the local surface radius of curvature in the flow direction. By taking into consideration the body curvature we can locate shock waves on the surface more accurately and reduce spurious entropy productions.[9]

D. Local Curvature

In 2D computation, the curvature is fixed as the tangential direction is fixed at any point on the surface. Hence, the curvature can be determined before starting the computation since it is independent of the flow direction. However in 3D computation, it is computationally more involved to find the surface curvature in the flow direction since it changes in the coarse of the computation. To minimize the computational effort for calculating curvature repeatedly, the normal pressure gradient can be modified into

$$\frac{\partial P}{\partial \vec{n}} = -\rho\left(\frac{{V_{t1}}^2}{R_1} + \frac{{V_{t2}}^2}{R_2}\right), \tag{30}$$

where R_1 and R_2 are the radius of curvature in the first and second tangential vectors, respectively. Although the above approach is not mathematically identical to Eq. (29), the computational cost can be greatly reduced since the radius of curvature is computed only once in the preprocessing part.

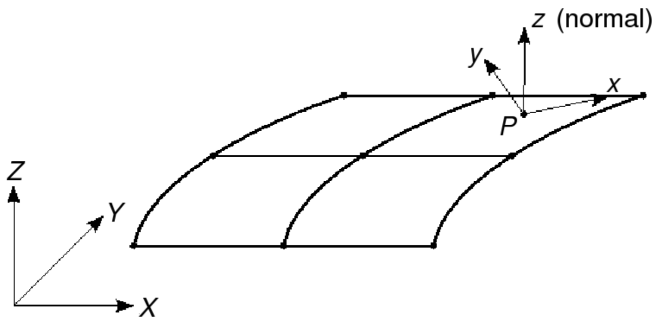


Figure 3. Nine-node quadrilateral element.

To illustrate the method used to calculate surface curvature in any direction, let us consider a 9-node quadrilateral element as shown in Figure 3. For this element, or in general for any arbitrary $m \times n$ element, we can find the normal vector on a particular point (e.g. point P). Then, we construct a new orthogonal xyz coordinate system with point P as the origin where the xy-plane is tangent to the surface and the z-axis is the normal vector obtained before. In this new system the x-axis is set to any preferred direction. Thus, the y-axis can be easily determined since x- and z-axis are known. Having found all the axis directions, the 9 nodal coordinates are now transformed from the global coordinate system XYZ to the the new one xyz. Generally, the equation of this surface can be expanded about this point into a polynomial giving the height z as a function of x and y:

$$z = f(x,y) = a + bx + cy + exy + fx^2 + gy^2 + hx^2y + ixy^2 + jx^3 + ky^3 + \dots \tag{31}$$

Assuming that the surface mesh is quite fine and for a cheaper computation, the function can be truncated into

$$z = f(x,y) = a + bx + cy + exy + fx^2 + gy^2 + hx^2y + ixy^2. \tag{32}$$

Then, coefficients $a, b, c, e, \dots, i$ can be solved in a least-square sense using the 9 nodal coordinates. The last step is to calculate the radius of curvature in the x and y direction

$$R_x = \frac{\left[1 + (\partial f/\partial x)^2\right]^{3/2}}{\partial^2 f/\partial x^2};\ R_y = \frac{\left[1 + (\partial f/\partial y)^2\right]^{3/2}}{\partial^2 f/\partial y^2}, \tag{33}$$

which can be easily done once the coefficients are known. In this work, the least-square approach is more preferred than the RBF for computing curvature especially because a least-square function can be differentiated easily and thus, the radius of curvature can be determined straightforward.

IV. Results and Discussion

Three examples are presented in this chapter to check the accuracy and efficiency of the three-dimensional Cartesian grid Euler solver using the radial basis function for boundary implementation. The first case is the flow over a symmetric NACA 0012 airfoil where the results from both 3D and 2D computation are compared. Two other examples, the ONERA M6 wing and the RAE wing-body, are also presented to show the robustness of the method. For comparison, the solution obtained from least-square fitting and from experiments are also provided.

A. NACA 0012 Airfoil

NACA 0012 airfoil is analyzed in three flow cases with different combinations of Mach number M and angle of attack α. Since in the 3D Cartesian code the span is uniform along the y-direction and the flow is also constant in the span-wise direction, the results can be compared with the solution obtained by using FLO52 code, which is an established 2D body-fitted grid code. The Cartesian multigrid system consists of four levels with a total of approximately 1.05 million grid points. The two finest Cartesian meshes and the body-fitted grid used to compute the flow are shown in Fig. 4(a). Symmetric boundary condition is set for planes XZ, whereas far field condition is applied to other planes. The Cartesian grid solution is obtained using the multiquadric RBF as described in Section 3. For each case, the distributions of the pressure coefficient C_p are plotted with the negative pressure upward, and the convergence histories shows the mean rate of change of the density.

For case 1 with $M = 0.85$ and $\alpha = 0.0°$, the C_p plot is depicted in Fig. 5(a). The results are obtained using 8 cloud points for RBF. For comparison, the case is also tested using 2$^{\text{nd}}$ order least-square fitting with 10 points since this is the minimum number of points required for 2$^{\text{nd}}$ order polynomial approximation in 3D domain. Although the RBF uses less points than the least-square, good agreement can be observed. However, both approaches show a rather shifted shock wave location. To check the effect of surface curvature for the pressure boundary condition implementation, we present the Cartesian solutions with and without the pressure correction in Fig. 5(b). It is clear that the C_p plot of both solutions are similar in general except for the location of the shock. With the surface curvature taken into consideration, the shock region can be captured correctly and the solution obtained is in excellent agreement with the reference solution.

In case 2, the subsonic flow of $M = 0.5$ and $\alpha = 3.0°$ is computed, and the Mach contour plot is presented in Fig.6(a). From the C_p plot in Fig. 6(b), we can see that the solution obtained is almost identical to the FLO52 solution. However, the body-fitted grid code is able to produce smoother results than the current approach due to the finer clustering near the nose. To achieve the same level of accuracy as the FLO52 solution, a slightly finer mesh is required for the Cartesian method. The solution converges fast as shown in Fig. 6(c), where the maximum residual is reduced by almost 5 orders of magnitude in about 300 iterations on the finest grid. Overall, the present method provides similarly good results for this subsonic case as the body-fitted grid method.

Figure 7(a) displays the Mach contour for case 3 with $M = 0.8$ and $\alpha = 1.25°$ where smooth distributions in the flowfield are observed. The C_p plot obtained employing RBF, least square, and FLO52 is depicted in Fig. 7(b). The solutions again show good agreement with the FLO52 results. However, a finer clustering is needed to capture shock waves more precisely. The convergence rate is somewhat slower than the previous case due to the shocks near leading edge and on lower surface.

B. ONERA M6 Wing

The ONERA M6 Wing is one of the classic tests used in numerous papers to validate CFD computation. The flow over the wing is computed with standard test condition of Mach number $M = 0.84$, angle of attack $\alpha = 3.06°$, and Reynolds number $Re = 1.17 \times 10^7$. Extensive pressure measurement over the wing was reported by Schmitt and Charpin.[22] The Cartesian solution is obtained using 7 levels of grids with 19 grid blocks. Figure 8 shows the arrangement of grid blocks used to compute the flow. The unit cell size of the finest grid is set approximately half of the wing mesh size in $x-$direction. Approximately 2.3 million grid points are used in this case. Since we adopt the multigrid technique, it is possible to place fine grids at regions which require flow refinement. Thus, in this example several fine grids are placed near the leading edge in order to resolve the curvature and shocks which might happen in this region.

The steady state solution for the transonic flow is obtained and the C_p contour in the flowfield and on the surface is presented in Fig. 9. To check the solution quality, we calculates the pressure profiles at four spanwise locations (20%, 44%, 65%, and 90% span) using multiquadric RBF with 10 cloud points. It is clear that the results shown in Fig. 10 compare very favorably with the experimental data obtained by Schmitt and Charpin.[22] The 20%, 44%, and 65% stations properly show two shocks on the upper surface while the 90% station shows a correct location of the single shock. For comparison, we also calculates the pressure distribution using 2$^{\text{nd}}$ least square fitting with the same number of supporting points (not shown here). The least square approach is also able to produce correct C_p profiles on the wing, however, the errors in predicting the leading edge suction peak almost double the errors obtained by RBF. Thus, RBF again shows its benefits over least square. The convergence history for this problem is depicted in Fig. 11 in terms of density residual. We can see that the multigrid technique considered is very effective in accelerating convergence. In about 260 iterations the maximum residual is decreased by more than 5 orders of magnitude on the finest grid. Hence, the current method has demonstrated its capability in analyzing transonic flow problems.

C. RAE Wing-Body

One of the main advantages of using Cartesian grid is in handling complex configurations in three-dimensional domain. In the current Cartesian solver, a complex geometry is broken down into smaller components and treated individually on a component by component basis in the preprocessor stage. This makes it easier for users in modelling the configuration and for the preprocessor to find the intersection points of the Cartesian grid blocks with each of the surface grids and store the geometrical properties needed for the flow computation. To demonstrate the capability of the Cartesian code in dealing with complex geometry, the three-dimensional flow over a wing-fuselage configuration (RAE) is computed. The case is performed at $M = 0.9$, $\alpha = 1.0°$, and Reynolds number based on semi span 3×10^6. Extensive pressure measurements over the wing and fuselage were reported by Treasogold et al.[25] The Cartesian multigrid system consists of 8 levels with a total of 19 grid blocks as depicted in Fig. 12. As in the previous example, grids near the leading edge are also refined. The unit cell size of the finest grid is set nearly half of the wing mesh size in $x-$direction. Very fine grids totalling over 4 million grid points are used to obtain the present results.

The transonic flow solution is calculated and the C_p contour plot is displayed in Fig. 13. Smooth distributions across the flowfield and the structure are observed. Figure 14 shows the C_p variations on the wing at 25%, 60%, and 92% span locations obtained using multiquadric RBF with 10 points. It is seen that the shock wave on the upper surface becomes stronger towards the wing tip. The general agreement of the current computation with the experimental results obtained by Treasogold et al.[25] is good. This test is also run using least square fitting with the same number of cloud points (results not shown here). Results obtained employing least square technique compare well with those from experiments. Nevertheless, the shock wave location near the wing tip is more delayed when compared to RBF prediction. Figure 15 shows the convergence history for this problem in terms of density residual. Approximately 300 iterations are needed to solve the Euler equations on the intermediate and finest levels with a reduction in the maximum residual of almost 6 orders of magnitude. From this example, the presented Cartesian method is found to be very favorable in computing the flow over complex geometry due to the easiness of grid generation and the accuracy as well.

V. Conclusion

A three-dimensional Cartesian grid approach combined with gridless method for the solution of the Euler equations is presented. The approach uses RBF to implement boundary condition, while a standard structured grid method is used everywhere else. The multiquadric RBF is implemented in this work due to its excellent approximations when compared to other functions as demonstrated in numerous papers. The RBF is adopted for this work due to its ability to produce accurate results without requiring a minimum number of supporting points as the commonly-used least-square fitting. Moreover, in contrast to the least squared method, RBF offers greater flexibility in regions where point selection may be very limited since the resulting matrix will be non-singular regardless of the sampling point's location. This greatly facilitates problems which involve complex geometries where it is not straightforward to ensure that the points selected are spread out and sufficiency apart from one another.

The method is applied to analyze a range of subsonic and/or transonic flows. For a two dimensional NACA 0012 airfoil test, the solutions obtained using RBF are in excellent overall agreement with those obtained by traditional body-fitted grid. It is also shown here that as compared to the least-square method, RBF provides similar accuracy with less cloud of points while for similar number of points better accuracy is achieved. This is further demonstrated in the test of the ONERA M6 finite wing test case where the results compare favorably with those from experiments where the suction peak near leading edge is better predicted using RBF.

To demonstrate the current solver capability in handling complex configurations, the flow over RAE wing-fuselage structure was also computed. The benefit of using Cartesian grids becomes more significant for this problem due to its ease of grid generation around complex structures. The results are again in good agreement with the experimental ones. By employing the multigrid technique, converged solutions for all cases can also be obtained only in a few hundred iterations. The method has advantages over purely gridless algorithms where it is not straightforward to incorporated multigrid methods, and issues with global conservation are present. Moreover, the advantages of a Cartesian grid method go beyond computational efficiency only; it reduces the man-hours needed for grid generation.

In general, the presented Cartesian solver offers great potential for different kinds of problems, especially those involving complex or isolated geometries. Future extension may include skewed/rotated Cartesian grids. It is pointed out here that the proposed approach of gridless boundary condition treatment with RBF method within the Cartesian framework is equally applicable to the solution of the Navier-Stokes equations. However its effectiveness is confined to the Navier-Stokes equations for low Reynolds number flows, if excessive grids are to be avoided.

References

[1]Alonso, J. J., and Jameson, A., "Fully-Implicit Time-Marching Aeroelastic Solutions," AIAA Paper 94-0056, Jan. 1994.

[2]Batina, J. T., "Unsteady Euler Algorithm with Unstructured Dynamic Mesh for Complex Aircraft Aerodynamic Analysis," *AIAA Journal*, Vol. 29, No. 3, 1991, pp. 327-333.

[3]Batina, J. T., "A Gridless Euler/Navier-Stokes Solution Algorithm for Complex-Aircraft Applications," AIAA Paper 93-0333, Jan. 1993.

[4]Batina, J. T., "A Gridless Euler/Navier-Stokes Solution Algorithm for Complex Two-Dimensional Applications," NASA TM-107631, June 1992.

[5]Berger, M. J., and LeVeque, R. J., "An Adaptive Cartesian Mesh Algorithm for the Euler Equations in Arbitrary Geometries," AIAA Paper 89-1930, AIAA Computational Fluid Dynamics Conference, 9th, Buffalo, NY, June 13-15, 1989.

[6]Blazek, J., *Computational Fluid Dynamics: Principles and Applications*, Elsevier, 2001.

[7]Clarke, D. K., Salas, M. D., and Hassan, H. A., "Euler Calculations for Multielement Airfoils Using Cartesian Grids," *AIAA Journal*, Vol. 24, No. 3, 1986, pp. 353-358.

[8]Coirier, W. J., and Powell, K. G., "An Accuracy Assessment of Cartesian Mesh Approaches for the Euler Equations," *Journal of Computational Physics*, Vol. 117, Issue 1, March 1995, pp. 121-131.

[9]Dadone, A., and Grossman, B., ”Surface Boundary Conditions for the Numerical Solution of the Euler Equations,” *AIAA Journal*, Vol. 32, No. 2, 1994, pp. 285-293.

[10]Foley, T. A., ”Near Optimal Parameter Selection for Multiquadric Interpolation,” *Journal of Applied Science and Computation*, Vol. 1, No. 1, June 1994, pp. 54-69.

[11]Forrer, H., and Jeltsch, R., ”A Higher-Order Boundary Treatment for Cartesian-Grid Methods,” *Journal of Computational Physics*, Vol. 140, No. 2, 1998, pp. 259-277.

[12]Franke, R., ”A Critical Comparison of Some Methods for Interpolation of Scattered Data,” *Technical Report NPS-53-79-003*, Naval Postgraduate School, 1979.

[13]Franke, R., ”Scattered Data Interpolation: Tests of Some Methods,” *Mathematics of Computation*, Vol. 38, No. 157, Jan. 1982, pp. 181-200.

[14]Hardy, R. L., ”Theory and Applications of the Multiquadric-Biharmonic Method. 20 Years of Discovery 1968-1988,” *Computers and Mathematics with Applications*, Vol. 19, No. 8-9, 1990, pp. 163-208.

[15]Hardy, R. L., ”Multiquadric Equations of Topography and Other Irregular Surfaces,” *Journal of Geophysical Research*, Vol. 76, 1971, pp. 1905-1915.

[16]Jameson, A., Schmidt, W., and Turkel, E., ”Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes,” AIAA Paper 81-1259, Fluid and Plasma Dynamics Conference, 14th, Palo Alto, CA, June 23-25, 1981.

[17]Kansa, E. J., ”Multiquadrics - A Scattered Data Approximation Scheme with Applications to Computational Fluid-Dynamics I,” *Computers and Mathematics with Applications*, Vol. 19, No. 8/9, 1990, pp. 127-145.

[18]Kirshman, D. J., and Liu, F., ”A Gridless Boundary Condition Method for the Solution fo the Euler Equations on Embedded Cartesian Meshes with Multigrid,” *Journal of Computational Physics*, Vol. 201, No. 1, 2004, pp. 119-147.

[19]Koh, E. P. C., Tsai, H. M., and Liu, F., ”Euler Solution Using Cartesian Grid with a Gridless Least-Squares Boundary Treatment,” *AIAA Journal*, Vol. 43, No. 2, 2005, pp. 246-255.

[20]Liu, J. L., and Su, S. J., ”A Potentially Gridless Solution Method for the Euler/Navier Stokes Equations,” AIAA Paper 96-0526, Jan. 1996.

[21]Luo, H., Baum, J. D., and Lohner, R., ”A Hybrid Cartesian Grid and Gridless Method for Compressible Flows,” *Journal of Computational Physics*, Vol. 214, No. 2, 2006, pp. 618-632.

[22]Schmitt, V., and Charpin, F., ”Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers,” *Experimental Data Base for Computer Program Assessment*, AGARD Advisory Report AR-138, 1979.

[23]Shu, C., Ding, H., and Yeo, K. S., ”Local Radial Basis Function-Based Differential Quadrature Method and Its Application to Solve Two-Dimensional Incompressible Navier-Stokes Equations,” *Computational Methods in Applied Mechanics and Engineering*, Vol. 192, 2003, pp. 941-954.

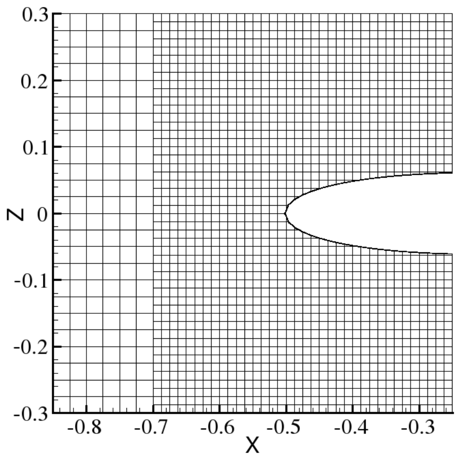
[24]Thompson, J. F., Thames, F. C., and Mastin, C., ”Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies,” *Journal of Computational Physics*, Vol. 15, pp. 299-319, 1974.

[25]Treasogold, D. A., Jones, A. F., and Wilson, K. H., ”Pressure Distribution Measured in the RAE 8ft x 6ft Transonic Wing Tunnel on RAE Wing A in Combination with an Axi-Symmetric Body at Mach Numbers of 0.4, 0.8, and 0.9,” AGARD-AR-138, May 1989, Chap. B-4.

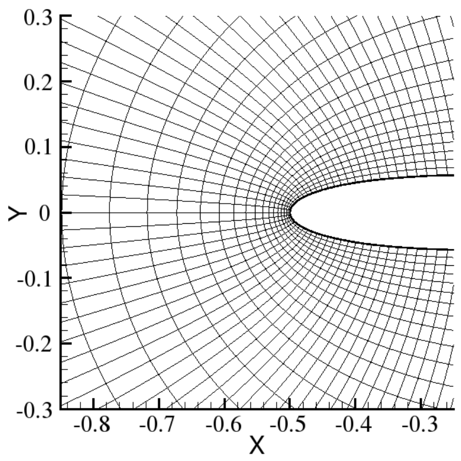
[26]Vanka, S. P., and Ploplys, N., ”Meshless Methods for Navier-Stokes Equations Using Radial Basis Function,” *Advances in Computational Engineering and Sciences*, Vol. 2, ICES, 2000.

[27]Weatherill, N. P., Hassan, O., Marchant, M. J., and Marcum, D. L., ”Adaptive Inviscid Flow Solutions for Aerospace Geometries Using Efficiently Generated Unstructured Tetrahedral Meshes,” AIAA Paper 93-3390, 1993.

[28]Ye, T., Mittal, R., Udaykumar, H. S., and Shyy, W., ”An Accurate Cartesian Grid Method for Viscous Incompressible Flows with Complex Immersed Boundaries,” *Journal of Computational Physics*, Vol. 156, No. 2, 1999, pp. 209-240.

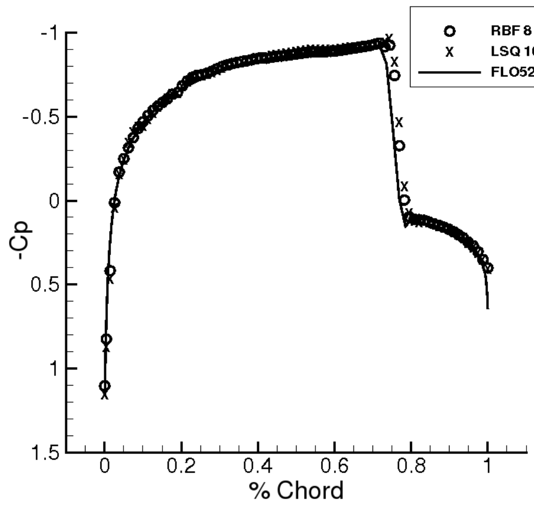


(a)

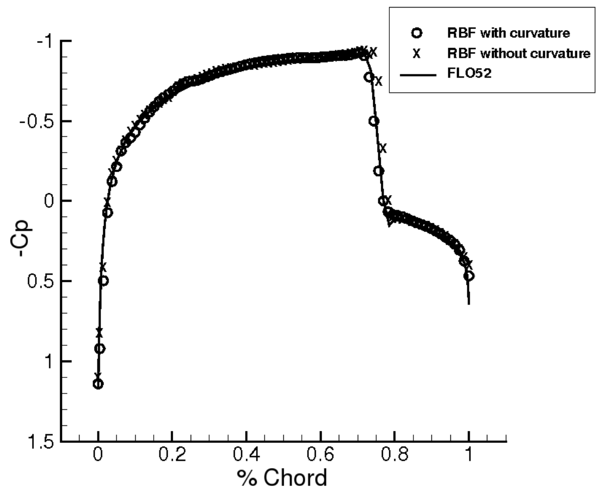


(b)

Figure 4. Grid blocks for NACA 0012 wing: (a) Cartesian multigrid; (b) Body-fitted grid.

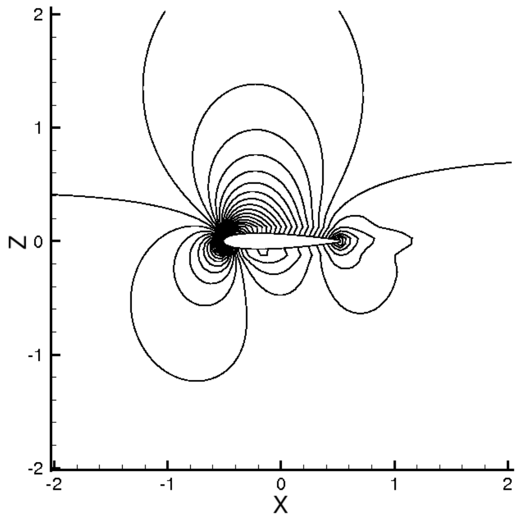


(a)

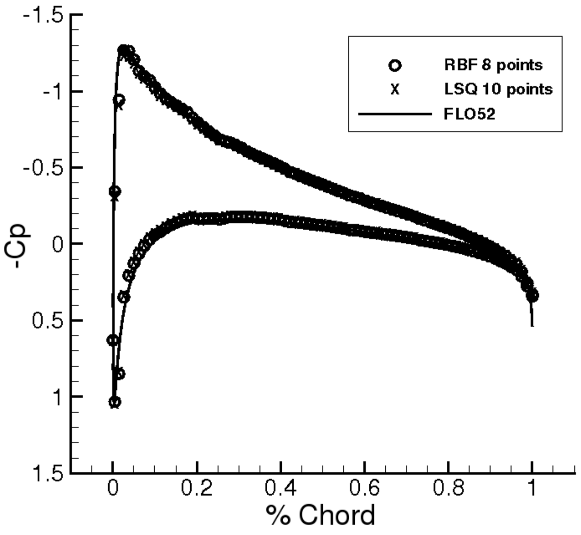


(b)

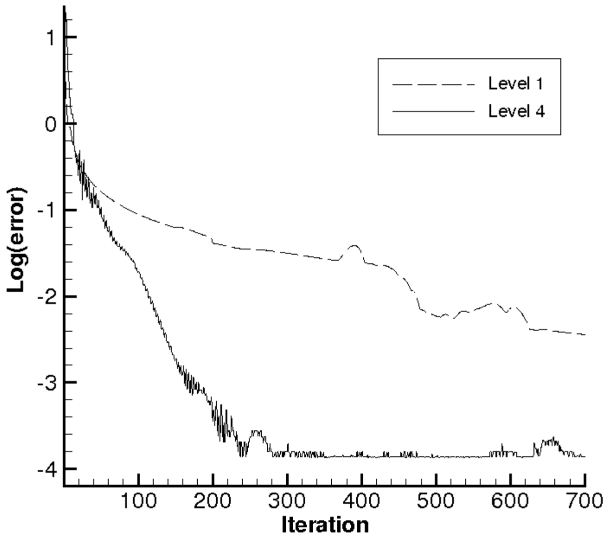
Figure 5. C_p plot for NACA 0012 wing with $M = 0.85$ and $\alpha = 0.0°$: (a) Comparison of RBF and least-square; (b) Effect of pressure correction.



(a)

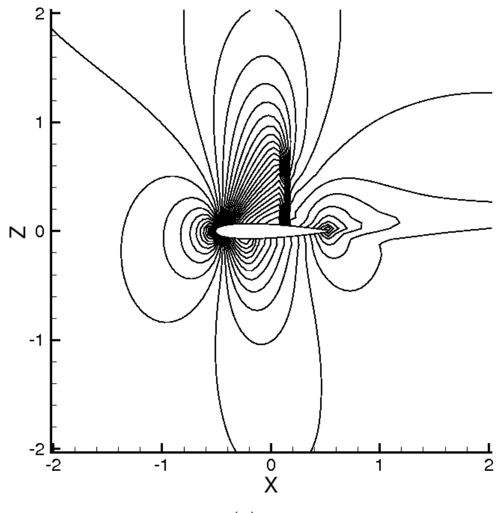


(b)



(c)

Figure 6. Solution for NACA 0012 wing with $M = 0.5$ and $\alpha = 3.0°$: (a) Mach contour plot with contour $M_{\min} = 0.0$, $M_{\max} = 0.73$, and interval = 0.01; (b) C_p plot; and (c) Convergence plot



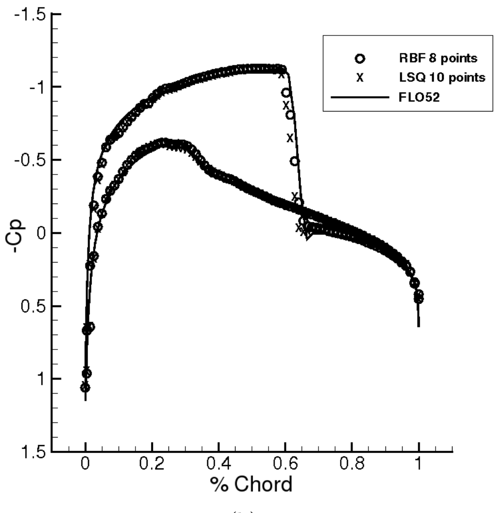


Figure 7. Solution for NACA 0012 wing with $M = 0.8$ and $\alpha = 1.25°$: (a) Mach contour plot with contour $M_{\min} = 0.0$, $M_{\max} = 1.375$, and interval = 0.025; (b) C_p plot.

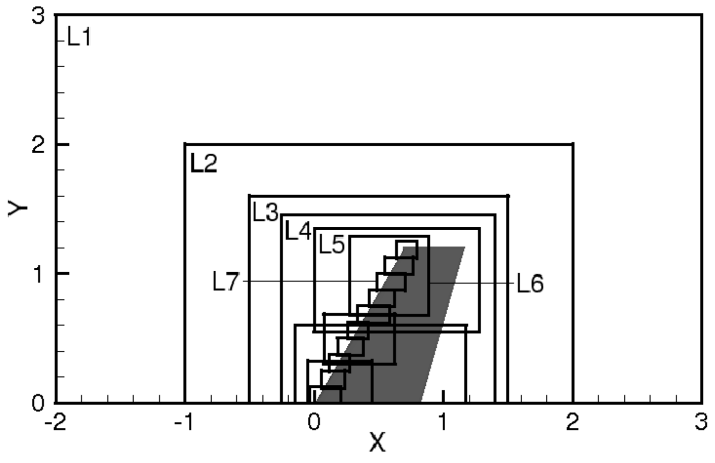


Figure 8. Grid blocks for ONERA M6 wing.

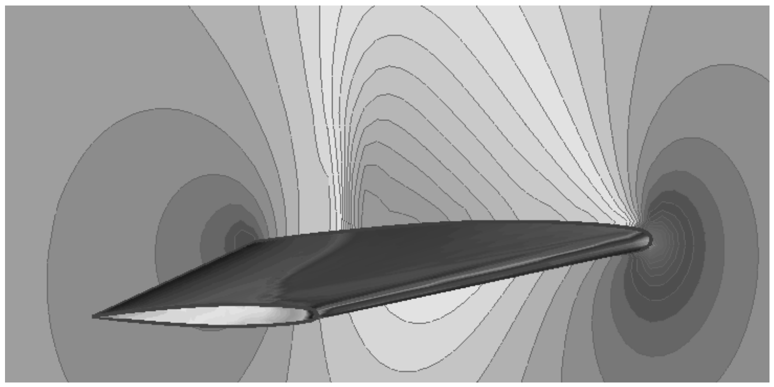


Figure 9. C_p contour for ONERA M6 wing.

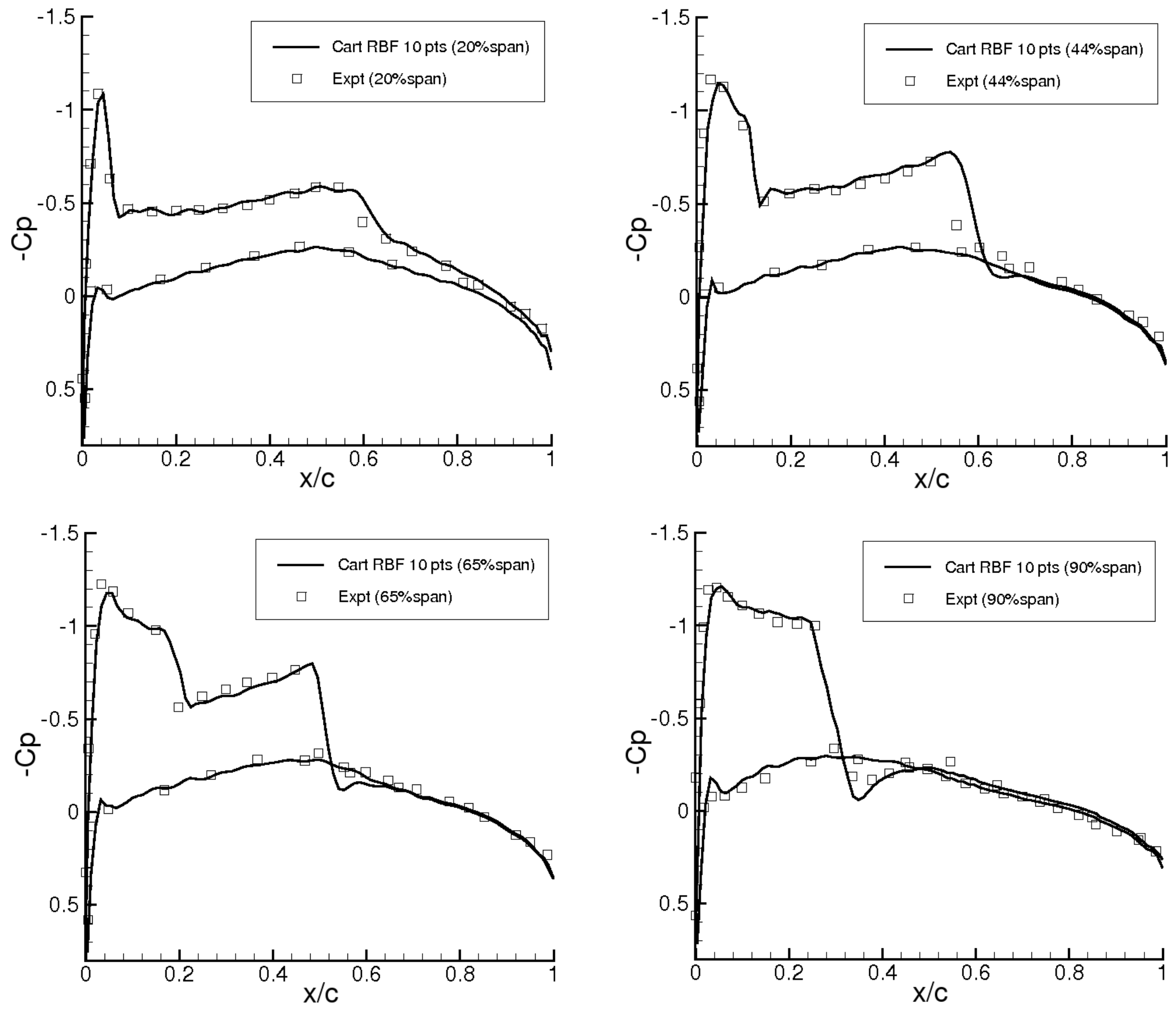


Figure 10. C_p **vs.** x/c **for ONERA M6 wing at four spanwise locations (**$M = 0.84$ **and** $\alpha = 3.06°$**).**

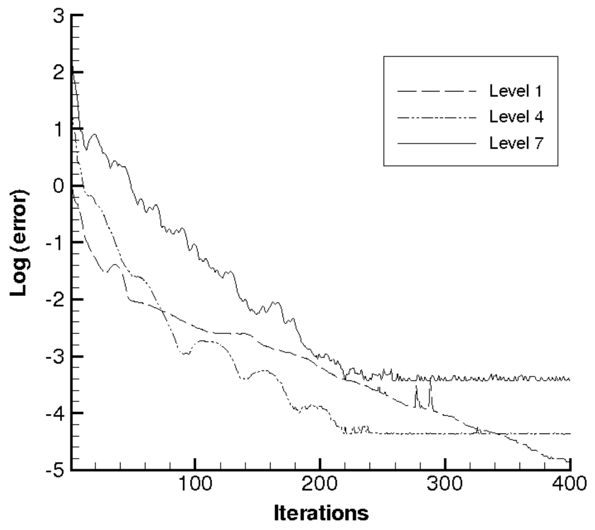


Figure 11. Convergence plot for ONERA M6 wing.

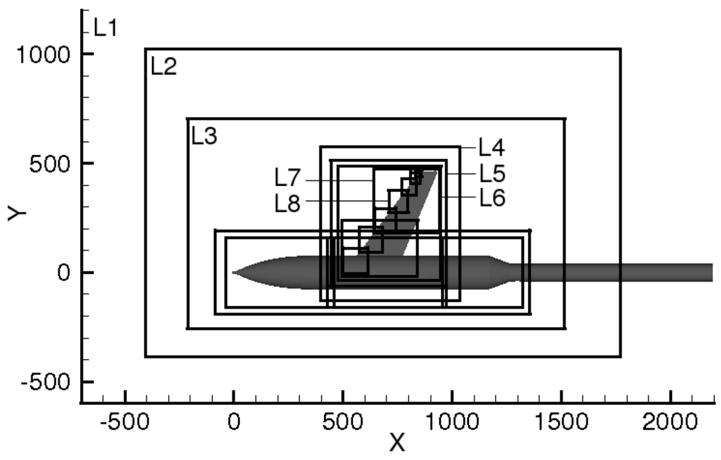


Figure 12. Grid blocks for RAE wing-body.

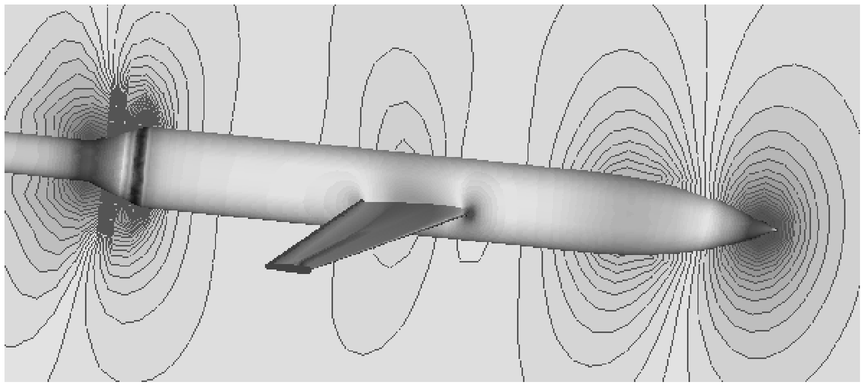


Figure 13. C_p contour for RAE wing-body.

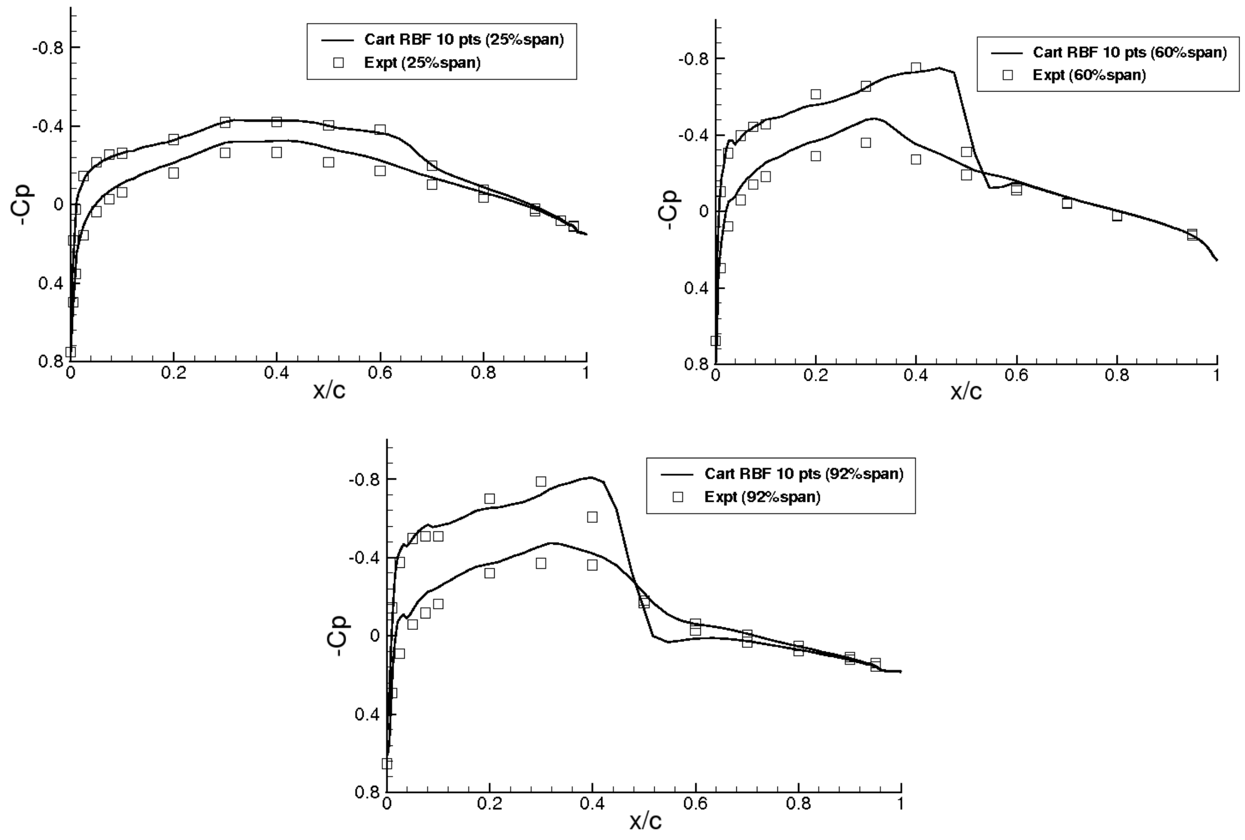


Figure 14. C_p vs. x/c for RAE wing-body at three spanwise locations ($M = 0.9$ and $\alpha = 1.0°$).

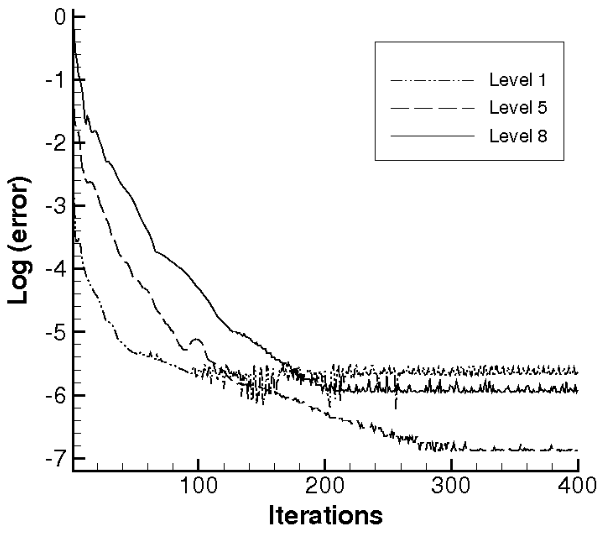


Figure 15. Convergence plot for RAE wing-body.