Commun. Comput. Phys.
doi: 10.4208/cicp.130510.150511s

Vol. **11**, No. 4, pp. 1372-1385
April 2012

Adaptive Cloud Refinement (ACR) – Adaptation in Meshless Framework

M. Somasekhar[1,*], S. Vivek[2], Keshav. S. Malagi[1], V. Ramesh[1] and S. M. Deshpande[3]

[1] *CTFD, CSIR – National Aerospace Laboratories, Bangalore, India.*
[2] *Dept. of Mechanical Engineering, Indian Institute of Technology Madras, Chennai, India.*
[3] *Engineering Mechanics Unit, JNCASR, Bangalore, India.*

Received 15 May 2010; Accepted (in revised version) 15 May 2011

Available online 30 November 2011

Abstract. In the present work adaptation in meshless framework is proposed. The grid adaptation or mesh adaptation is quite well developed area in case of conventional grid based solvers and is popularly known as Adaptive mesh refinement (AMR). In such cases the adaptation is done by subdividing the cells or elements into finer cells or elements. In case of meshless methods there are no cells or elements but only a cloud of points. In this work we propose to achieve the meshless adaptation by locally refining the point density in the regions demanding higher resolution. This results into an adaptive enriched cloud of points. We call this method as Adaptive Cloud Refinement (ACR). The meshless solvers need connectivity information, which is a set of neighboring nodes. It is crucial part of meshless solvers. Obviously because of refining point density, the connectivity of nodes in such regions gets modified and hence has to be updated. An efficient connectivity update must exploit the fact that the node distribution would be largely unaffected except the region of adaptation. Hence connectivity updating needs to be done locally, only in these regions. In this paper we also present an extremely fast algorithm to update connectivity over adapted cloud called as ACU (Automatic Connectivity Update).

AMS subject classifications: 76-04, 76M25, 65M50

Key words: KFVS, LSKUM, meshless methods, adaptation, ACR, ACU.

*Corresponding author. *Email addresses:* somumachani@gmail.com (M. Somasekhar), svivek2006@gmail.com (S. Vivek), keshav@ctfd.cmmacs.ernet.in (K. S. Malagi), vr@ctfd.cmmacs.ernet.in (V. Ramesh), smd@jncasr.ac.in (S. M. Deshpande)

1 Introduction

Mesh adaptation is a technique to reduce the errors in approximations used to solve the PDE's. This can be done by redistributing the grid (r-refinement), refining the grid (h-refinement) or increasing the order of approximation (p-refinement). In the conventional solvers i.e., FDM, FVM or FEM, h-adaptation is done by subdividing the cells or elements into finer ones. These techniques, called Adaptive Mesh Refinement (AMR) are fairly well developed [1, 2]. The idea is to have finer discretization. The subdivision of cells or elements has to be done subject to quality constraints on resulting mesh. In particular the adapted mesh should avoid hanging nodes or edges and highly skewed cells. The subdivision and formation of new cells or elements has to reflect in the book keeping followed for the mesh and the solver. The data of cells, edges and nodes gets modified. In the AMR these challenges have to be faced to achieve adaptation for mesh based solvers.

The meshless methods have been very successful in solving several challenging CFD problems [3–5]. These solvers ease the process of grid generation which is often a bottleneck in CFD simulations. These methods in general require only a cloud of points and connectivity information at each of the nodes, where connectivity is defined as the set of nearest neighbouring nodes to a given node. Hence only node data is sufficient. Managing only node data offers considerable simplicity in book keeping. Further the quality constraints on the cloud of points are far less when compared to mesh based solvers. These observations are very encouraging to explore adaptation in meshless environment. The meshless methods do not contain any cells or elements, hence adaptation by subdividing the cells or elements on mesh based solvers is not relevant here. A new approach has to be followed for adaptation in meshless solvers. Keeping with the idea of finer discretisation we propose to refine the point density in regions demanding higher resolution. This results in an adapted and enriched cloud of points. We call this method as Adaptive cloud refinement (ACR). Connectivity which is defined as the set of neighbouring nodes is the crucial component of meshless solvers. When we refine the point distribution by increasing the local point density, the connectivity of the nodes in that region gets altered. The new nodes which are added do not have any connectivity. Hence the connectivity has to be updated for the nodes from initial cloud and has to be generated for new nodes. This should as well be carried out locally as the connectivity of nodes in unrefined regions is largely unaffected. In this work an efficient algorithm has been developed for this purpose and is called as Automatic Connectivity Update (ACU).

Adaptation through h-refinement essentially involves obtaining solutions on an initial discretized domain. Suitable sensors are used to mark the regions requiring refinement. Refinement in these regions obviously alters the domain discretization locally. The solver is run on the adapted domain to obtain a better solution. Several cycles of adaptation can be carried out to achieve good results i.e., accurate resolution of flow scales. We have chosen Least Squares Kinetic Upwind Method (LSKUM) [6] for computing flow solutions. It is a meshless solver requiring only a cloud of points and the connectivity information.

2 LSKUM: A meshless solver

Least squares kinetic upwind method (LSKUM) is a kinetic theory based meshless solver [6,7]. It needs only point distribution over the domain of interest and connectivity information for each node in the cloud of points. Connectivity is defined as the set of nearest neighbors of a given node which are used for computation at the node. The meshless methods do not need any topological information. LSKUM is based on the fact that suitable moments of Boltzmann equation lead to Euler equations [7]. Numerical schemes are derived at Boltzmann level. Update scheme at Boltzmann level is mapped to Euler level through moment method [6]. An upwind scheme is achieved at Boltzmann level by splitting molecular velocity into positive and negative parts using the Courant-Isaacson-Reeves (CIR) splitting. Spatial derivatives are approximated by using least squares approximation. The second order accuracy can be achieved by using entropy variables and defect correction [8] or modified CIR (MCIR) splitting [9]. The time derivative can be suitably discretized to arrive at update scheme. Much advancement has taken place since its inception [4]. Recently, Konark developed weighted LSKUM (WLSKUM) [10] which increases the robustness of the LSKUM. The MCIR of Ramesh achieves higher order accuracy in single step. These have been implemented in the code being used for the work. A multistage R-K method is used for time stepping. This code has been used to obtain solutions. Error sensors are used with solution so obtained to identify regions requiring adaptation.

3 Formulation of LSKUM

Here we briefly describe the formulation of 2-D LSKUM [6,7]. Consider the 2-D Boltzmann equation

$$\frac{\partial f}{\partial t}+v_1\frac{\partial f}{\partial x}+v_2\frac{\partial f}{\partial y}=J, \tag{3.1}$$

where f is the velocity distribution function, v_1 and v_2 are the Cartesian components of the molecular velocity. J represents a collision term which vanishes in the Euler limit, when f is a Maxwellian distribution F, which in two dimensions is given by

$$F=\frac{\rho}{I_0}\frac{\beta}{\pi}\exp\Big[-\beta(v_1-u_1)^2-\beta(v_2-u_2)^2-\frac{I}{I_0}\Big] \tag{3.2}$$

where $\beta=1/(2RT)$, ρ is the fluid density, I is the internal energy variable, I_0 is the internal energy due to non-translational degrees of freedom, and u_1 and u_2 are the Cartesian components of the fluid velocity, R is the gas constant and T is the absolute temperature of the fluid. Therefore in the Euler limit it is enough to consider

$$\frac{\partial F}{\partial t}+v_1\frac{\partial F}{\partial x}+v_2\frac{\partial F}{\partial y}=0. \tag{3.3}$$

Using CIR splitting, v_1 and v_2 can be written as

$$v_1 = \frac{v_1+|v_1|}{2} + \frac{v_1-|v_1|}{2}, \qquad v_2 = \frac{v_2+|v_2|}{2} + \frac{v_2-|v_2|}{2}. \tag{3.4}$$

Using CIR splitting for the components of molecular velocity, the Boltzmann equation can be written as

$$\frac{\partial F}{\partial t} + \frac{v_1+|v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_1-|v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_2+|v_2|}{2}\frac{\partial F}{\partial y} + \frac{v_2-|v_2|}{2}\frac{\partial F}{\partial y} = 0. \tag{3.5}$$

We define the moment vector Ψ by

$$\Psi = \left[1, v_1, v_2, I + \frac{v_1^2+v_2^2}{2}\right]^T, \tag{3.6}$$

and define the Ψ moment as

$$<\Psi, F> = \int_0^\infty dI \int_{-\infty}^\infty dv_1 \int_{-\infty}^\infty dv_2 \Psi F. \tag{3.7}$$

Now the Ψ moment of the Eq. (3.5) will lead to Kinetic Flux Vector Split Euler equations

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}(GX^+) + \frac{\partial}{\partial x}(GX^-) + \frac{\partial}{\partial y}(GY^+) + + \frac{\partial}{\partial y}(GY^-) = 0, \tag{3.8}$$

where

$$U = <\Psi, F>, \quad GX^\pm = \left\langle \Psi, \frac{v_1 \pm |v_1|}{2} F \right\rangle, \quad GY^\pm = \left\langle \Psi, \frac{v_2 \pm |v_2|}{2} F \right\rangle.$$

U is the state vector given by $U = (\rho, \rho u_1, \rho u_2, \rho e)^T$, e is the internal energy per unit mass given by $e = \frac{\rho}{\rho(\gamma-1)} + \frac{1}{2}(u_1^2+u_2^2)$, $GX^\pm$ and $GY^\pm$ are the split fluxes. In order to develop the update scheme we need to evaluate the spatial derivates of split fluxes. In LSKUM the space derivates are evaluated using least squares approximation [6]. Assume that values of F are available at a node P_0 and its immediate surroundings nodes, $C(P_0)$, referred to as connectivity of point P_0. Then using Taylor series expansion for F around P_0,

$$F_i = F_0 + \Delta x_i F_{x0} + \Delta y_i F_{y0} + H.O.T., \quad \forall i \in C(P_0). \tag{3.9}$$

Considering only the first three terms of the series and defining error 'e_i' as

$$e_i = F_i - (F_0 + \Delta x_i F_{x0} + \Delta y_i F_{y0}) \tag{3.10}$$

and w_i as the weight associated with point 'i', we can take the weighted sum of squares of error and minimize it w.r.t. F_{x0} and F_{y0} to get the first order approximation to the

derivates F_{x0} and F_{y0} which are then given by the following formulae

$$F_{x0}=\frac{\sum w_i\Delta y_i{}^2\sum w_i\Delta x_i\Delta F_i-\sum w_i\Delta x_i\Delta y_i\sum w_i\Delta y_i\Delta F_i}{\sum w_i\Delta x_i{}^2\sum w_i\Delta y_i{}^2-\left(\sum w_i\Delta x_i\Delta y_i\right)^2}, \tag{3.11a}$$

$$F_{y0}=\frac{\sum w_i\Delta x_i{}^2\sum w_i\Delta y_i\Delta F_i-\sum w_i\Delta x_i\Delta y_i\sum w_i\Delta x_i\Delta F_i}{\sum w_i\Delta x_i{}^2\sum w_i\Delta y_i{}^2-\left(\sum w_i\Delta x_i\Delta y_i\right)^2}. \tag{3.11b}$$

The weights can be chosen based on distance or based on eigen weights or a combination of both [10]. Discretising the time derivative in Eq. (3.8) to first order and using least squares formulae Eq. (3.11) to approximate the derivatives of the various split fluxes, a first order update scheme for 2-D KFVS Euler equations can be written as

$$\begin{aligned}U_0^{n+1}=U_0^n-\nabla t\Bigg[&\left\{\frac{\partial}{\partial x}(GX^+)\right\}_{\Delta x_i<0}+\left\{\frac{\partial}{\partial x}(GX^-)\right\}_{\Delta x_i>0}\\&+\left\{\frac{\partial}{\partial x}(GY^+)\right\}_{\Delta y_i<0}+\left\{\frac{\partial}{\partial x}(GY^-)\right\}_{\Delta y_i>0}\Bigg].\end{aligned} \tag{3.12}$$

The spatial derivatives in the above equation are evaluated using least squares formulae Eq. (3.11) over a suitable subset of points in the connectivity (referred to as sub-stencil) to ensure that the signal propagation property is not violated. The subscripts to the various flux derivative approximations in the above equation indicates the sub-stencil chosen from the full connectivity set.

To achieve second order accuracy we follow the modified KFVS (MKFVS) method developed by Ramesh and Deshpande [9]. In MKFVS, modified CIR (MCIR) splitting is used to split the velocity components. The MCIR is given by

$$v_1=\frac{v_1+\phi|v_1|}{2}+\frac{v_1-\phi|v_1|}{2},\qquad v_2=\frac{v_2+\phi|v_2|}{2}+\frac{v_2-\phi|v_2|}{2}, \tag{3.13}$$

where, a parameter ϕ is introduced in the usual CIR splitting (Eq. (3.4)). This parameter helps in reducing the dissipation present in the scheme. Using MCIR splitting for the components of molecular velocity, the Boltzmann equation can be written as

$$\frac{\partial F}{\partial t}+\frac{v_1+\phi|v_1|}{2}\frac{\partial F}{\partial x}+\frac{v_1-\phi|v_1|}{2}\frac{\partial F}{\partial x}+\frac{v_2+\phi|v_2|}{2}\frac{\partial F}{\partial y}+\frac{v_2-\phi|v_2|}{2}\frac{\partial F}{\partial y}=0. \tag{3.14}$$

Now taking the Ψ-moments of the above equation as done early, we get the Modified Kinetic Flux Vector Split (MKFVS) Euler equations

$$\frac{\partial U}{\partial t}+\frac{\partial}{\partial x}(GXm^+)+\frac{\partial}{\partial x}(GXm^-)+\frac{\partial}{\partial y}(GYm^+)+\frac{\partial}{\partial y}(GYm^-)=0, \tag{3.15}$$

where $GXm^{\mp}$ and $GYm^{\mp}$ are modified fluxes given by

$$GXm^{\pm}=\left\langle\Psi,\frac{v_1\pm\phi|v_1|}{2}F\right\rangle\quad\text{and}\quad GYm^{\pm}=\left\langle\Psi,\frac{v_2\pm\phi|v_2|}{2}F\right\rangle.$$

Then we can follow the same discretisation procedure as in Eq. (3.12) and use modified fluxes (MKFVS fluxes) in place of usual fluxes (KFVS fluxes) to obtain the state update equation. If we take $\phi=\Delta x^p$ where 'p' is between 0 and 1 and Δx is the local characteristic length in the connectivity, the resulting state update equation will be second order accurate. This can be easily verified by performing the MPDE analysis where the parameter ϕ appears as a multiplying factor with leading dissipation term which contains Δx as well. It is interesting to note that the structure of the formulae for second order accurate approximation remains the same as formulae for first order accurate approximation.

4 Adaptive Cloud Refinement (ACR)

The meshless methods do not contain any cells or elements; they contain only a cloud of points and connectivity data. Hence the concepts used in conventional methods of adaptation to subdivide cells or elements are not relevant to it. In meshless methods we achieve the finer discretization by increasing the local point density. i.e., adaptation using h-refinement is done by adding more points locally in the regions of rapid flow variations. In the present work we are using LSKUM [6], which is a meshless method, to obtain flow solutions. Then a suitable sensor is used to identify nodes where refinement is necessary. The nodes so identified are called parent nodes. Refinement is carried out by increasing point density around parent nodes. The new nodes added around such parent nodes are called child nodes. Several options are open to increase the point density around parent node. In this work we are using (X) stencil to add four points at some percentage of average radial distance in the connectivity of parent node as shown in the Fig. 1. The point enrichment process might lead to some child nodes crossing the domain of interest. Such nodes are deleted by using blanking algorithm [3].

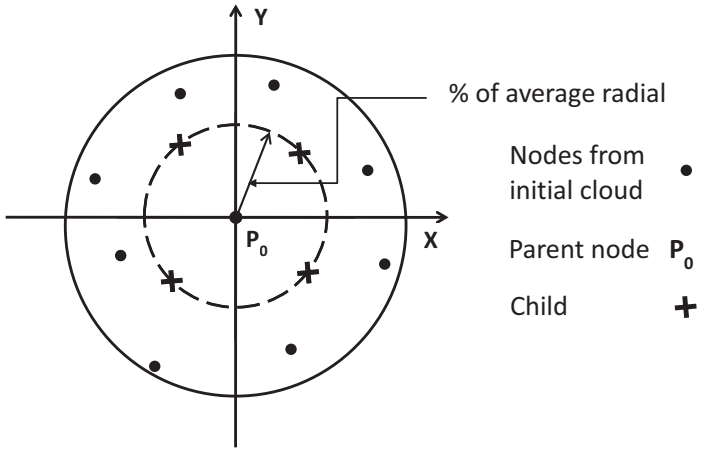

Figure 1: Point enrichment with (X) stencil.

5 Adaptive Connectivity Update (ACU)

Central to the meshless solver is the connectivity [11]. Connectivity is defined as the set of nearest neighbouring nodes to a given node which are used for computations at that node. When we refine the local point density, the neighbourhood of the nodes in that region gets modified. Hence the connectivity of nodes in such regions has to be modified accordingly to reflect the refinement. Also there are new nodes added to the domain. The connectivity of these new nodes has to be generated. Quadtree based connectivity generation [12], which is commonly used in meshless solvers to generate connectivity from cloud of points, can be used for this purpose. This starts with the enriched cloud as one new cloud and generates connectivity for each node afresh. This approach throws away existing connectivity information from initial cloud. It is not local; starts as one new cloud. The node distribution would be largely unaffected except in the region of adaptation. Hence connectivity modification and generation needs to be done only locally in the region by of adaptation. Such a local approach will be more efficient. Hence quadtree is an inefficient approach. In this work we propose an algorithm which locally modifies or generates connectivity only in the region of adaptation. The algorithm makes use of existing information of connectivity from initial cloud to modify or generate the connectivity over refined cloud. This is achieved by considering the connectivity of a parent node and connectivities of all nodes in the connectivity of parent node. The connectivities of these nodes form a superset around a parent node from which we can deduce the connectivity set required. We call this method of generating or modifying the connectivity as Automatic Connectivity Update (ACU). It is further explained below.

Consider any parent node P_0 (node marked for refinement), shown in Fig. 2 by Green Square. Let $C(P_0)$ be its connectivity from initial cloud, shown by black circles in Fig. 2 and $N_c(P_0)$ be the set of newly added nodes, shown in Fig. 2 by red dots. These are child nodes of P_0. $C(P_0)=\{P_i,\ i=1,\cdots,m\}$ where m is the number of nodes in $C(P_0)$. Similarly, $C(P_i)$ is the connectivity of any node $P_i \in C(P_0)$. Modify these connectivities from initial cloud by adding child nodes. The modified connectivities will then be

$$\widetilde{C}(P_0)=C(P_0)\cup N_c(P_0), \qquad \widetilde{C}(P_i)=C(P_i)\cup N_c(P_i).$$

Fig. 3 shows the modified connectivity for P_0. Let $C_s(P_0)$ be the super set of connectivities which include the above mentioned connectivities.

$$C_s(P_0)=\widetilde{C}(P_0)\cup\{\widetilde{C}(P_i),\ i=1,\cdots,m\}.$$

This set is shown in Fig. 4 by blue squares. The connectivity of any child node of P_0 will be a subset of $C_s(P_0)$. Hence the connectivities of all children of P_0 can be deduced from the $C_s(P_0)$. Fig. 5 shows the connectivity generated for a child node using superset $C_s(P_0)$. The connectivity of P_0 can also be modified using this superset to reflect the finer distribution of nodes due to adaptive refinement of nodes in the domain. Fig. 6 shows the modified final connectivity of parent node. Thus the connectivity can be generated for child nodes as well as connectivity can be updated for parent nodes using this algorithm.

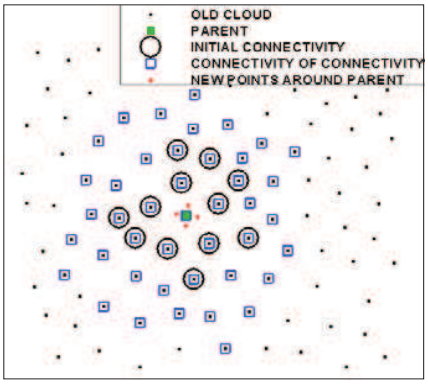

Figure 2: Set of old points to build connectivity.

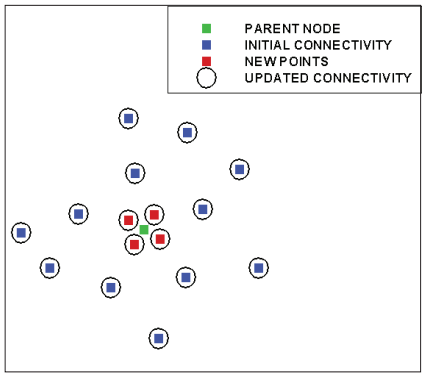

Figure 3: Modified Connectivity including children.

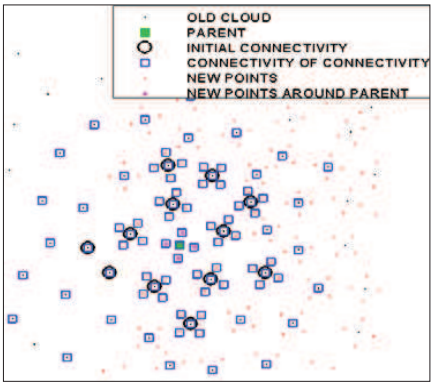

Figure 4: Connectivity super set.

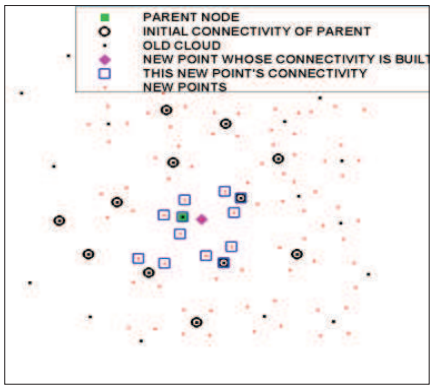

Figure 5: Generated connectivity of a child.

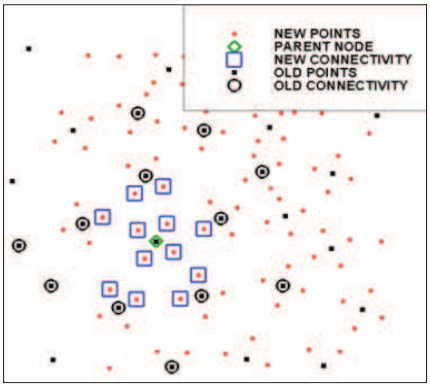

Figure 6: Final modified connectivity of parent node.

6 Sensors

In adaptation, sensors are used to identify regions requiring refinement. The sensors generally try to locate such regions by measuring the variation in some flow parameters. In the present work we have not made any studies on sensors. A brief description is included for the sake of completeness. Commonly used sensors are pressure gradient, density gradient, vorticity, entropy etc. We have used pressure gradient, entropy and D^2 distance [14] based sensor in our work. In multivariate statistics, a measure of the distance between two multivariate normal distributions is given by the directed divergence or D^2 distance. Here distance is not the Euclidean distance. It is a measure of how different two distributions are. Since the Maxwellian velocity distribution is also a normal distribution, a sensor at the Boltzmann level can be formulated in detecting two different Maxwellian distributions using this directed divergence. The corresponding sensor at the Euler level is obtained by taking Ψ moments of the Boltzmann level sensor. This sensor is called the Mahalanobish distance. In 2D the expression for D^2 sensor is

$$D(F_i,F_j)=\rho_i\Big(\frac{\rho_i}{\rho_j}-1\Big)\ln\left[\frac{\rho_j}{\rho_i}\Big(\frac{T_i}{T_j}\Big)^2\right]+\rho_i\Big(\frac{\rho_j}{\rho_i}+\frac{T_i}{T_j}\Big)\frac{(u_{1i}-u_{1j})^2+(u_{2i}-u_{2j})^2}{2RT_i}$$
$$+2\rho_i\Big(\frac{T_j}{T_i}-1\Big)\Big(\frac{\rho_j}{\rho_i}-\frac{T_i}{T_j}\Big). \tag{6.1}$$

7 Results and discussions

The ACR along with new connectivity generation algorithm, ACU has been tested on the standard NACA0012 2D test cases and MDA three element airfoil to assess the performance of enrichment and connectivity generation. In the following figures the black dots represent initial cloud points and red dots represent newly added points.

7.1 NACA0012 test cases

The LSKUM [6] solver has been used to obtain the solution. The initial cloud of points is obtained using unstructured grid with 49,046 points. The number of points on the body are 240 and number of points on the outer boundary are 120. The solution on the initial cloud is obtained using LSKUM solver. Suitable sensors have been used to identify the nodes for refinement. Enrichment and connectivity generation with ACR & ACU are carried out to generate the adapted cloud of points with connectivity information.

7.1.1 Transonic test case $M_\infty=0.85$, $\alpha=1^0$

The transonic test case has a shock on the upper surface and a weak shock on the lower surface. In this test case we have used D^2 based sensor [13], which is a measure of distance between two Maxwellians, to identify regions requiring refinement. Fig. 7 & Fig. 8 show the initial and adapted point distribution after one cycle of adaptation respectively. It clearly shows that refinement has taken place in the regions having dominant flow

Table 1: $M_\infty = 0.85$, $\alpha = 1^0$ Transonic Case.

Cloud from Unstructured Grid	C_l	C_d
Initial	0.379	0.062
Enriched	0.343	0.055
AGARD	0.330-0.389	0.0464-0.0590

Table 2: $M_\infty = 1.2$, $\alpha = 0^0$ Supersonic Case.

Cloud from Unstructured Grid	C_l	C_d
Initial	0.000975	0.0956
Enriched	-0.000399	0.0953
AGARD	0	0.0946-0.096

Table 3: $M_\infty = 0.63$, $\alpha = 2^0$ Subsonic Case.

Cloud from Unstructured Grid	C_l	C_d
Initial	0.346	0.005
Enriched	0.38	0.00036
GAMM	0.329-0.336	0.0003-0.0007

features. The total number of points after adaptation is 56226. Fig. 9 & Fig. 10 shows the pressure contours on initial cloud and adapted cloud. Fig. 11 shows comparison of C_p plots for adapted cloud and initial cloud. We can easily see that the flow features have been captured more accurately on the adapted cloud. Table 1 shows the C_L and C_D comparison with AGARD [14] results. The C_D has been predicted well on the refined cloud.

7.1.2 Supersonic test case $M_\infty = 1.2$, $\alpha = 0^0$

The supersonic test case has a bow shock ahead of airfoil and fish tail shock at trailing edge. In this test case we have used D^2 based sensor [14] to identify regions requiring refinement. Fig. 12 shows the adapted cloud of points. The total number of points after enrichment is 60674. It is clearly seen that refinement has taken place in the regions of bow shock and fish tail shocks. Pressure contours (Fig. 13 & Fig. 14) indicate improved performance on the adaptive cloud. Table 2 shows C_L and C_D comparison with AGARD [13] results. C_L should be zero for this case which is predicted well by the adapted cloud.

7.1.3 Subsonic test case $M_\infty = 0.63$, $\alpha = 2^0$

In this case there are rapid flow variations near the leading edge. The suction peak appearing near leading edge is of interest. In this case we have used an entropy based sensor to capture the entropy layer. The enriched cloud Fig. 15 shows refinement in this region. The C_p plot Fig. 16 shows that suction peak is captured well by adapted cloud. The total number of points after enrichment in this case is 74222. For this test case C_D should be zero. Note from the Table 3 that C_D is more close to zero on the refined cloud.

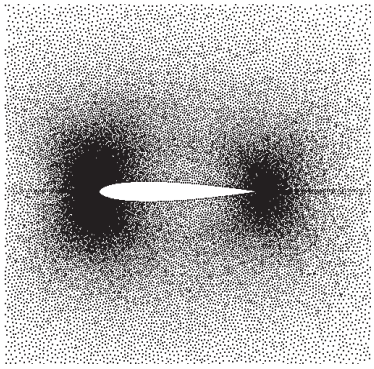

Figure 7: Initial Cloud.

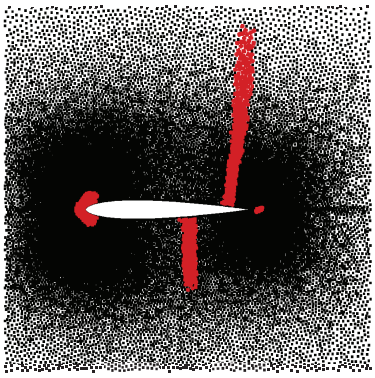

Figure 8: Adapted Cloud.

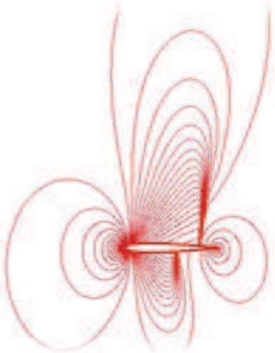

Figure 9: Pressure contours on initial cloud.

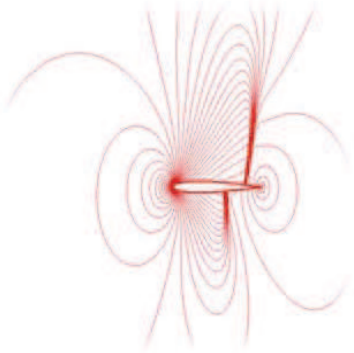

Figure 10: Pressure contours on adapted cloud.

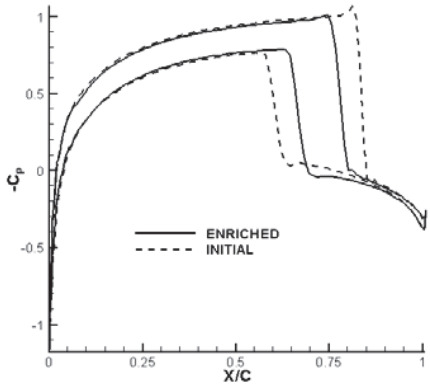

Figure 11: C_p plots for Transonic case.

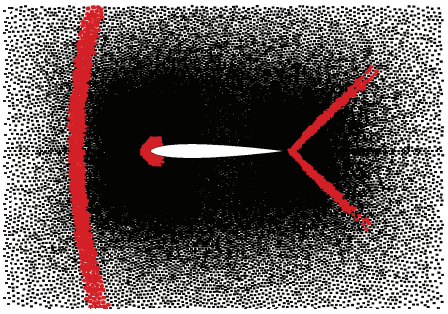

Figure 12: Adapted point Distribution for supersonic case.

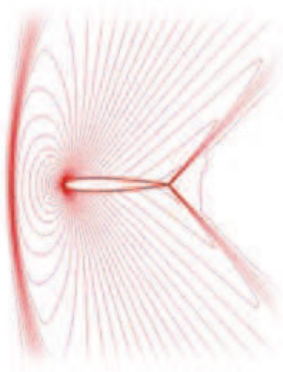

Figure 13: Pressure contours on initial cloud.

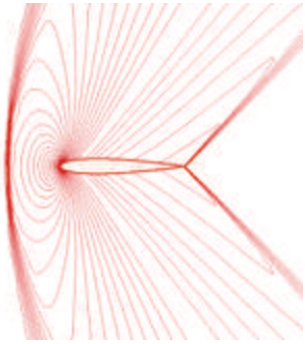

Figure 14: Pressure contours on adapted cloud.

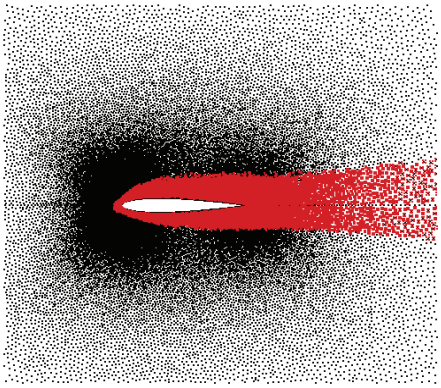

Figure 15: Adapted cloud.

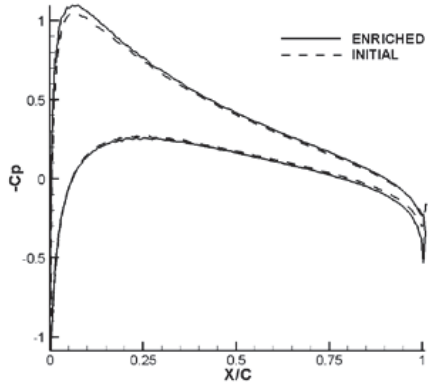

Figure 16: Cp plots for subsonic case.

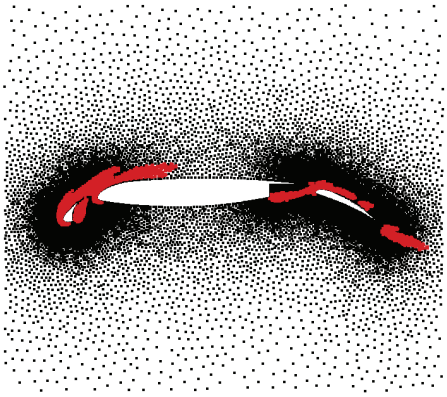

Figure 17: Adapted cloud.

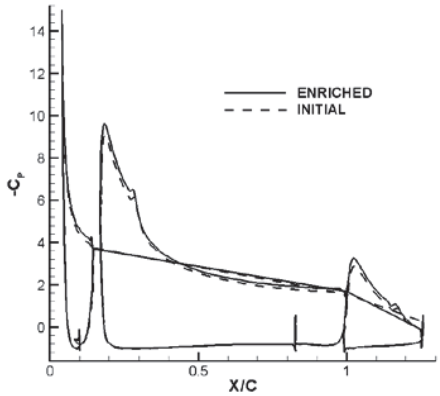

Figure 18: Cp plots for subsonic case.

7.2 Subsonic test case MDA 3 element airfoil

This is a high lift configuration with free stream Mach number 0.2 and angle of attack 16° [15]. Like the earlier subsonic test case, in this case too there are rapid flow variations near each of the leading and trailing edges of the 3 elements present. The slat surface has 141 points. The main airfoil surface is made up of 267 points. The flap is made up of 191 points. In all, the inner boundaries are made of 599 points. The outer boundary is made up of 80 points. In order to find out the exact locations where gradients are high, we have used an entropy gradient sensor. The initial distribution (black points) has 23092 points. The enriched distribution Fig. 17 has 34060 points. The suction peak on the slat has improved from 13 to nearly 15 (Fig. 18) as a consequence of ACR.

8 Conclusion

We have successfully demonstrated an adaptation methodology (ACR+ACU) for meshless methods. The strategy employed for adaptation of cloud of points is to increase the local point density and is termed as Adaptive Cloud refinement (ACR). The refinement leads to alteration of neighbourhood, necessitating connectivity updation process. This has been very efficiently addressed by developing Automatic Connectivity Update (ACU) algorithm. The efficiency of ACU is due to the fact that we are able to exploit locality of refinement and prior connectivity information from initial cloud while generating connectivity unlike tree based searches, used commonly in meshless methods, which are not local in nature. The ACR along with ACU has been successfully demonstrated on standard test cases of NACA 0012 airfoil and MDA 3 element airfoil configuration. The capture of sharp gradients and better values of coefficients over adapted cloud in the standard test case of NACA 0012 airfoil and the complicated MDA 3 element airfoil configuration have demonstrated the success of ACR with ACU. The algorithm is directly applicable to adaptation in 3D. The changes required in code for adaptation in 3D are very minimal as one needs to work with only node data.

Even though AMR is pretty matured and ACR is still in developmental stage, some observations are worth noting when we compare ACR with AMR. In AMR one needs to take care of hanging nodes and edges [16]. This would lead to refinement even in cells or elements not marked for refinement causing excessive refinement. Such a problem does not arise in ACR. The points can be added arbitrarily if one decides to do so. Data structure requirements for ACR present a good case for it. The meshless methods require data only from the connectivity set. Hence data structure would consist of nodes in connectivity set and other data at these nodes. The ACR also uses the similar data structure and only needs node data. The data structure of AMR generally needs node, edge, element, data structure [16]. The data structure becomes further complicated in case of hybrid grids and hierarchical grids [17]. Obviously the overheads due to data structure of adaptation in ACR are far less when compared to AMR.

Even though the present work has dealt with only inviscid flows, some comments

regarding viscous flows are relevant here. In the case of viscous flows the point distribution or points cloud and connectivity will have in general some directionality i.e., the distribution of points in the connectivity will be anisotropic to resolve the flow features properly. In the present work we have adopted the approach of adding points isotropically. This approach needs to be changed when handling adaptation with viscous flows. Some measure of directionality has to be considered while adding points. While this seems not so difficult, further research and study are required to establish the methodology. However the methodology to update the connectivity, ACU, would rather remain very much similar.

References

[1] M.J. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys., 82, pp. 67-84, 1989.
[2] R. Lohner, Mesh adaptation in fluid mechanics, Eng. Fract. Mech., 50(5), pp. 819-847, 1995.
[3] G. Harish, M. Pavanakumar, K. Anandhanarayanan, Store separation dynamics using grid-free Euler solver, 24th Applied Aerodynamics Conference, AIAA 2006-3650, June 2006.
[4] S.M. Deshpande, Theory and Application of Kinetic Method to Aerospace CFD, Keynote lecture in East West Speed Flow Field Conference (EWSFF 2005), Beijing, Oct 19-22, 2005.
[5] V. Ramesh, S.M. Deshpande, Unsteady flow computations for flow past multiple moving boundaries using LSKUM, Comput. Fluids, 36(10), pp. 1592-1608, 2007.
[6] A.K. Ghosh, S.M. Deshpande, Least squares kinetic upwind method for inviscid compressible flows, AIAA Paper 95-1735, 1995.
[7] J.C. Mandal, S.M. Deshpande, Kinetic flux vector splitting for Euler equations, Comput. Fluids, 23(2), pp. 447-478, 1994.
[8] S.M. Deshpande, Anandanarayanan, C. Praveen, V. Ramesh, Theory and applications of 3-D LSKUM based on entropy variables, Int. J. Numer. Meth. Fluids, 40(1-2), pp. 47-62, 2002.
[9] V. Ramesh, S.M. Deshpande, Low dissipation grid free upwind kinetic scheme with modified CIR splitting, Fluid Mechanics Report, (2004), FM 20, Centre of Excellence in Aerospace CFD, Dept. of Aero. Eng., Indian Institute of Science, Bangalore.
[10] K. Arora, N.K.S. Rajan, S.M. Deshpande, Weighted Least Squares Kinetic Upwind Method (WLSKUM) for computation of flowthrough blade passage with Kinetic Periodic Boundary Condition (KPBC), CFD J., 16(3), pp. 31-48, 2005.
[11] C. Praveen, A.K. Ghosh, S.M. Deshpande, Positivity preservation, stencil selection and applications of LSKUM to 3-D inviscid flows, Comput. Fluids, 38(8), pp. 1481-1494, 2009.
[12] V. Ramesh, Least Squares Grid Free Kinetic Upwind Method, Ph.D Thesis, Indian Institute of Science, Bangalore, July 2001.
[13] AGARD-AR-211, Test cases for inviscid flow field methods.
[14] N.V. Raghavendra, D^2-Distance bases 3-D Grid Adaptation for a Generic Fighter Aircraft Wing, ME Thesis, Indian Institute of Science, Bangalore, 2000.
[15] Z.J. Wang, A fast nested multi-grid viscous flow solver for adaptive Cartesian/Quad grids, Int. J. Numer. Meth. Fluids, 33, pp. 657-680, 2000.
[16] Y. Kallinderis, C. Kavouklis, A dynamic adaptation for general 3-D hybrid meshes, Comput. Meth. App. Mech. Eng., 194, pp. 5019-5050, 2005.
[17] D.J. Mavriplis, Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes, Int. J. Numer. Meth. Fluids, 34, pp. 93-111, 2000.