Fluid Mechanics Report

Weighted Least Squares Kinetic Upwind Method using Eigenvector Basis

Konark Arora

PhD. Student, CFD Centre
Department of Aerospace Engineering
Indian Institute of Science
Bangalore 560012, India
Email: *arora@aero.iisc.ernet.in*

S.M. Deshpande

Professor, Engineering Mechanics Unit
Jawaharlal Nehru Centre for Advanced Scientific Research,Jakkur,
Bangalore 560064, India
Email: *smd@jncasr.ac.in*

Abstract

The least squares grid-free method, though having the ability to work effectively on any distribution of points is limited by the requirement of a good connectivity around a node. This report deals with a fundamental improvement over the usual least squares grid-free method to overcome this limitation of the least squares grid-free methods. The new approach involves the use of the weights to diagonalize the least squares matrix A such that the x and y directions become the eigen directions along which the higher dimensional least squares formulae reduce to the one dimensional formulae. A very important advantage of this approach (apart from improving the convergence characteristics of the grid-free solver) is that it helps in tackling the problems of code divergence due to the degenerate and other cases of bad connectivity. Appropriate methods of finding suitable weights to diagonalize the two and three dimensional least squares matrix have been discussed in this report. Finally, some two dimensional results have been given in support of our claim.

1 Introduction

The numerical solution of the partial differential equations requires the discretization of the computational domain as well as the discretization of the spatial derivatives at discrete points (sometimes called nodes) in the computational domain. The process of discretization of the computational domain into discrete points or finite volumes is generally called grid generation. This is an essential prerequisite for the numerical solution of the partial differential equations. This is by no means a trivial task and consumes many man-hours and large computer time as well. To overcome this difficulty (ie. to reduce man-hours and to make grid generation less dependent on user intervention), research has been going on in the field of grid-free methods since a number of years [1, 2, 3, 6, 7, 8]. It has been proved to be extremely successful and holds out a lot of promise. The main advantage of the grid free methods is their ability to work on any arbitrary distribution of points. Compared to the grid generation, point generation is a relatively simple task [1, 11]. The grid generated has to conform to the boundaries of the body. Further, in case of complex configurations, it is essential to resolve the fine geometric features (sharp edges, wing-body intersections, trailing and leading edges, etc.) and this is an extremely difficult task. FAME [7] is an attempt to generate multiple chimera grids with clustering to resolve several geometric details. The grid dependent solvers while working on chimera mesh need to have a higher order accurate interpolation strategy so as to transfer data from one grid to another. This results in a loss of accuracy for these solvers in the overlap regions, which is not so in the case of grid-free methods. Grid-free methods with a higher density of points in required regions are able to resolve the geometric details without a consequent penalty of loss of accuracy in the solution even when points are generated by different grids or different methods. However, the calculation of the derivatives at a point requires the neighbouring information. The points in the neighbourhood of a node are called the "connectivity" of the node. It has been found that a good connectivity of the nodes in the computational domain is very important for the successful use of the grid-free solvers. Bad connectivity leads not only to the loss in accuracy of the computations but even to the code divergence [7]. There are a number of cases of bad connectivity which affect the accuracy of the grid-free solvers [7]. In a nutshell, it can be said that now the problem of grid generation has been replaced by the problem of appropriate point and connectivity generation. The present report on the other hand approaches the problem from another angle. This report deals with a fundamental improvement over the usual least squares grid-free method. Numerical experiments conducted show that the new approach improves the convergence characteristics of the grid-free solver. An added and very important advantage of the new approach is that it helps in tackling the problems of code divergence due to the degenerate and other cases of bad connectivity [7]. Here in this report, the least squares and the weighted least squares grid-free method will be first explained followed by the basic idea behind the new approach. The various cases of bad and degenerate connectivity will be explained and it will be shown how these can be tackled with the new approach. Finally some results will be discussed in support of the claim of the advantages of the new approach.

2 *Least Squares Method*

The basic idea behind the method is to obtain the derivative of a function at any node by minimizing the sum of the squares of the error. Consider in 1D, a distribution of points

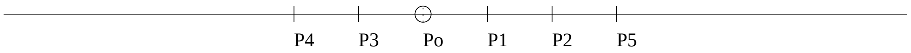


Figure 1: Point distribution for 1D least squares formula

as shown in Fig.1. Suppose it is desired to get the derivative of a function $F(x)$ at point P_o shown in Fig.1. Expand F_i around point P_o in terms of Taylors series :

$$F_i = F_o + (x_i - x_o)F_{xo} + O(\Delta x)^2 \tag{2.1}$$

Define

$$\Delta x_i = x_i - x_o, \qquad \Delta F_i = F_i - F_o$$

Taking the RHS in Eq.(2.1) on LHS and neglecting the higher order terms, we get the error e_i defined as

$$e_i = (\Delta F_i - \Delta x_i F_{xo}) \tag{2.2}$$

then the sum of the squares of errors or deviations at point P_o is given by

$$E = \sum_{i=1}^{p} e_i^2 = \sum_{i=1}^{p} (\Delta F_i - \Delta x_i F_{xo})^2 \tag{2.3}$$

where p is the number of points in the stencil or connectivity. Minimizing E in Eq.(2.3) with respect to F_{xo} and simplifying, we get the first order accurate least squares formula for the derivative in one dimension as

$$F_{xo} = \frac{\sum_{i=1}^{p} \Delta x_i \Delta F_i}{\sum_{i=1}^{p} {\Delta x_i}^2} \tag{2.4}$$

For 2-D, the system of equations that has to be solved to obtain the value of the derivatives is

$$A\,(grad\ F)_o^T = b \tag{2.5}$$

where the vector $(grad\ F)_o^T$ has the meaning

$$(grad\ F)_o^T = \begin{bmatrix} F_{xo} \\ F_{y_o} \end{bmatrix}$$

The matrix A and the corresponding vector b in two dimensions are :

$$A = \begin{bmatrix} \sum {\Delta x_i}^2 & \sum \Delta x_i \Delta y_i \\ \sum \Delta x_i \Delta y_i & \sum {\Delta y_i}^2 \end{bmatrix}, b = \begin{bmatrix} \sum \Delta x_i \Delta F_i \\ \sum \Delta y_i \Delta F_i \end{bmatrix} \tag{2.6}$$

In three dimension, the matrix and corresponding vectors are :

$$A = \begin{bmatrix} \sum {\Delta x_i}^2 & \sum \Delta x_i \Delta y_i & \sum \Delta x_i \Delta z_i \\ \sum \Delta x_i \Delta y_i & \sum {\Delta y_i}^2 & \sum \Delta y_i \Delta z_i \\ \sum \Delta x_i \Delta z_i & \sum \Delta y_i \Delta z_i & \sum {\Delta z_i}^2 \end{bmatrix},$$

$$(grad\ F)_o^T = \begin{bmatrix} F_{xo} \\ F_{y_o} \\ F_{zo} \end{bmatrix}, b = \begin{bmatrix} \sum \Delta x_i \Delta F_i \\ \sum \Delta y_i \Delta F_i \\ \sum \Delta z_i \Delta F_i \end{bmatrix} \tag{2.7}$$

The formulae for the derivatives obtained in Eqs.(2.4),(2.6) and (2.7) above are first order accurate as the Taylor's series used to derive this formula has been truncated to $O(\Delta x)^2$. To increase the order of accuracy of the formulae, the Taylor's series ought to be truncated to $O(\Delta x)^3$. There are two ways of deriving the second order accurate least squares formulae. The first way is to proceed in exactly the same manner as described above, and minimizing the sum of the squares of the error with respect to F_{xo} and F_{xxo}. Thus we have to solve a system of equations to get second order accurate least squares formula for the first derivative. Again referring to Fig.1, expanding the function $F(x)$

about point P_o in terms of Taylor series and retaining more terms so as to achieve second order accuracy, the sum of the squares of the errors is

$$E = \sum_{i=1}^{p} \left(\Delta F_i - \Delta x_i F_{xo} - \frac{\Delta {x_i}^2}{2} F_{xxo} \right)^2 \tag{2.8}$$

After minimizing the error with respect to F_{xo} and F_{xxo}, the system of equations required to be solved to get second order accurate first derivative in 1-D is

$$\begin{bmatrix} \sum \Delta {x_i}^2 & \sum \dfrac{\Delta {x_i}^3}{2} \\ \sum \dfrac{\Delta {x_i}^3}{2} & \sum \dfrac{\Delta {x_i}^4}{4} \end{bmatrix} \begin{bmatrix} F_{xo} \\ F_{xxo} \end{bmatrix} = \begin{bmatrix} \sum \Delta x_i \Delta F_i \\ \sum \dfrac{\Delta {x_i}^2}{2} \Delta F_i \end{bmatrix} \tag{2.9}$$

Thus we see that even to find the first derivative up to second order accuracy, the above approach results in loss of simplicity and additional equations have to be solved. A different method called the defect correction therefore is used [2] to obtain higher order accurate least squares formulae. The advantages of this approach will be discussed shortly. Eq.(2.1) gives the Taylors series truncated up to $O(\Delta x)^2$. Taking the derivative of this series with respect to x, we get

$$F_{xi} = F_{xo} + \Delta x_i F_{xxo} + HOT$$

On simplifying, we get

$$\Delta F_{xi} \Delta x_i = \Delta {x_i}^2 F_{xxo} + HOT \tag{2.10}$$

Substituting Eq.(2.10) in Eq.(2.8), the sum of the squares of the error becomes

$$E = \sum_{i=1}^{p} \left(\Delta F_i - \Delta x_i F_{xo} - \frac{\Delta x_i}{2} \Delta F_{xi} \right)^2 \tag{2.11}$$

Define the modified difference as :

$$\Delta \widetilde{F} = \Delta F - \frac{\Delta x_i}{2} \Delta F_{xi}$$

In terms of this modified difference, Eq.(2.11) becomes

$$E = \sum_{i=1}^{p} \left(\Delta \widetilde{F}_i - \Delta x_i F_{xo} \right)^2 \tag{2.12}$$

On minimizing the error, the second order accurate least squares formula is

$$F_{xo} = \frac{\sum_{i=1}^{p} \Delta x_i \Delta \widetilde{F}_i}{\sum_{i=1}^{p} \Delta {x_i}^2} \tag{2.13}$$

Similarly to get second order accurate least squares formula in two dimensions, ΔF_i in Eq.(2.6) is replaced by $\Delta \widetilde{F}_i$ where

$$\Delta \widetilde{F}_i = \Delta F_i - \frac{\Delta x_i}{2} \Delta F_{xi} - \frac{\Delta y_i}{2} \Delta F_{y_i}$$

On similar lines, to get second order accurate least squares formula in 3-D, ΔF in Eq.(2.7) is replaced by $\Delta \widetilde{F}$ where

$$\Delta \widetilde{F}_i = \Delta F_i - \frac{\Delta x_i}{2} \Delta F_{xi} - \frac{\Delta y_i}{2} \Delta F_{y_i} - \frac{\Delta z_i}{2} \Delta F_{zi}$$

It is important to note a few points in the above defect correction approach :

(1) In this method, we need to know the value of F_{xo} to second order accuracy to get F_{xo} second order accurate. But since this itself is the quantity being calculated, we take the initial estimate as the first order accurate F_{xo} that can be calculated and then perform inner iterations to correct the value of F_{xo}.

(2) There is no change in the formula used to get the first order and the second order derivatives. Only a modified difference $\Delta \widetilde{F}_i$ defined above appears in the second order accurate formulae. This enables the same routine to be used in calculating the first order accurate as well as second order accurate derivatives.

(3) As compared to the first approach, the second approach (defect correction approach) is simple. Since the system of linear algebraic equations has the same matrix A, all its good characteristics listed below are retained and these are made use of in the second order accurate formulae.

2.1 Properties of Least Squares Matrix A

The least squares matrix A obtained above has several interesting mathematical and geometrical characteristics.

(a) The matrix A is purely a geometric matrix. All the elements of this matrix are functions of coordinates of nodes in the connectivity. This matrix can be inverted to find the derivatives at a node. This directly explains the importance of the connectivity of the node in grid-free solvers. The bad connectivity [7] can degrade solution accuracy due to singularity or ill-conditioning of A [7] or due to some other cause. Sometimes, even a well conditioned matrix coupled with a bad algorithm can give unacceptable solution.

(b) The matrix A is a real symmetric matrix. From the matrix theory, we know that a real symmetric matrix has all real eigen-values and a complete set of real distinct eigenvectors. Assume e_1 and e_2 to be the two eigenvectors of A and λ_1 and λ_2 be the corresponding eigen-values. We then have

$$Ae_1 = \lambda_1 e_1$$

$$Ae_2 = \lambda_2 e_2$$

Now, the scalar product of Ae_1 and e_1 is :

$$(Ae_1, e_1) = \lambda_1 (e_1, e_1) = \lambda_1 \|e_1\|^2 \tag{2.14}$$

and

$$(Ae_2, e_2) = \lambda_2 (e_2, e_2) = \lambda_2 \|e_2\|^2 \tag{2.15}$$

But we know from linear algebra that

$$(Aa, b) = \left(a, A^T b\right)$$

where A^T is the transpose of matrix A. For symmetric matrix, $A = A^T$, Hence

$$(Aa, b) = (a, Ab)$$

Now considering the scalar product,

$$(Ae_1, e_2) = (\lambda_1 e_1, e_2) = \lambda_1 (e_1, e_2) \tag{2.16}$$

and

$$\begin{aligned}(Ae_2, e_1) = (\lambda_2 e_2, e_1) = \lambda_2 (e_2, e_1) = \lambda_2 (e_1, e_2) = \\ (Ae_2)^T e_1 = {e_2}^T Ae_1 = (e_2, Ae_1) = (Ae_1, e_2)\end{aligned} \tag{2.17}$$

Now subtracting Eqs.(2.17) and (2.16) above, we get :

$$(Ae_2, e_1) - (Ae_1, e_2) = 0 = (\lambda_1 - \lambda_2)(e_1, e_2) \tag{2.18}$$

Hence if $\lambda_1 \neq \lambda_2$ then e_1 and e_2 must be orthogonal. By a theorem in Linear Algebra, if A is a symmetric matrix, then it is orthogonally diagonalizable [10]. Thus, even if the eigenvalues of A are equal, matrix A will be diagonalizable, but then eigenvectors of A are linearly independent and not necessarily orthogonal. However using Gram-Schmidt method, an orthogonal set of eigen-vectors can be easily constructed. Thus in this case, we can still obtain a new set of orthogonal basis even if two or more eigen-values are equal.

(c) If we rotate the coordinate frame from (x, y, z) in 3-D to (x', y', z') such that each of the new coordinate directions is an eigenvector of A, we will obtain a diagonalized matrix A'. It is interesting to note that in this new rotated frame, the least squares formulae for the derivatives reduce to the one dimensional formulae along each new eigendirection. Considering the two dimensional example, the least squares formula for the x-derivative in (x, y) frame is

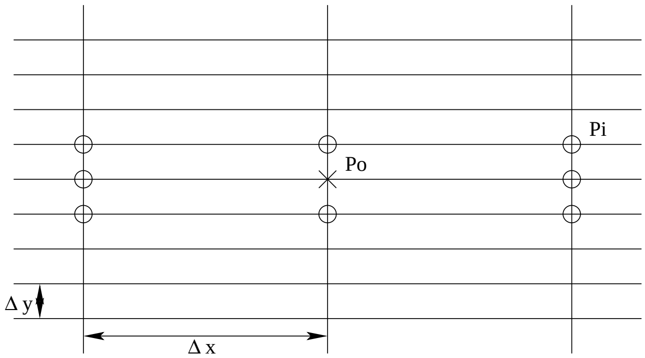
$$\left(\frac{\partial F}{\partial x}\right)_o = F_{xo} = \frac{\sum \Delta {y_i}^2 \sum \Delta x_i \Delta F_i - \sum \Delta x_i \Delta y_i \sum \Delta y_i \Delta F_i}{\sum \Delta {x_i}^2 \sum \Delta {y_i}^2 - \left(\sum \Delta x_i \Delta y_i\right)^2} \tag{2.19}$$

Along the new coordinate directions, $\sum \Delta x'_i \Delta y'_i = 0$, so the above formula reduces to

$$\left(\frac{\partial F}{\partial x'}\right)_o = \frac{\sum \Delta {y'}_i^2 \sum \Delta {x'}_i \Delta F_i}{\sum \Delta {x'}_i^2 \sum \Delta {y'}_i^2} = \frac{\sum \Delta {x'}_i \Delta F_i}{\sum \Delta {x'}_i^2} \tag{2.20}$$

which is one dimensional formula along the x' direction.

Let us study the advantages in using 1-D formula. Consider a uniform structured but highly stretched cartesian grid shown in the Fig.2 The figure shows the point



Uniform cartesian grid, node Po denoted by cross, with its connectivity denoted by circ

Figure 2: Highly stretched cartesian grid showing a node and its connectivity

P_o indicated by a cross and its connectivity points are indicated by circles. Let us find the derivative at the point P_o with least squares method using the connectivity points shown in the figure using Eq.(2.19). The least squares matrix A for such a connectivity is

$$A = \begin{bmatrix} \sum \Delta {x_i}^2 & \sum \Delta x_i \Delta y_i \\ \sum \Delta x_i \Delta y_i & \sum \Delta {y_i}^2 \end{bmatrix} \tag{2.21}$$

It is easily observed that the cross product term in the matrix A vanishes for the connectivity shown in the figure. Thus the least squares matrix reduces to

$$A = \begin{bmatrix} \sum \Delta {x_i}^2 & 0.0 \\ 0.0 & \sum \Delta {y_i}^2 \end{bmatrix} \tag{2.22}$$

The eigen-values of the matrix are now $\sum \Delta {x_i}^2$ and $\sum \Delta {y_i}^2$. Since the grid shown in the Fig.2 is highly stretched, $\Delta y \ll \Delta x$. Hence, the matrix A is highly illconditioned. Use of 2-D formula Eq.(2.19) for such a case leads nearly to 0/0 singularity because

$$\sum \Delta {y_i}^2 \ll \sum \Delta {x_i}^2, \qquad \sum \Delta x_i \Delta y_i = 0$$

Even when $\sum \Delta x_i \Delta y_i$ does not exactly vanish, it can be vanishingly small and therefore the numerator and the denominator in Eq.(2.19) becomes difference between two small numbers, thus leading to loss of accuracy in the estimate of the derivative. However, use of 1-D formula

$$F_{xo} = \frac{\sum \Delta x_i \Delta F_i \sum \Delta {y_i}^2}{\sum \Delta {x_i}^2 \sum \Delta {y_i}^2} = \frac{\sum \Delta x_i \Delta F_i}{\sum \Delta {x_i}^2}, \qquad F_{y_o} = \frac{\sum \Delta y_i \Delta F_i}{\sum \Delta {y_i}^2} \tag{2.23}$$

is free from above problem. This is due to the cancellation of small quantity $\sum \Delta {y_i}^2$ from the numerator as well as the denominator as shown in the derivation of 1-D formula in Eq.(2.23).Thus the same connectivity of points which was unable to give accurate value of the derivative due to the illconditioned matrix A now gives accurate value of the derivative. The 1 D finite difference formula works perfectly fine on a highly stretched cartesian mesh. So the idea is to reduce the 2 D least squares formulea to 1 D formulae by diagonalization of matrix A, then illconditioning of A does not post any problem. This effect will be numerically shown in the results of the test cases subsequently in this report.

3 LSKUM with rotation along the Eigen directions

The eigen directions offer some advantages in least squares formulation. The eigenvalues of the real symmetric matrix A in two dimensions are

$$\lambda_{1,2} = \frac{\left(\sum (\Delta x_i)^2 + \sum (\Delta y_i)^2\right) \pm \sqrt{\left(\sum (\Delta x_i)^2 - \sum (\Delta y_i)^2\right)^2 + 4(\sum \Delta x_i \Delta y_i)^2}}{2} \tag{3.24}$$

where the summation is over the full stencil. The corresponding eigenvectors of the matrix A are

$$e_1 = \begin{bmatrix} -\sum(\Delta x_i \Delta y_i) \\ \sum(\Delta x_i)^2 - \lambda_1 \end{bmatrix} \quad e_2 = \begin{bmatrix} -\sum(\Delta x_i \Delta y_i) \\ \sum(\Delta x_i)^2 - \lambda_2 \end{bmatrix} \tag{3.25}$$

The corresponding eigen-angle θ through which the (x, y) coordinate frame has to be rotated so that the new frame lies along the eigen directions, is

$$\theta = tan^{-1}\left(\frac{\sum(\Delta x_i)^2 - \lambda_1}{-\sum(\Delta x_i \Delta y_i)}\right) \tag{3.26}$$

Consider the 2-D split Euler equation

$$\frac{\partial U}{\partial t} + \frac{\partial G_x^+}{\partial x} + \frac{\partial G_x^-}{\partial x} + \frac{\partial G_y^+}{\partial y} + \frac{\partial G_y^-}{\partial y} = 0 \tag{3.27}$$

Each of the spatial derivative in Eq.(3.27) when discretized by the least squares method has its unique least squares matrix A depending upon the split stencil used to calculate the derivative. As a result, it is not possible to use the one dimensional formula simultaneously for all the spatial derivatives. This can however be achieved by use of the appropriate weights such that the x and y directions become the eigen directions along which the higher dimensional least squares formula reduce to the corresponding one dimensional formula. This will be explained later on in this report while considering the weighted least squares method. The ability of the least squares formula to permit local rotation is made use of in implementing LSKUM with rotation along the eigendirection. The local stencil of the node is first rotated along the eigen directions as shown in the Fig. 3. The upwinding is now done by splitting the stencil in this new (x', y') frame as shown in the Fig. 4 and Fig. 5. The fluxes are calculated and the conserved variables are updated in the new (x', y') frame. Finally, the conserved variables are rotated back to the global frame.

The subsonic test case of NACA 0012 aerofoil has been run at Mach number of 0.63 at the angle of attack of 2.0^o for the unrotated LSKUM and the rotated LSKUM along eigen directions given by Eq.(3.26). Fig.6 shows the comparison of residue fall for first order LSKUM. Both versions of first order LSKUM have exactly the same residue behaviour. The Cp plot and the pressure and density contours for the above test case are shown in Figs.7 to 11. Figs.12 to 17 show the corresponding comparison of the second order results. It has been observed that there is no significant difference in the results of the unrotated LSKUM and LSKUM with rotation along the eigen-direction.

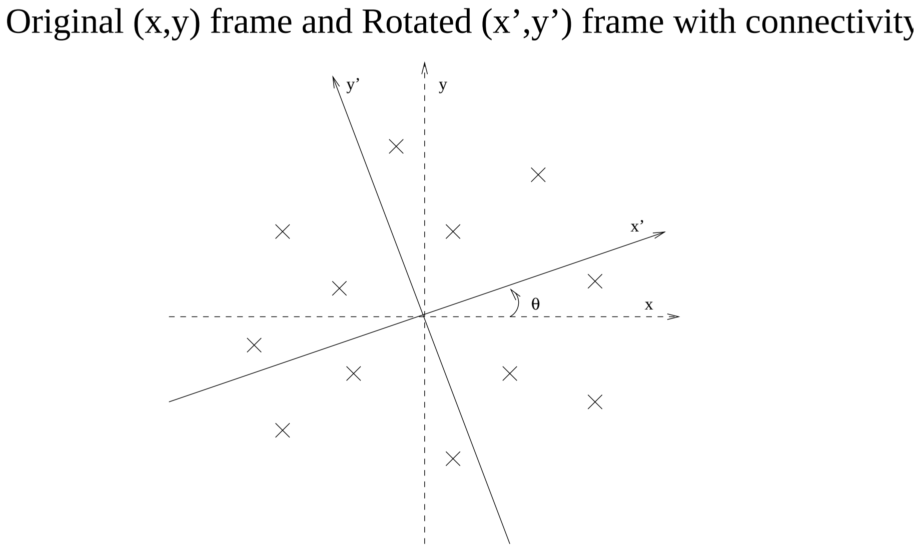


Figure 3: New rotated frame(x',y') along eigen directions

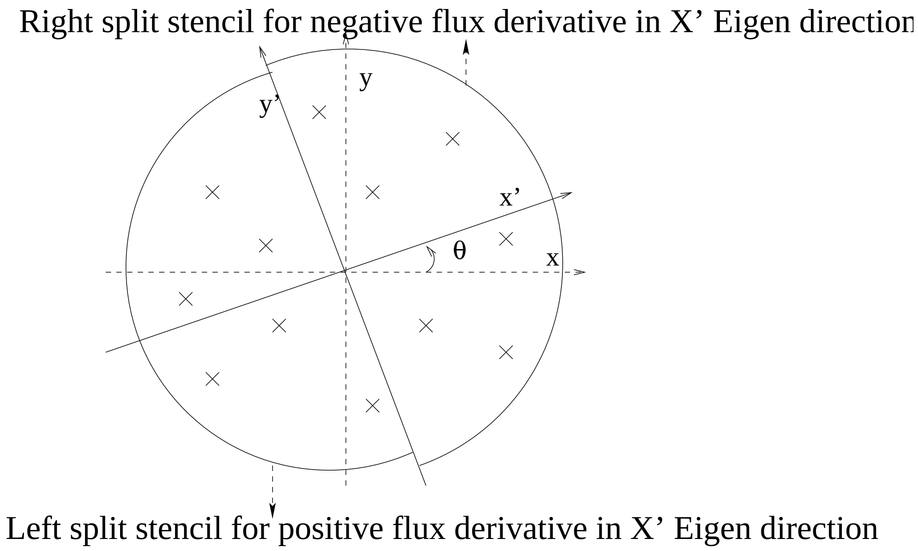


Figure 4: Split Stencil in the x Direction in the new eigen-frame

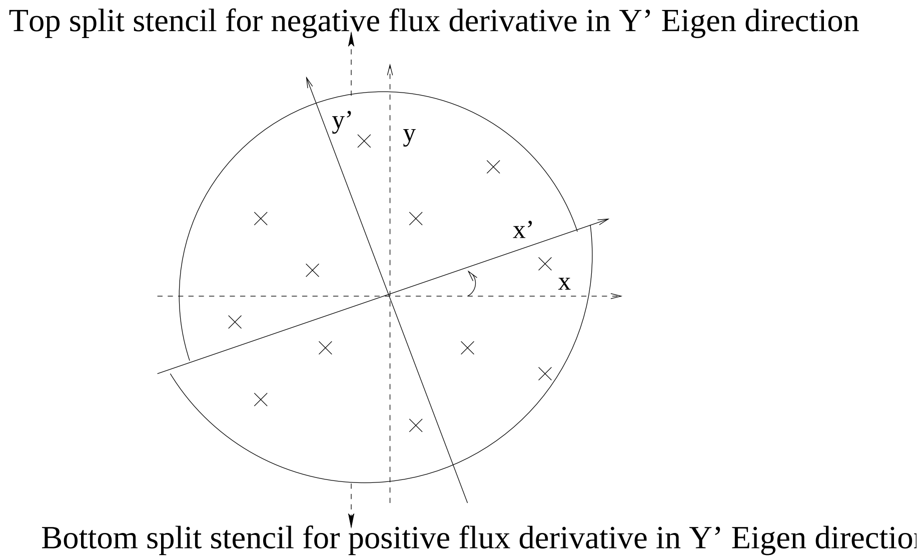


Figure 5: Split Stencil in the y Direction in the new eigen-frame

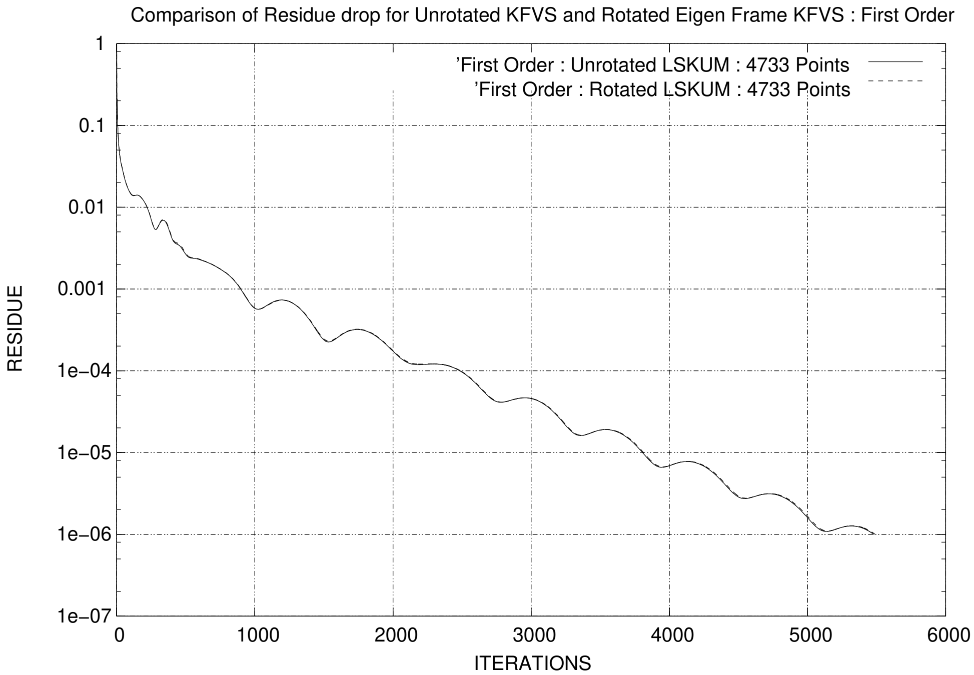


Figure 6: Comparison of Residue drop for the Rotated and Unrotated LSKUM : First Order

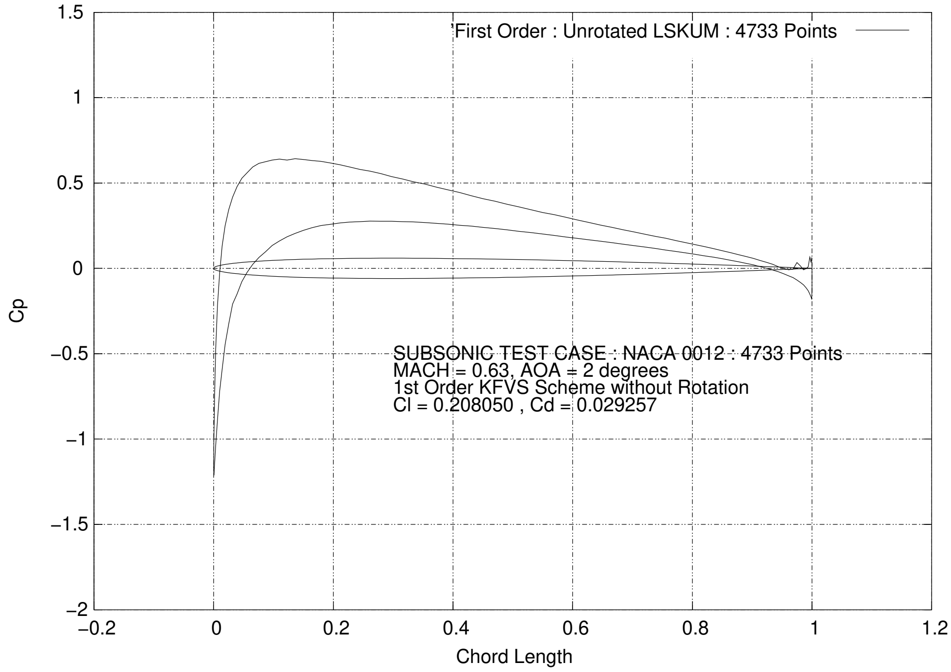


Figure 7: Cp Plot for first order LSKUM without rotation

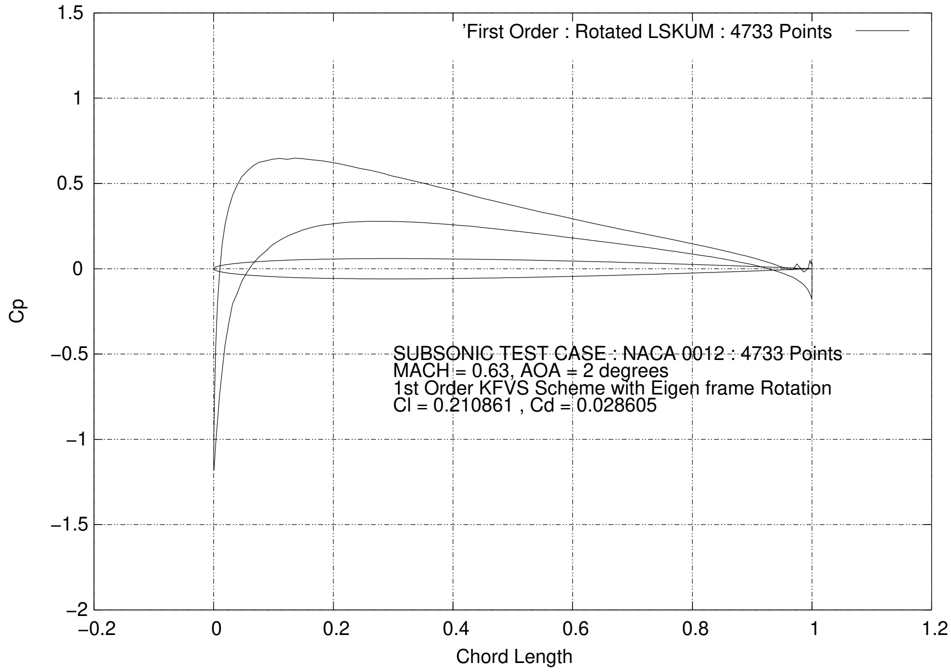


Figure 8: Cp Plot for first order LSKUM with eigen-frame rotation

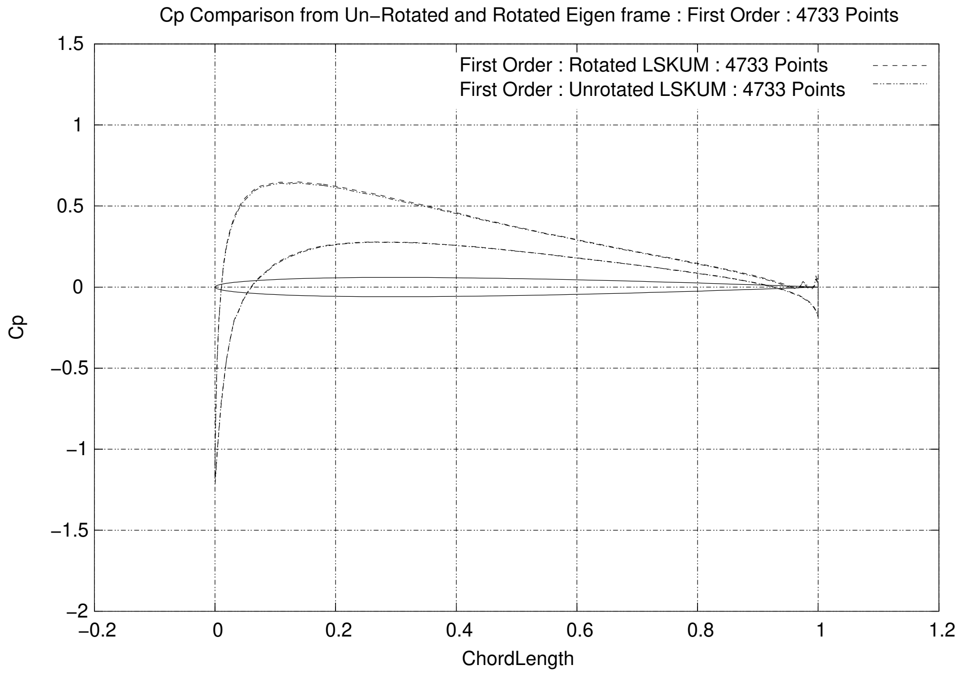


Figure 9: Comparison of Cp plot for LSKUM without and with eigen-frame rotation : First Order

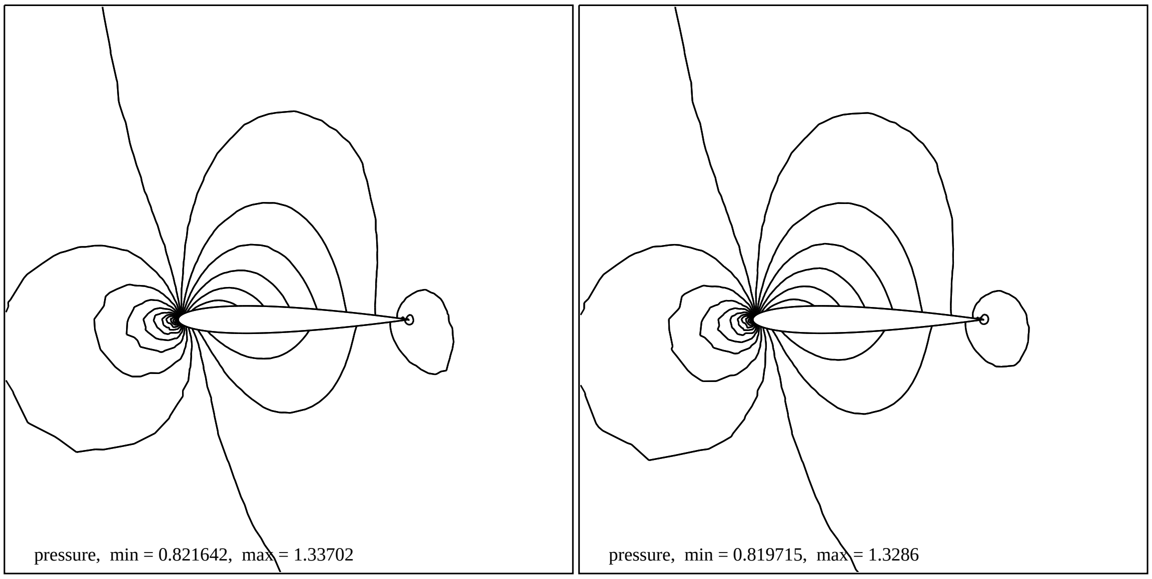


Figure 10: Presssure contours for subsonic flow using LSKUM with and without eigen-frame rotation : First Order

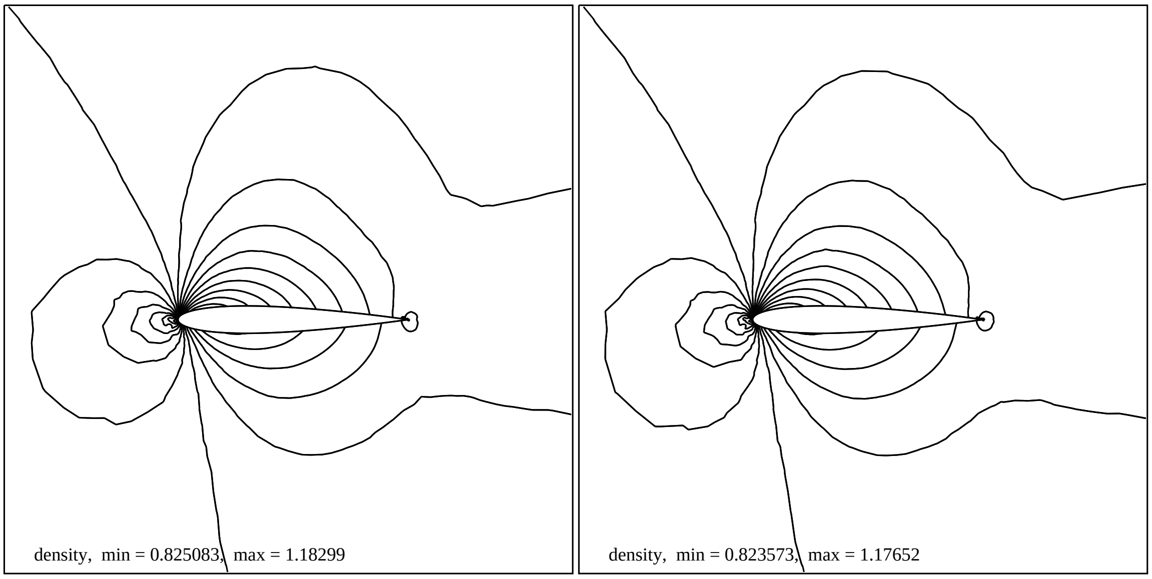


Figure 11: Density contours for subsonic flow using LSKUM with and without eigen-frame rotation : First Order

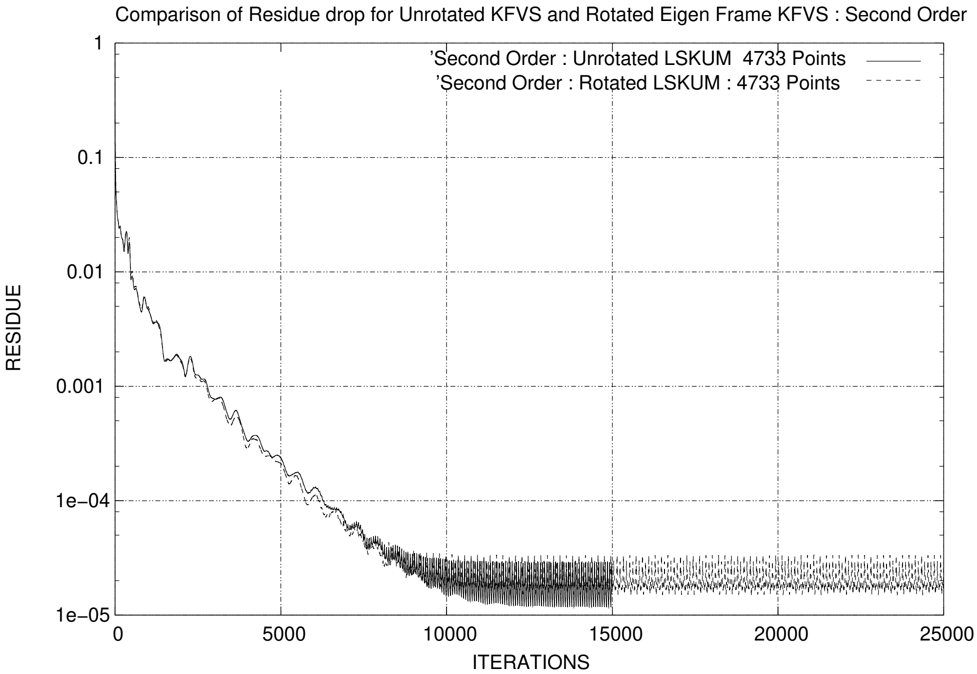


Figure 12: Comparison of Residue drop for the Rotated and Unrotated LSKUM : Second Order

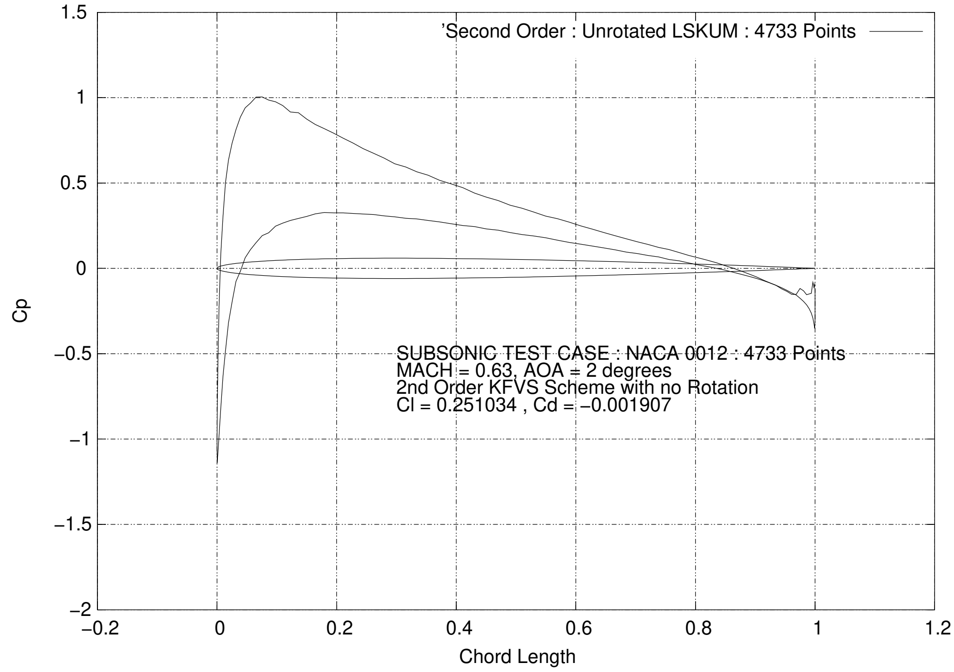


Figure 13: Cp Plot for second order LSKUM without rotation

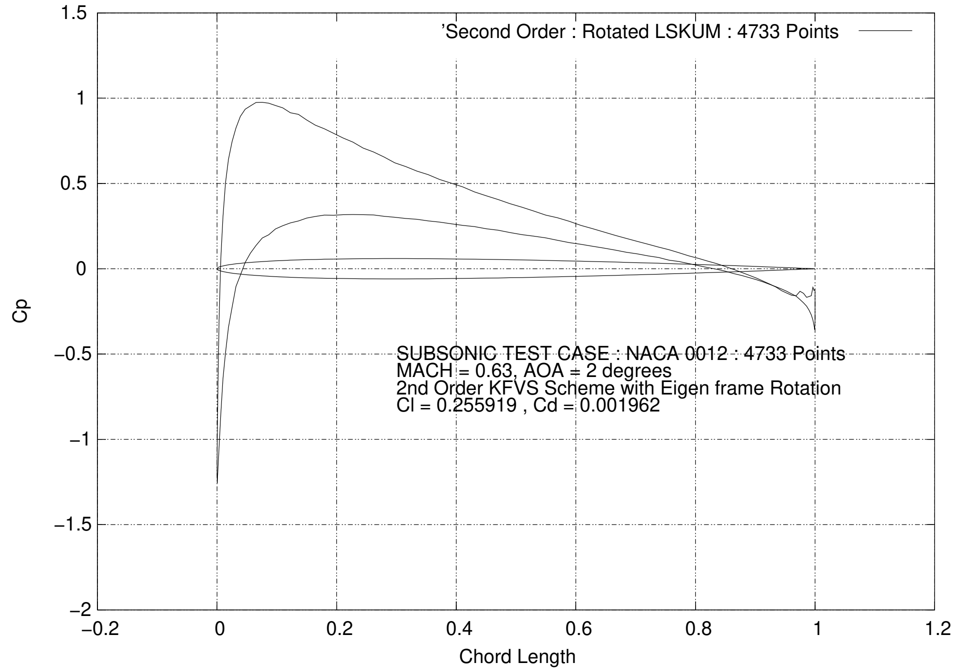


Figure 14: Cp Plot for second order LSKUM with eigen-frame rotation

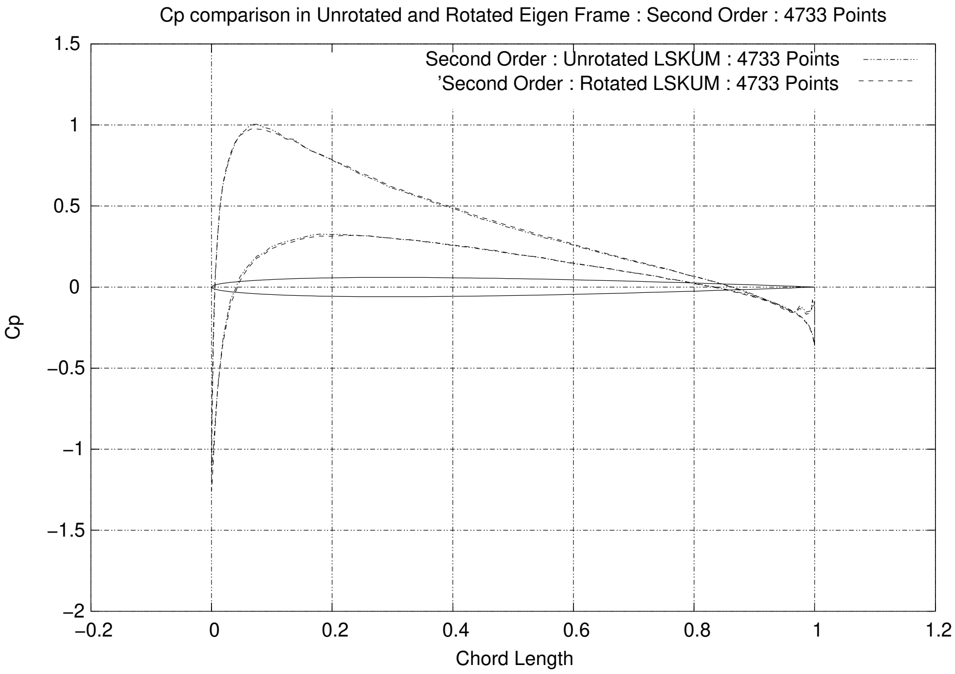


Figure 15: Comparison of Cp plot for LSKUM without and with eigen-frame rotation : Second Order

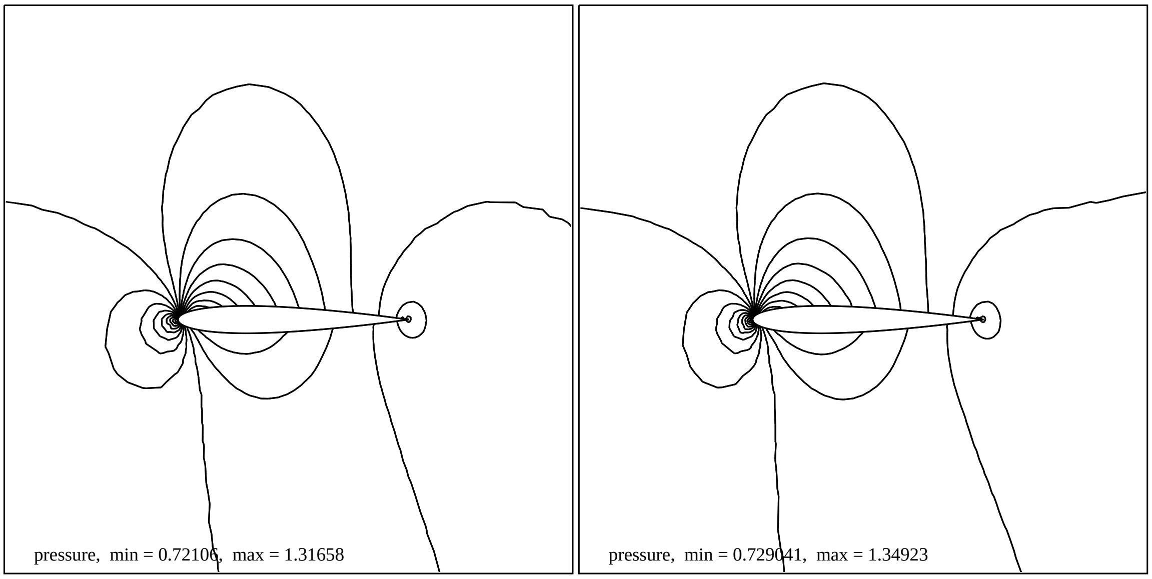


Figure 16: Presssure contours for subsonic flow using LSKUM with and without eigen-frame rotation : Second Order

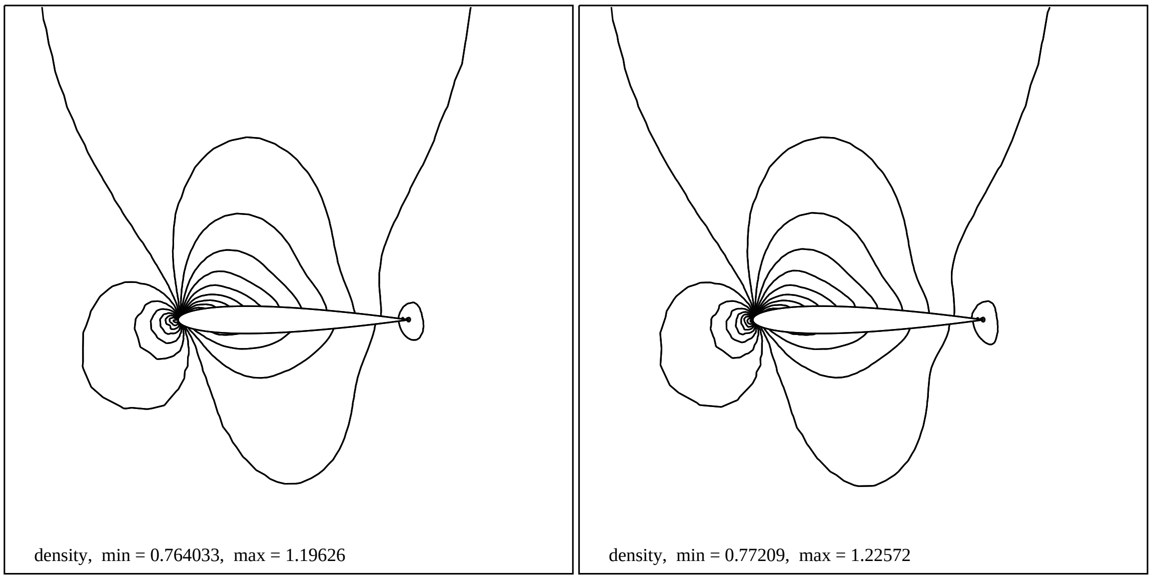


Figure 17: Density contours for subsonic flow using LSKUM with and without eigen-frame rotation : Second Order

4 Weighted Least Squares Method in 2-D

The weighted and the unweighted least squares formula for the derivatives are derived in an exactly similar manner. Here, first the unweighted least squares method for the calculation of the derivatives in 2-D will be discussed. This will be followed by the description of the weighted least squares method. The calculation of derivatives using the least squares method involves minimization of the sum of the squares of error. Consider Fig.18. To get the derivative of the function $F(x, y)$ at the point P_o shown in Fig. 18, we expand F_i around point P_o in terms of Taylors series :

$$F_i = F_o + (x_i - x_o)F_{xo} + (y_i - y_o)F_{y_o} + O\left(\Delta x^2, \Delta y^2\right) \tag{4.28}$$

Now as before, define

$$\Delta x_i = x_i - x_o, \quad \Delta y_i = y_i - y_o, \quad \Delta F_i = F_i - F_o$$

and the sum of the squares of errors at point P_o (after truncating Taylors Series in Eq.(4.28)).is given by

$$E = \sum_{i=1}^{p} \left(\Delta F_i - \Delta x_i F_{xo} - \Delta y_i F_{y_o}\right)^2 \tag{4.29}$$

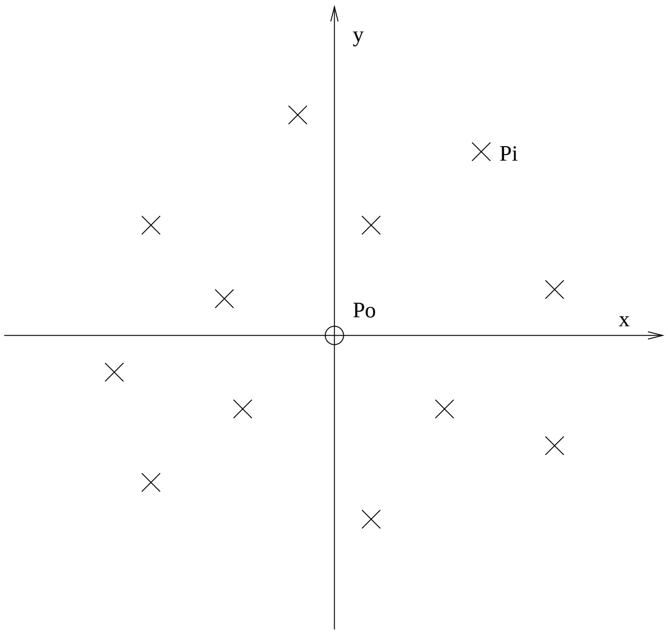


Figure 18: Connectivity for node o in 2 dimension

Minimizing E in Eq.(4.29) with respect to F_{xo} and F_{y_o}, we get the following system of equations to be solved

$$A\,(grad\;F)_o^T = b \tag{4.30}$$

where

$$A = \begin{bmatrix} \sum \Delta x_i{}^2 & \sum \Delta x_i \Delta y_i \\ \sum \Delta x_i \Delta y_i & \sum \Delta y_i{}^2 \end{bmatrix}, (grad\;F)_o^T = \begin{bmatrix} F_x \\ F_y \end{bmatrix}, b = \begin{bmatrix} \sum \Delta x_i \Delta F_i \\ \sum \Delta y_i \Delta F_i \end{bmatrix}$$

Similarly, the process of minimization of the weighted sum of the squares of the error leads us to the weighted least squares(wls) method.

$$E = \sum_{i=1}^{p} w_i \big(\Delta F_i - \Delta x_i F_{xo} - \Delta y_i F_{y_o}\big)^2 \tag{4.31}$$

where w_i is the weight assigned to each node, E is the weighted sum of the squares of error. It is desirable to have positive weights so as to retain the LED property of the least squares formulae (as will be explained later on in this report). Minimizing the weighted sum of the squares of the error given in Eq.(4.31) with respect to F_{xo} and F_{y_o} as before, we get the following system of equations to be solved

$$A(w)\,(grad\;F)_o^T = b\,(w) \tag{4.32}$$

where

$$A(w) = \begin{bmatrix} \sum w_i (\Delta x_i)^2 & \sum w_i (\Delta x_i \Delta y_i) \\ \sum w_i (\Delta x_i \Delta y_i) & \sum w_i (\Delta y_i)^2 \end{bmatrix},$$

$$(grad\ F)_o^T = \begin{bmatrix} F_x \\ F_y \end{bmatrix}, b(w) = \begin{bmatrix} \sum w_i (\Delta x_i \Delta F_i) \\ \sum w_i (\Delta y_i \Delta F_i) \end{bmatrix} \tag{4.33}$$

The weights used in the least squares formulae can serve various purposes:

(i) The weights can be used to increase the accuracy of the least squares formulae [2].

(ii) The weights can be selected so as to increase the spectral resolution of the least squares formulae [9].

(iii) The weights can help to maintain positivity and local extremum diminishing (LED) property of the least squares formulae.

(iv) The weights can help in improving the convergence characteristics of the grid-free solver which uses the least squares formulae.

It is interesting to note that an appropriate choice of the weights can even change the nature of the matrix A. So an interesting question naturally arises : Whether suitable weights can be chosen which favourably change the condition number of the least squares matrix such that the solution accuracy and robustness is improved ? The answer is affirmative. We will show later that the weights can be suitably determined such that the weighted least squares matrix $A(w)$ is diagonal. It has been observed earlier that the diagonalization of the least squares matrix reduces the two and three dimensional least squares formulae of the derivatives to the corresponding one dimensional least squares formulae in the appropriate direction. It is expected that $A(w)$ with suitable weights will help in overcoming the problems of bad connectivity to a great extent.

4.1 Calculation of weights for two dimensional least squares formulae

Consider Fig.19 which shows the four quadrants of the split stencil normally used for upwinding. It is observed that the product of Δx and Δy is always positive in quadrants I and III, while it is always negative in quadrants II and IV. Whenever we are using x-y splitting, each split stencil involves two quadrants. One of the quadrants always contributes to the positive product $\Delta x \Delta y$ while the other quadrant always contributes to the negative product $\Delta x \Delta y$. Suppose we want to find the weights for the least squares

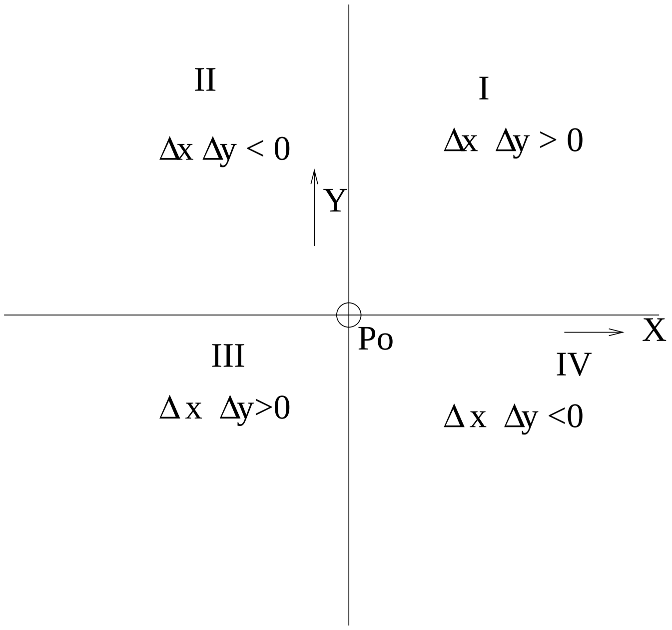


Figure 19: Quadrants for the split stencil of a node

formula when we are using the point distribution in the left stencil only. The left stencil comprises of quadrants II and III. Making use of the above observation, we can easily obtain the weights for the points in the left stencil such that $\sum_{II+III} w_i \left(\Delta x_i \Delta y_i\right) = 0$ while ensuring that the weights calculated are always positive. It mus be kept in mind that a primary requirement of the connectivity of a node is that none of the quadrants shown in Fig.19 should be empty. Let w_{II} be the weight assigned to the points lying in the quadrant II of the stencil while w_{III} be the weight assigned to the points lying in the quadrant III of the stencil. We then enforce

$$w_{III}\Big(\sum \Delta x_i \Delta y_i\Big)_{III} + w_{II}\Big(\sum \Delta x_i \Delta y_i\Big)_{II} = 0 \tag{4.34}$$

Introducing the notation for cross products,

$$C_{xy}^{III} = \textstyle\sum_{III} \Delta x_i \Delta y_i, \qquad C_{xy}^{II} = \textstyle\sum_{II} \Delta x_i \Delta y_i$$

In the notation, the superscripts II and III on C denote the quadrants over which the summation is taken and subscript xy to C stands for the fact the cross products $\Delta x_i \Delta y_i$ in the $x - y$ plane. The equation 4.34 can now be re-written as

$$w_{III} C_{xy}^{III} + w_{II} C_{xy}^{II} = 0$$

As per the observations made before :

$$C_{xy}^{II} < 0, \qquad C_{xy}^{III} > 0$$

In terms of the above quadrantwise cross products, we get

$$\frac{w_{III}}{w_{II}} = -\frac{C_{xy}^{II}}{C_{xy}^{III}} > 0 \tag{4.35}$$

From the Eq.(4.35) above, it is seen that the ratio of the weights obtained above is always positive as the product $\Delta x \Delta y$ is positive in quadrant III while it is negative in quadrant II. The ratio goes to infinity when the denominator in the Eq.(4.35) goes to zero. This can only happen when the quadrant III is empty, which goes against our requirement that the region contributing to the derivative at a node must have atleast a point. Thus, the ratio in Eq.(4.35) can never go to infinity. The similar procedure can be applied to find the weights for the derivatives using the points in any of the other split stencil : right, top or bottom. In each of these cases, it is observed that the two dimensional least squares formula reduces to the corresponding one dimensional formula in the respective directions. The fact that by the use of appropriate weights, the least squares formula in two dimensions reduces to the corresponding one dimensional formula is quite significant. The matrix A for the two and higher dimensional least squares formula can become bad in a wide variety of ways [7]. As stated earlier, the matrix A being a purely geometric matrix depends solely on the connectivity of the node under consideration. The bad connectivity may result in loss of accuracy of the computations, or even lead to the code divergence [7]. Thus, great care has to be taken in the pre-processor stage itself to avoid the generation of the bad connectivity. There are various examples of bad connectivity [7] and some of these are

(i) The case when all the points in the connectivity lie in a small band passing (degenerate case) through the node itself is one example of a bad connectivity. In this case, the matrix A becomes nearly singular and hence leads to inaccuracy in the solution of $A\,(grad\,F)_o^T = b$. This type of connectivity is shown in Fig 20

(ii) Highly anisotropic connectivity is also another example of bad connectivity. This type of connectivity occurs when the points are very close to each other in a particular direction but in the other direction, they are far apart from each other. Such type of connectivity generally occurs near the wing root and fuselage junction in three

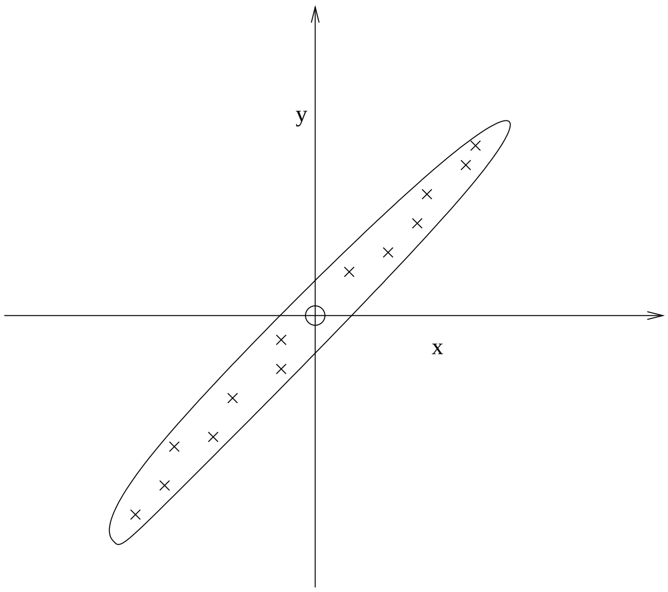


Figure 20: Bad connectivity : Connectivity points lying in a small band passing through the node

dimensional problems. Another example where such type of connectivity occurs is in the boundary layer regions. This is because the gradient varies very rapidly in one direction as compared to the gradient in another direction. In boundary layer regions, it becomes necessary to have more points in the normal direction. However, this type of connectivity makes the matrix highly illconditioned leading to the loss in accuracy of the computations or even code divergence. This type of connectivity is shown in Fig 21. As mentioned before, when the connectivity nearly collapses to a clustered cartesian mesh, the condition number of A becomes large. Use of 1-D formulae along coordinate directions still gives accurate estimate of derivatives.

(iii) Sometimes the neighbours of a node are selected such that one of the quadrants around the node is empty. This type of connectivity is also another example of a bad connectivity, which leads to the code divergence. This type of connectivity is shown in Fig 22

Contrary to the usual two and higher dimensional least squares formulae for the derivatives, the one dimensional least squares formula for the derivative can never fail as long as the quadrants are not empty. Use of appropriate weights help in mitigating the effect

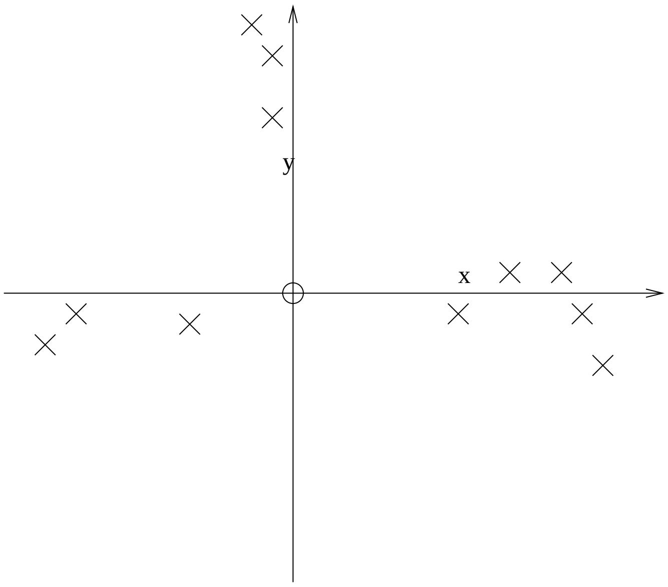


Figure 21: Highly anisotropic connectivity

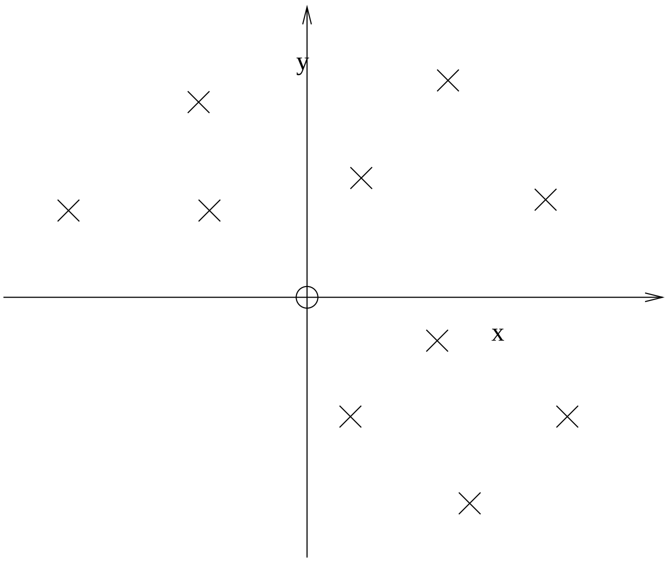


Figure 22: Connectivity with neighbour points absent in a quadrant

of bad connectivity thus preserving the accuracy of the least squares formulae. Hence it is claimed that if the weights are chosen such that the two dimensional least squares formulae reduce to the one dimensional formulae, the problem of code divergence due to bad connectivity can be effectively tackled. Thus the definition of bad connectivity to some extent depends on which formula for the derivative we are using.

Results of test case of subsonic flow past NACA 0012 aerofoil at Mach number 0.63 and angle of attack of 2^o have been shown to support the claim. Fig.23 shows the comparison of the residue drop for the first order and second order accurate results of weighted and unweighted LSKUM on a computational domain with 4733 nodes. The weights have been calculated such that the two dimensional formulae reduce to the one dimensional formulae in the corresponding coordinate directions. It is observed that the residue drops smoothly to a lower value even for the second order accurate computations done using the weighted LSKUM while it becomes saturated to a relatively higher value for the unweighted LSKUM. Similar observation are valid when the same test case is run on a finer grid of 12388 nodes as shown in the Fig.24.

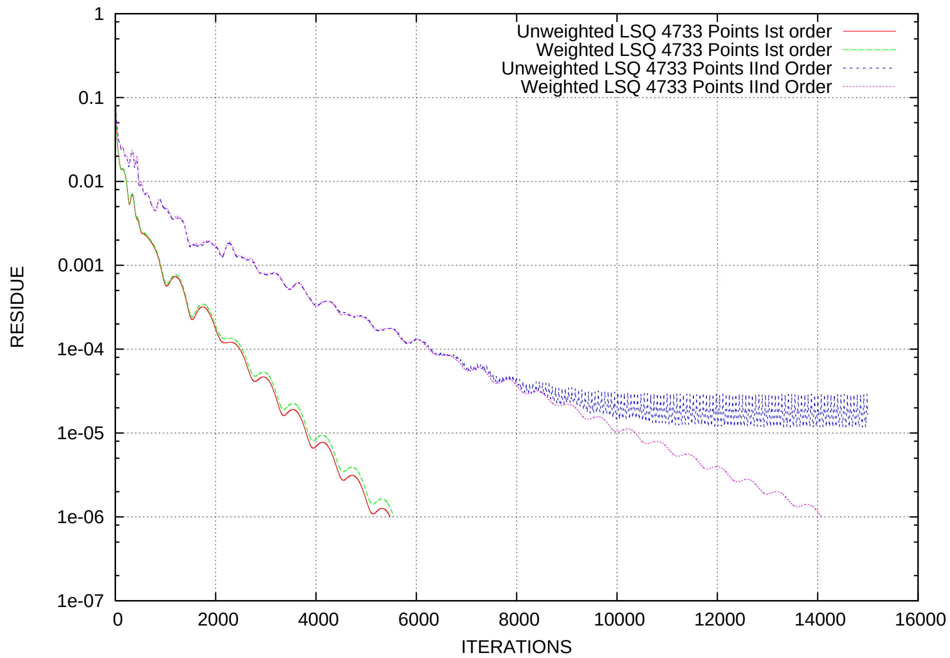


Figure 23: Residue drop for weighted and unweighted Least Squares Method : 4733 Points

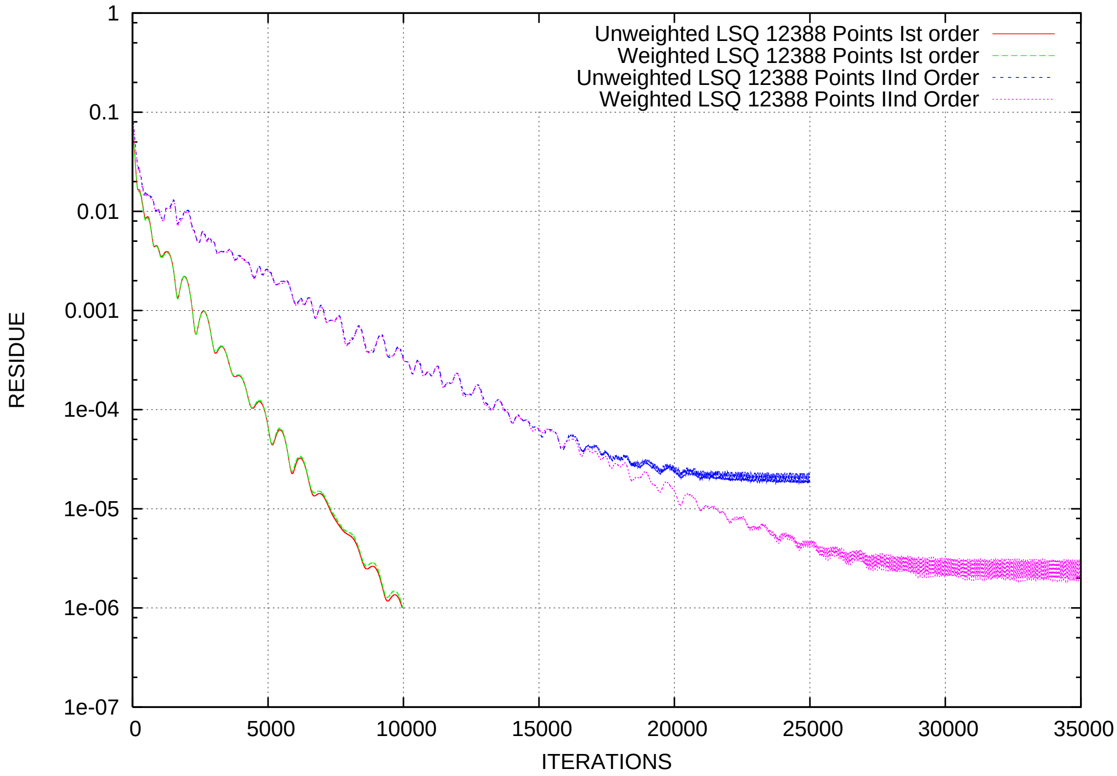


Figure 24: Residue drop for weighted and unweighted Least Squares Method : 12388 Points

In support of the claim made above about code divergence, a point distribution generated by Delaunay triangulation as shown in Fig.25 has been used. The triangulation near the aerofoil has been intentionally tampered so as to make it an extreme case of a bad connectivity. The initial connectivity of the node is shown in Fig.26. The node under consideration is indicated by a circle in Fig.26. After tampering with triangulation, the good connectivity has been converted into an extremely bad connectivity as shown in Fig.27. Then, the tampered point distribution (with bad connectivity) was used as an input for the code using weighted LSKUM and unweighted LSKUM. As was expected, the unweighted LSKUM code was unable to run on this extreme case of bad connectivity as is evident from the Fig.28. However, the weighted LSKUM code did not encounter any problems and was successful in generating results for the subsonic test case as is evident from the residue plot shown in Fig.29. The pressure and density contours are shown in Figs 30 and 31.

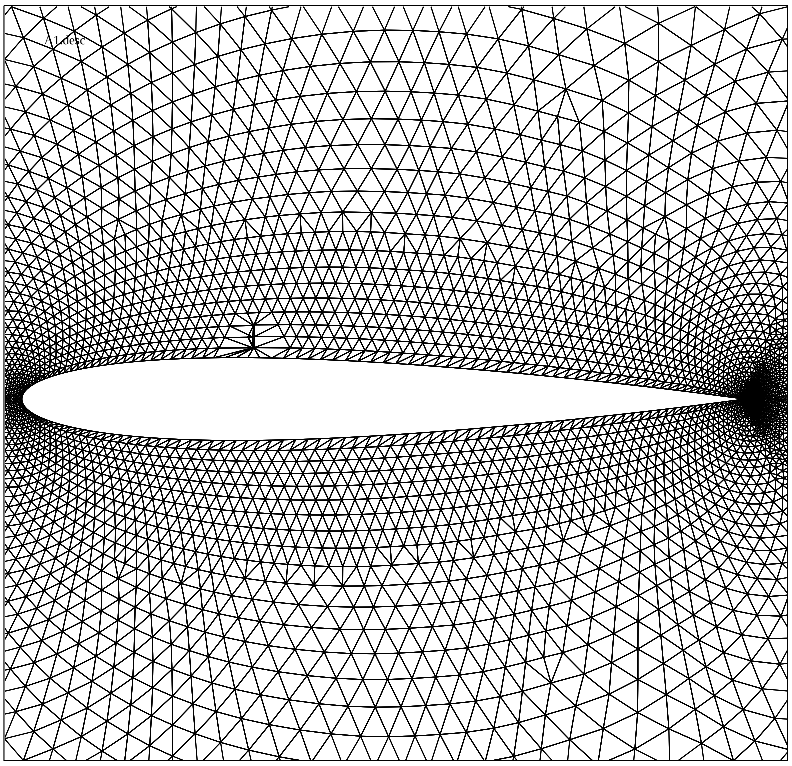


Figure 25: Bad connectivity on aerofoil : 7269 points in domain

5 Weighted Least Squares compared with Finite Difference Method

Finite difference method of discretization is generally used on uniform cartesian grids. We observe that if we apply the least squares formulae to a node having symmetric connectivity consisting of points lying on a uniform cartesian grid, the two dimensional least squares formulae reduces to the standard finite difference formulae along individual coordinate directions. The eigenvectors of unweighted least squares matrix A for such connectivity are parallel to x and y axes. Further matrix A is diagonal. Such a nice property is lost when connectivity is not regular. We have shown that for arbitrary connectivity (ie. points distributed in an irregular fashion) the eigenvectors of A are not along the coordinate axes. The weighted least squares matrix $A(w)$ however with a suitable choice of weights overcomes this problem by forcing eigenvectors of $A(w)$ to be parallel to the coordinate axes. Thus the weighted least squares is a kind of generalization

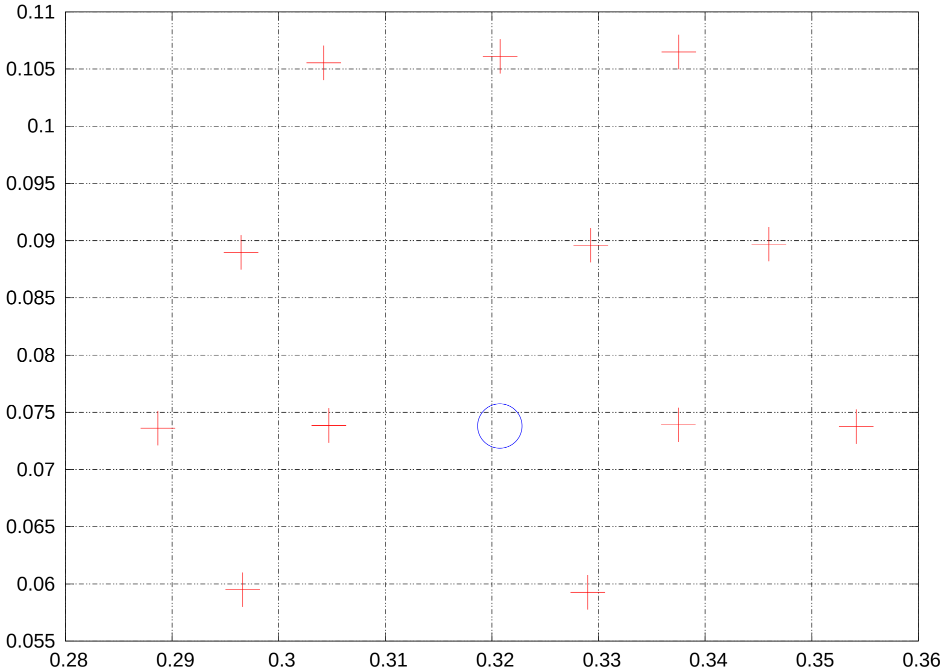


Figure 26: Original Good Connectivity

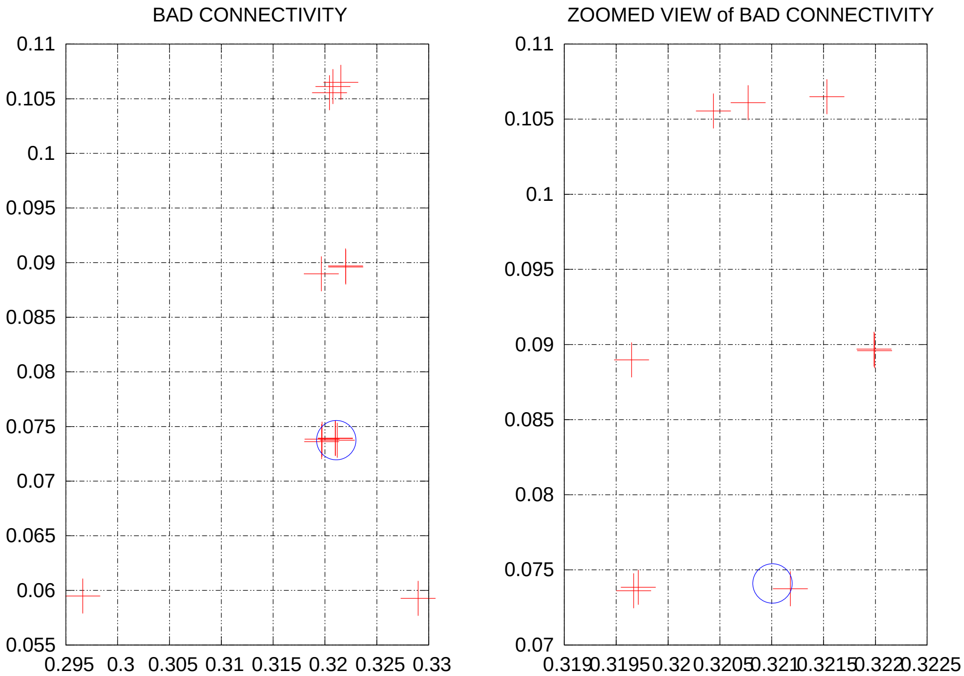


Figure 27: Bad Connectivity

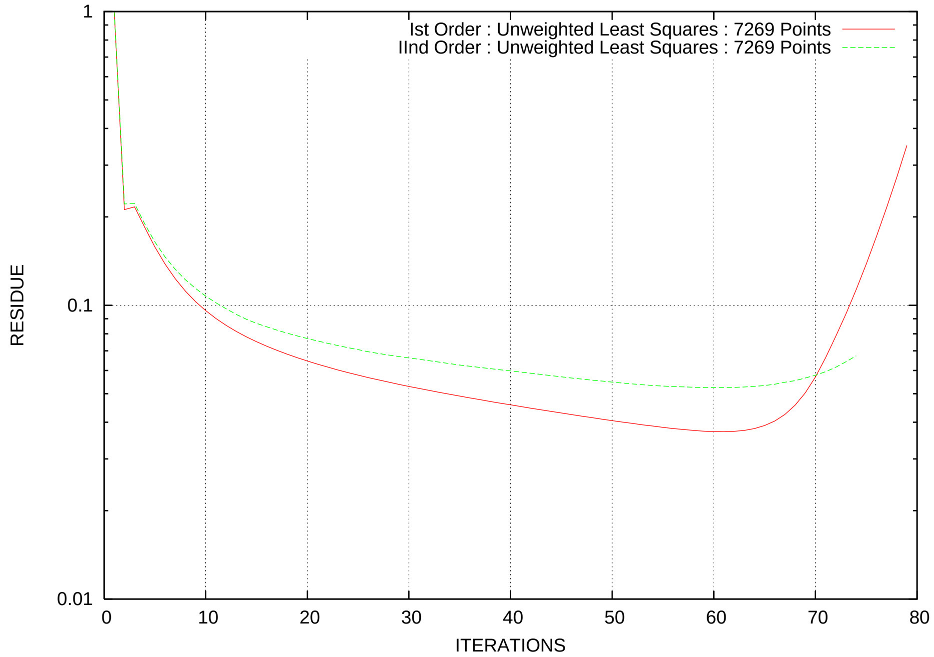


Figure 28: Residue drop for unweighted least squares using bad connectivity : First Order and Second Order

of FD approach, the generalization allows use of 1 D finite difference like formulae on an arbitrary connectivity.

6 Weighted Least Squares and the Local Extremum Diminishing (LED) Property

A scheme is said to satisfy local extremum diminishing (LED) property if as a consequence of update, maxima do not increase and the minima do not decrease. The semi-discrete form of the conservation law in general is

$$\frac{dF_o}{dt} = \sum_{i \neq o} c_i \left(F_i - F_o\right) \tag{6.36}$$

The scheme given by Eq.(6.36) is LED if it satisfies the following constraint

$$c_i \geq 0 \tag{6.37}$$

Some important characteristics of LED schemes are

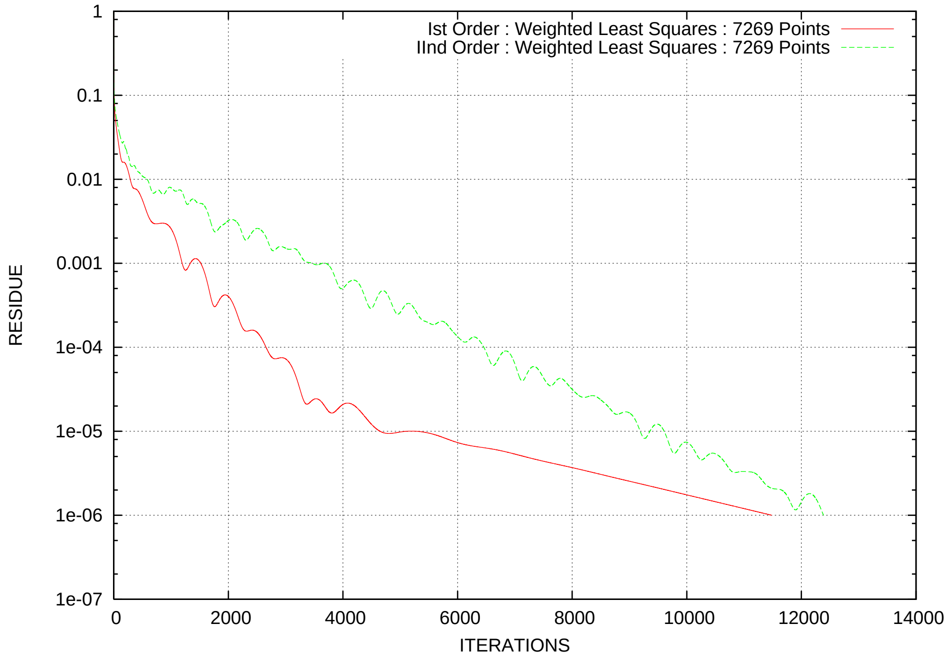


Figure 29: Residue drop for weighted least squares using bad connectivity : First Order and Second Order

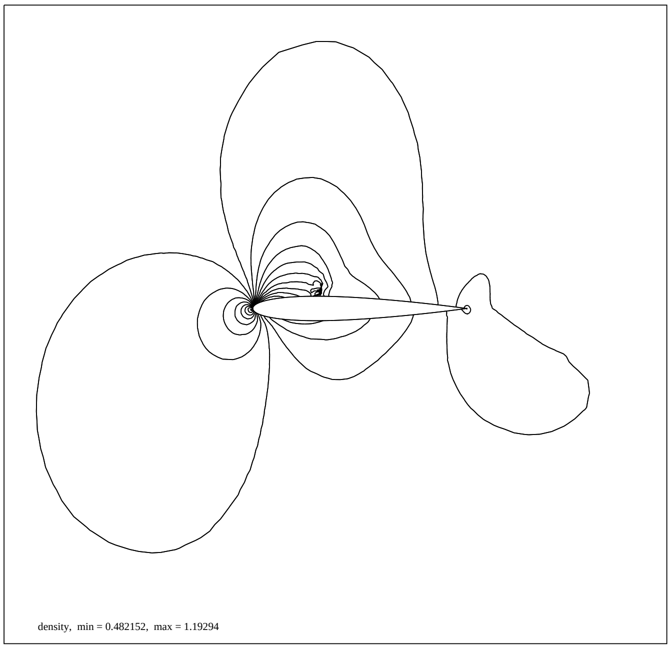


Figure 30: Density Contours : Weighted Least Squares : Second Order : Bad connectivity

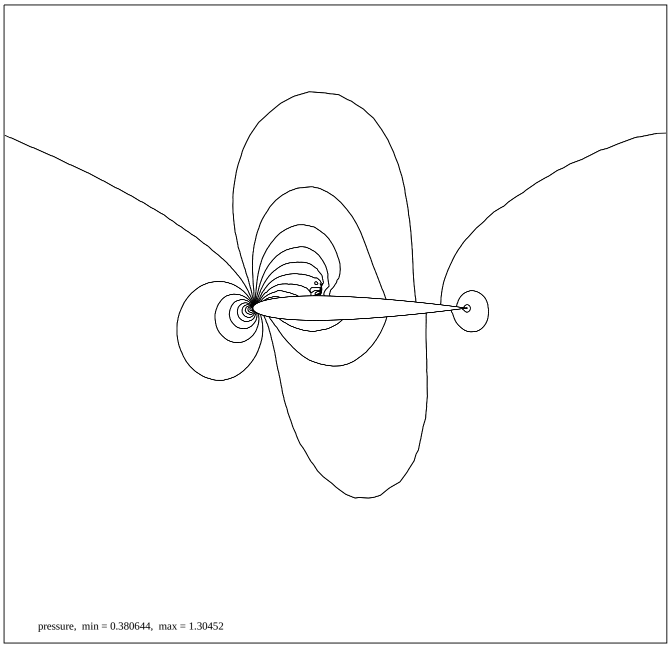


Figure 31: Pressure Contours : Weighted Least Squares : Second Order : Bad connectivity

1 Positivity condition mentioned in Eq.(6.37) ensures that no oscillations arise in the numerical solution. LED scheme leads to diagonally dominant matrices [4] which ensures good convergence properties for implicit formulations.

2 Positivity condition mentioned in Eq.(6.37) along with CFL condition is sufficient to ensure stability in the L_∞ norm[4]. Thus they provide a stringent condition of stability for the scheme.

Consider the Boltzmann equation without collision term

$$\frac{\partial F}{\partial t} + v\frac{\partial F}{\partial x} = 0 \tag{6.38}$$

The CIR splitting for the Boltzmann equation (6.38) leads to

$$\frac{\partial F}{\partial t} + v^+\frac{\partial F}{\partial x} + v^-\frac{\partial F}{\partial x} = 0 \tag{6.39}$$

where

$$v^+ = \frac{v+|v|}{2}, \qquad v^- = \frac{v-|v|}{2}$$

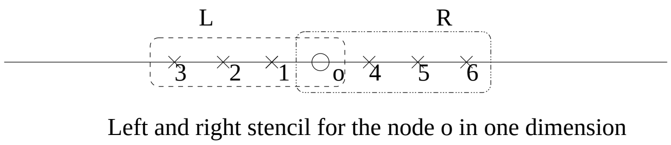


Figure 32: Left and right stencil for the node o in one dimension

Now discretizing the spatial derivatives in Eq.(6.39) using the weighted least squares method, we obtain the semidiscrete Boltzmann equation

$$\frac{dF_o}{dt} = -\left(v^+ \frac{\sum_i w_{Li} \Delta x_i \Delta F_i}{\sum_i w_L i \Delta x_i^{\,2}}\right)_L - \left(v^- \frac{\sum_i w_{Ri} \Delta x_i \Delta F_i}{\sum_i w_{Ri} \Delta x_i^{\,2}}\right)_R$$

where L refers to the sub stencil comprising of the points to the left of the point P_o in Fig.32, w_L refers to the weight assigned to the nodes in the left split sub stencil, R refers to the sub stencil comprising of the points to the right of the point P_o in Fig.32 and w_R refers to the weight assigned to the nodes in the right split sub stencil. The semi-discrete Boltzmann equation in 1-D can be further re-written as

$$\frac{dF_o}{dt} = \sum_{i \neq o, i \in L} c_{iL} \left(F_i - F_o\right) + \sum_{i \neq o, i \in R} c_{iR} \left(F_i - F_o\right) \tag{6.40}$$

where the derivative is calculated at the node P_o and the summation is over all the neighbours i in the appropriate stencil. Comparing Eqs.(6.36)and (6.40), we get the following expansions for the coefficients :

$$c_{iL} = \frac{-v^+ w_{Li} \Delta x_i}{\sum_{j \in L} w_{Lj} \Delta x_j^{\,2}}$$

$$c_{iR} = \frac{-v^- w_{Ri} \Delta x_i}{\sum_{j \in R} w_{Rj} \Delta x_j^{\,2}}$$

Since $v^+ > 0$, $v^- < 0$ and $\Delta x_i < 0$ if $i \in L$ and $\Delta x_i > 0$ if $i \in R$, the coefficients c_{iR} and c_{iL} are non-negative. Thus we see that in 1-D LSKUM is naturally LED due to its upwind character. The weights in LS must evidently be positive to ensure the LED property of LSKUM. In general this property does not carry over to 2-D LSKUM except in special circumstances like cartesian point distributions and one sided FD formulae for spatial derivatives. If the weights are determined to satisfy $\sum \Delta x_i \Delta y_i = 0$ as described previously, then the LED property is recovered because all the formulae for the derivatives reduce to 1-D form. Consider the Boltzmann equation without collision term in two dimensions

$$\frac{\partial F}{\partial t} + v_1 \frac{\partial F}{\partial x} + v_2 \frac{\partial F}{\partial y} = 0 \tag{6.41}$$

The CIR splitting for the Boltzmann equation (6.41) leads to

$$\frac{\partial F_o}{\partial t} + v_1^+ \frac{\partial F}{\partial x} + v_1^- \frac{\partial F}{\partial x} + v_2^+ \frac{\partial F}{\partial y} + v_2^- \frac{\partial F}{\partial y} = 0 \tag{6.42}$$

where

$$v_1^+ = \frac{v_1 + |v_1|}{2}, \quad v_1^- = \frac{v_1 - |v_1|}{2}, \quad v_2^+ = \frac{v_2 + |v_2|}{2}, \quad v_2^- = \frac{v_2 - |v_2|}{2}$$

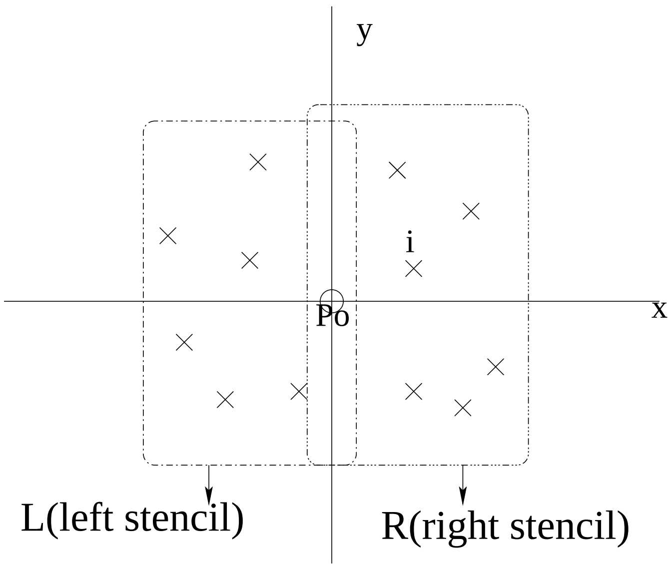


Figure 33: Left and right sub stencils for the node P_o under consideration

Now discretizing the spatial derivatives in Eq.(6.42) using weighted least squares where the weights satisfy the condition $\sum \Delta x_i \Delta y_i = 0$, we obtain the semi-discrete Boltzmann equation in 2-D as

$$\frac{dF_o}{dt} = -\left(v_1^+ \frac{\sum_i w_{Li} \Delta x_i \Delta F_i}{\sum_i w_{Li} \Delta {x_i}^2} \right)_L - \left(v_1^- \frac{\sum_i w_{Ri} \Delta x_i \Delta F_i}{\sum_i w_{Ri} \Delta {x_i}^2} \right)_R - \left(v_2^+ \frac{\sum_i w_{Bi} \Delta y_i \Delta F_i}{\sum_i w_{Bi} \Delta {y_i}^2} \right)_B - \left(v_2^- \frac{\sum_i w_{Ti} \Delta y_i \Delta F_i}{\sum_i w_{Ti} \Delta {y_i}^2} \right)_T$$

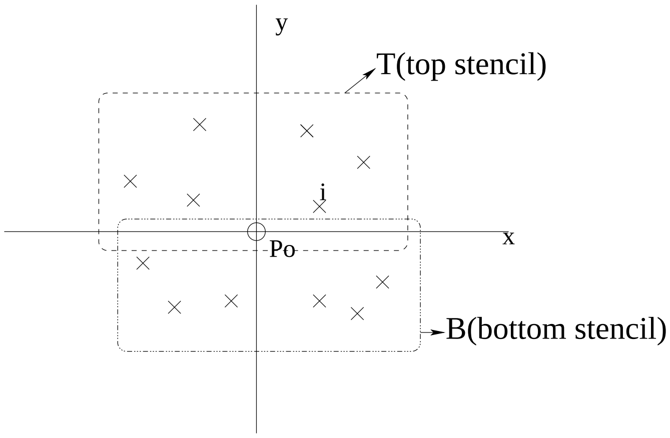


Figure 34: Top and bottom sub stencils for the node P_o under consideration

where L refers to the sub stencil comprising of the nodes to the left of the node P_o in Fig.33, w_L refers to the weight assigned to the nodes in the left split sub stencil, R refers to the sub stencil comprising of the nodes to the right of the node P_o in Fig.33 and w_R refers to the weight assigned to the nodes in the right split sub stencil, T refers to the sub stencil comprising of the nodes located above the node P_o in Fig.34, w_T refers to the weight assigned to the nodes in the top split sub stencil, B refers to the sub stencil comprising of the nodes below the node P_o in Fig.34 and w_B refers to the weight assigned to the nodes lying in the bottom split sub stencil.

The semi-discrete Boltzmznn equation in 2-D can be further re-written as

$$\frac{dF_o}{dt} = \sum_{i\neq o, i\in L} c_{iL}\left(F_i - F_o\right) + \sum_{i\neq o, i\in R} c_{iR}\left(F_i - F_o\right) + \sum_{i\neq o, i\in B} c_{iB}\left(F_i - F_o\right) + \sum_{i\neq o, i\in T} c_{iT}\left(F_i - F_o\right) \tag{6.43}$$

Comparing Eqs.(6.36)and (6.43), we get the following expansions for the coefficients :

$$\begin{aligned} c_{iL} &= \frac{-{\upsilon_1}^{+} w_{Li} \Delta x_i}{\sum_{j\in L} w_{Lj} {\Delta x_j}^2} \\ c_{iR} &= \frac{-{\upsilon_1}^{-} w_{Ri} \Delta x_i}{\sum_{j\in R} w_{Rj} {\Delta x_j}^2} \\ c_{iB} &= \frac{-{\upsilon_2}^{+} w_{Bi} \Delta x_i}{\sum_{j\in B} w_{Bj} {\Delta x_j}^2} \\ c_{iT} &= \frac{-{\upsilon_2}^{-} w_{Ti} \Delta x_i}{\sum_{j\in T} w_{Tj} {\Delta x_j}^2} \end{aligned} \tag{6.44}$$

Now for left sub stencil (L) as shown in the Fig.33, $\Delta x_i < 0$ and if $v_1 > 0$, then the coefficient c_{iL} above is positive provided w_{Li} is positive. Similarly it can be shown that other coefficients c_{iR}, c_{iT} and c_{iB} are positive provided corresponding weights in Eq.(6.44) are positive. Thus we have seen that by reducing the multi dimensional least squares formulae for the derivatives to one dimensional formulae by the appropriate choice of weights, we not only can prove the LED property of the least squares formulae but can also show the connection between LED and positivity of weights.

7 Weighted Least Squares Method in 3-D

The weighted least squares formulae for the derivatives in 3-D are derived in the same way as described before, that is, by minimizing the sum of the squares of the error at the node. In case of 3-D weighted least squares method, the weighted sum of the squares of the error to be minimized is defined by

$$E = \sum_{i=1}^{p} w_i \big(\Delta F_i - \Delta x_i F_{xo} - \Delta y_i F_{y_o} - \Delta z_i F_{zo}\big)^2 \tag{7.45}$$

where w_i is the weight assigned to each node and it is desirable to have $w_i > 0$. Minimizing the weighted sum of the squares of the error with respect to F_{xo}, $F_{y_o}$and F_{zo} , we get the following system of linear algebraic equations

$$A(w)\,(grad\;F)_o^T = b\,(w) \tag{7.46}$$

where

$$A\,(w) = \begin{bmatrix} \sum w_i(\Delta x_i)^2 & \sum w_i(\Delta x_i \Delta y_i) & \sum w_i(\Delta x_i \Delta z_i) \\ \sum w_i(\Delta x_i \Delta y_i) & \sum w_i(\Delta y_i)^2 & \sum w_i(\Delta y_i \Delta z_i) \\ \sum w_i(\Delta x_i \Delta z_i) & \sum w_i(\Delta y_i \Delta z_i) & \sum w_i(\Delta z_i)^2 \end{bmatrix},$$

$$(grad\;F)_o^T = \begin{bmatrix} F_x \\ F_y \end{bmatrix}, b\,(w) = \begin{bmatrix} \sum w_i(\Delta x_i \Delta F_i) \\ \sum w_i(\Delta y_i \Delta F_i) \\ \sum w_i(\Delta z_i \Delta F_i) \end{bmatrix} \tag{7.47}$$

The problem now consists in suitably determining weights so that as mentioned above, the 3-D least squares formulae for derivatives reduce to 1-D formulae.

7.1 Calculation of the weights for three dimensional least squares formulae

There are various methods for calculation of weights. We describe two methods below.

(I) One method of calculation of the weights for the three dimensional formulae is similar to that of the two dimensional formulae, but the weights have to be calculated such that the products $\sum w_i\,(\Delta x_i \Delta y_i)$, $\sum w_i\,(\Delta y \Delta z_i)$ and $\sum w_i\,(\Delta z_i \Delta x_i)$ simultaneously go to zero. Consider Fig.35 which shows the quadrants of split stencil

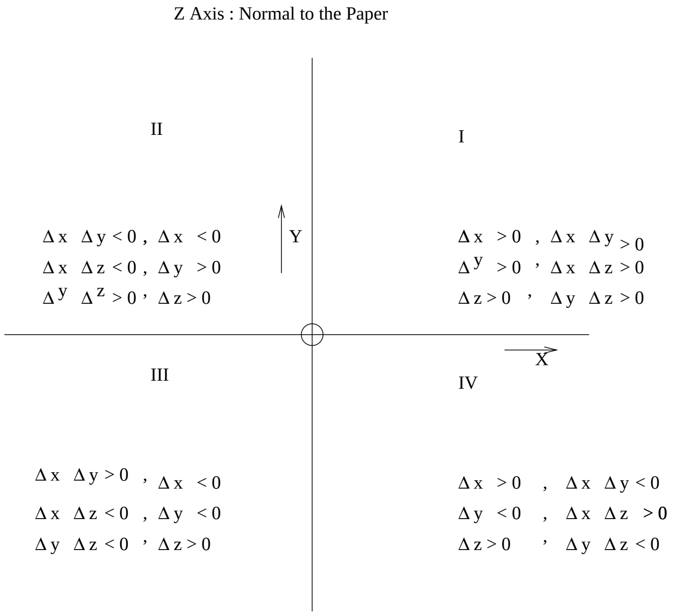


Figure 35: Quadrants of split stencil in 3-Dimension

necessary for enforcing upwinding for the case $\Delta z_i > 0$.This split stencil is used for calculating the negative fluxes in the z direction making use of the points in all the four quadrants shown as I, II,III and IV in the figure. Here we observe that each of the products $\Delta x_i \Delta y_i$, $\Delta y_i \Delta z_i$ and $\Delta z_i \Delta x_i$ is always positive in two of the quadrants while negative in the remaining two. This fact is used to calculate the weights for the corresponding points in the quadrants. Let w_I, w_{II}, w_{III} and w_{IV} be the weights assigned to the points in each of the quadrants I, II, III and IV respectively. The signs of cross products in the quadrants are shown in Table 1.

Table 1

Sr. No.	Quadrant	$\Delta x \Delta y$	$\Delta y \Delta z$	$\Delta z \Delta x$
1.	I	+	+	+
2.	II	-	+	-
3.	III	+	-	-
4.	IV	-	-	+

Now the conditions to be satisfied are :

$$\sum w_i \left(\Delta x_i \Delta y_i\right) = 0$$

$$\sum w_i \left(\Delta y_i \Delta z_i\right) = 0$$

$$\sum w_i \left(\Delta z_i \Delta x_i\right) = 0.$$

Using the notations introduced before, we obtain the following equations

$$w_I C_{xy}^I + w_{II} C_{xy}^{II} + w_{III} C_{xy}^{III} + w_{IV} C_{xy}^{IV} = 0 \tag{7.48}$$

$$w_I C_{yz}^I + w_{II} C_{yz}^{II} + w_{III} C_{yz}^{III} + w_{IV} C_{yz}^{IV} = 0 \tag{7.49}$$

$$w_I C_{zx}^I + w_{II} C_{zx}^{II} + w_{III} C_{zx}^{III} + w_{IV} C_{zx}^{IV} = 0 \tag{7.50}$$

Writing the system of equations in matrix form

$$\begin{bmatrix} C_{xy}^I & C_{xy}^{II} & C_{xy}^{III} & C_{xy}^{IV} \\ C_{yz}^I & C_{yz}^{II} & C_{yz}^{III} & C_{yz}^{IV} \\ C_{zx}^I & C_{zx}^{II} & C_{zx}^{III} & C_{zx}^{IV} \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{III} \\ w_{IV} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.51}$$

Looking at Eq.(7.51), we see that we have to solve a system of three equations for the four unknowns viz. w_I, w_{II}, w_{III} and w_{IV} Using the signs given in Table 1, the above system of equations (7.51) can be written as

$$\begin{bmatrix} |C_{xy}^I| & -|C_{xy}^{II}| & |C_{xy}^{III}| & -|C_{xy}^{IV}| \\ |C_{yz}^I| & |C_{yz}^{II}| & -|C_{yz}^{III}| & -|C_{yz}^{IV}| \\ |C_{zx}^I| & -|C_{zx}^{II}| & -|C_{zx}^{III}| & |C_{zx}^{IV}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{III} \\ w_{IV} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.52}$$

Now, the system of equations in (7.52)can be written in the following ways

$$\begin{bmatrix} |C^{I}_{xy}| & -|C^{II}_{xy}| & |C^{III}_{xy}| \\ |C^{I}_{yz}| & |C^{II}_{yz}| & -|C^{III}_{yz}| \\ |C^{I}_{zx}| & -|C^{II}_{zx}| & -|C^{III}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{III} \end{bmatrix} = -w_{IV} \begin{bmatrix} -|C^{IV}_{xy}| \\ -|C^{IV}_{yz}| \\ |C^{IV}_{zx}| \end{bmatrix} \tag{7.53}$$

$$\begin{bmatrix} |C^{I}_{xy}| & -|C^{II}_{xy}| & -|C^{IV}_{xy}| \\ |C^{I}_{yz}| & |C^{II}_{yz}| & -|C^{IV}_{yz}| \\ |C^{I}_{zx}| & -|C^{II}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{IV} \end{bmatrix} = -w_{III} \begin{bmatrix} |C^{III}_{xy}| \\ -|C^{III}_{yz}| \\ -|C^{III}_{zx}| \end{bmatrix} \tag{7.54}$$

$$\begin{bmatrix} |C^{I}_{xy}| & |C^{III}_{xy}| & -|C^{IV}_{xy}| \\ |C^{I}_{yz}| & -|C^{III}_{yz}| & -|C^{IV}_{yz}| \\ |C^{I}_{zx}| & -|C^{III}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{III} \\ w_{IV} \end{bmatrix} = -w_{II} \begin{bmatrix} -|C^{II}_{xy}| \\ |C^{II}_{yz}| \\ -|C^{II}_{zx}| \end{bmatrix} \tag{7.55}$$

$$\begin{bmatrix} -|C^{II}_{xy}| & |C^{III}_{xy}| & -|C^{IV}_{xy}| \\ |C^{II}_{yz}| & -|C^{III}_{yz}| & -|C^{IV}_{yz}| \\ -|C^{II}_{zx}| & -|C^{III}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_{II} \\ w_{III} \\ w_{IV} \end{bmatrix} = -w_{I} \begin{bmatrix} |C^{I}_{xy}| \\ |C^{I}_{yz}| \\ |C^{I}_{zx}| \end{bmatrix} \tag{7.56}$$

We can use any of the above linear set of Eqns. (7.53),(7.54), (7.55) or (7.56) to obtain weights by assuming the free parameter as unity. For example, we can use the system (7.53) and assume $w_{IV} = 1.0$ and then obtain w_I,w_{II} and w_{III} as the solution of (7.53). It will be shown later sometimes weights become negative, if say we solve (7.53). In such a case, we can consider then another system (7.54) and keep shifting the system to be solved till we get positive weights. In many cases, as will be shown later, this method works.

(II) In the second method, we are not interested in finding the weights such that all the (x, y, z) directions become the eigen directions as done in the first method described above. Suppose, we want to calculate the x derivative using the given connectivity of the node. In order to reduce the three dimensional formula for the x derivative to the one dimensional formula for the x derivative at the node, we require that only the x direction be made the eigen-direction of the least squares matrix $A\,(w)$. This condition is less restrictive and gives more flexibility in calculating the weights ensuring the positivity constraint. Mathematically , if we minimize the weighted sum of the squares of the error in Eq.(7.45) with respect to F_{xo}, we get

$$\begin{aligned} \frac{dE^2}{dF_{xo}} &= -2\sum w_i\left(\Delta F_i - \Delta x_i F_{xo} - \Delta y_i F_{y_o} - \Delta z_i F_{zo}\right)(\Delta x_i) \\ &= -2\sum \left(w_i \Delta F_i \Delta x_i - w_i {\Delta x_i}^2 F_{xo} - w_i \Delta y_i \Delta x_i F_{y_o} - w_i \Delta z_i \Delta x_i F_{zo}\right) \end{aligned} \tag{7.57}$$

From Eq.(7.57), we see that if we satisfy the conditions $\sum(w_i \Delta y_i \Delta x_i) = 0$ and $\sum(w_i \Delta z_i \Delta x_i) = 0$, we will be able to use only a 1-D formula for the x derivative instead of a 3-D formula. Now, using the notations described above, we have to solve the following two equations for the four unknown weights:

$$w_I C^I_{xy} + w_{II} C^{II}_{xy} + w_{III} C^{III}_{xy} + w_{IV} C^{IV}_{xy} = 0 \tag{7.58}$$

$$w_I C^I_{zx} + w_{II} C^{II}_{zx} + w_{III} C^{III}_{zx} + w_{IV} C^{IV}_{zx} = 0 \tag{7.59}$$

The system of equations (7.59) can be written in matrix form as

$$\begin{bmatrix} C^I_{xy} & C^{II}_{xy} & C^{III}_{xy} & C^{IV}_{xy} \\ C^I_{zx} & C^{II}_{zx} & C^{III}_{zx} & C^{IV}_{zx} \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{III} \\ w_{IV} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.60}$$

Using the signs given in Table 1, the above system of equations (7.60) reduces to

$$\begin{bmatrix} |C^I_{xy}| & -|C^{II}_{xy}| & |C^{III}_{xy}| & -|C^{IV}_{xy}| \\ |C^I_{zx}| & -|C^{II}_{zx}| & -|C^{III}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \\ w_{III} \\ w_{IV} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.61}$$

Now there are again various combinations in which the system of equations in (7.61) can be written. Different possible ways are

$$\begin{bmatrix} |C^I_{xy}| & -|C^{II}_{xy}| \\ |C^I_{zx}| & -|C^{II}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{II} \end{bmatrix} = - \begin{bmatrix} |C^{III}_{xy}| & -|C^{IV}_{xy}| \\ -|C^{III}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_{III} \\ w_{IV} \end{bmatrix} \tag{7.62}$$

$$\begin{bmatrix} |C^I_{xy}| & |C^{III}_{xy}| \\ |C^I_{zx}| & -|C^{III}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{III} \end{bmatrix} = - \begin{bmatrix} -|C^{II}_{xy}| & -|C^{IV}_{xy}| \\ -|C^{II}_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_{II} \\ w_{IV} \end{bmatrix} \tag{7.63}$$

$$\begin{bmatrix} |C^I_{xy}| & -|C^{IV}_{xy}| \\ |C^I_{zx}| & |C^{IV}_{zx}| \end{bmatrix} \begin{bmatrix} w_I \\ w_{IV} \end{bmatrix} = - \begin{bmatrix} -|C^{II}_{xy}| & |C^{III}_{xy}| \\ -|C^{II}_{zx}| & -|C^{III}_{zx}| \end{bmatrix} \begin{bmatrix} w_{II} \\ w_{III} \end{bmatrix} \tag{7.64}$$

We observe that as against method (I), here we have two free variables for each of the system of linear algebraic equations. We can form other combinations of the system of equations to solve for the weights, each such combination giving us one possible set of weights satisfying $\sum w_i \Delta x_i \Delta y_i = \sum w_i \Delta x_i \Delta z_i = 0$. Thus this method gives us more flexibility in calculating the weights enforcing the positivity constraint.

7.2 A few examples of weight calculation by above methods in 3-D

The least squares matrix for a 3-D distribution of points is

$$A = \begin{bmatrix} \sum \Delta x_i^2 & \sum \Delta x_i \Delta y_i & \sum \Delta x_i \Delta z_i \\ \sum \Delta x_i \Delta y_i & \sum \Delta y_i^2 & \sum \Delta y_i \Delta z_i \\ \sum \Delta x_i \Delta z_i & \sum \Delta y_i \Delta z_i & \sum \Delta z_i^2 \end{bmatrix}$$

while the weighted least squares matrix for a 3-D distribution of points is

$$A(w) = \begin{bmatrix} \sum w_i \Delta x_i^2 & \sum w_i \Delta x_i \Delta y_i & \sum w_i \Delta x_i \Delta z_i \\ \sum w_i \Delta x_i \Delta y_i & \sum w_i \Delta y_i^2 & \sum w_i \Delta y_i \Delta z_i \\ \sum w_i \Delta x_i \Delta z_i & \sum w_i \Delta y_i \Delta z_i & \sum w_i \Delta z_i^2 \end{bmatrix}$$

1 Consider a node with its neighbours in 3-D having the coordinates as given in the Table 2:

Table 2

Sr. No.	X Coordinate	Y Coordinate	Z Coordinate
Node: 0.	-0.201690	0.067675	1.030315
Nbr : 1.	-0.193736	0.089075	1.030315
Nbr : 2.	-0.101401	0.034024	1.047563
Nbr : 3.	-0.197736	0.064248	0.978148
Nbr : 4.	-0.103389	0.035747	1.100604
Nbr : 5.	-0.099412	0.032301	0.994522
Nbr : 6.	-0.189937	0.084565	0.978148
Nbr : 7.	-0.197534	0.093586	1.082483
Nbr : 8.	-0.097401	0.044783	1.047563
Nbr : 9.	-0.099311	0.047051	1.100604
Nbr :10.	-0.095492	0.042516	0.994522
Nbr :11.	-0.104289	0.022892	1.047563
Nbr :12.	-0.106334	0.024051	1.100604
Nbr :13.	-0.102244	0.021733	0.994522

The various views of the node and its connectivity tabulated in Table 2 are plotted in Figs.(36),(37),(38)and (39). The node in the figures is represented by a 'circle' while its neighbours are represented by a 'plus' sign. The least squares matrix for

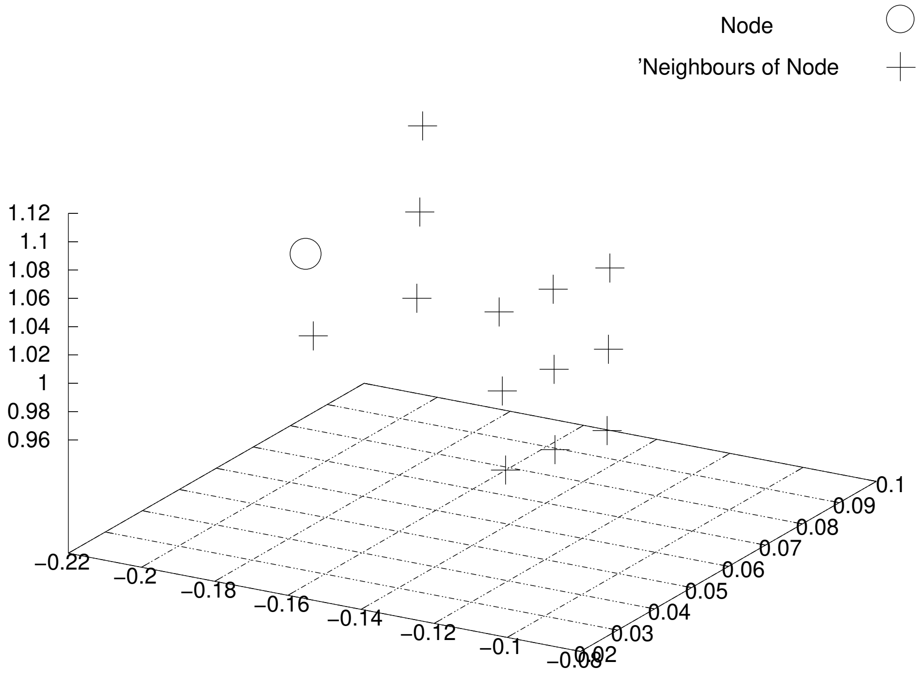


Figure 36: Isometric view of the node and its connectivity tabulated in Table 2

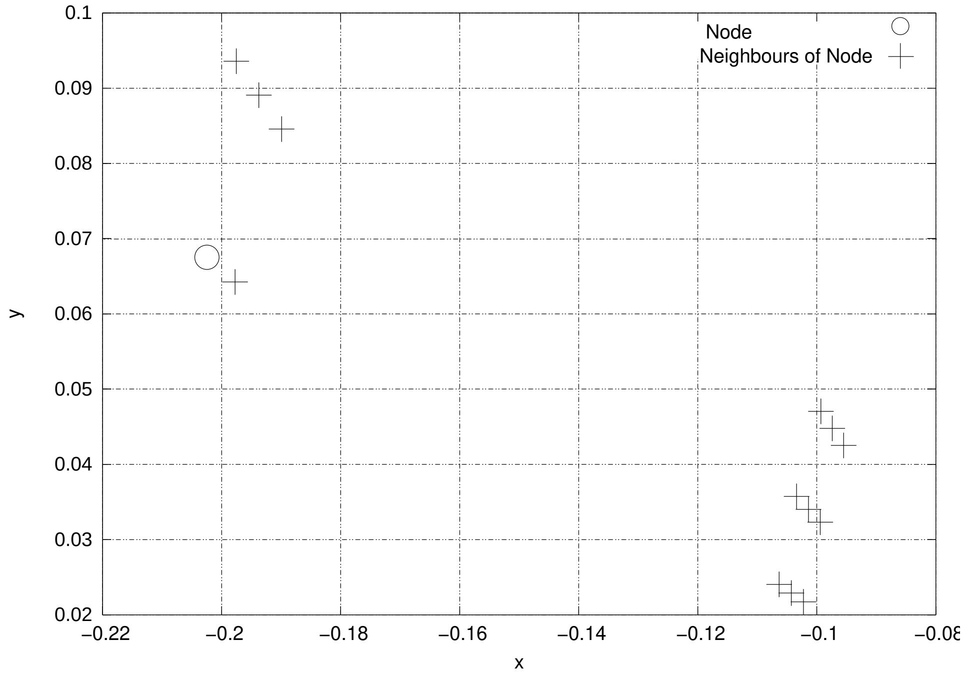


Figure 37: xy view of the node and its connectivity tabulated in Table 2

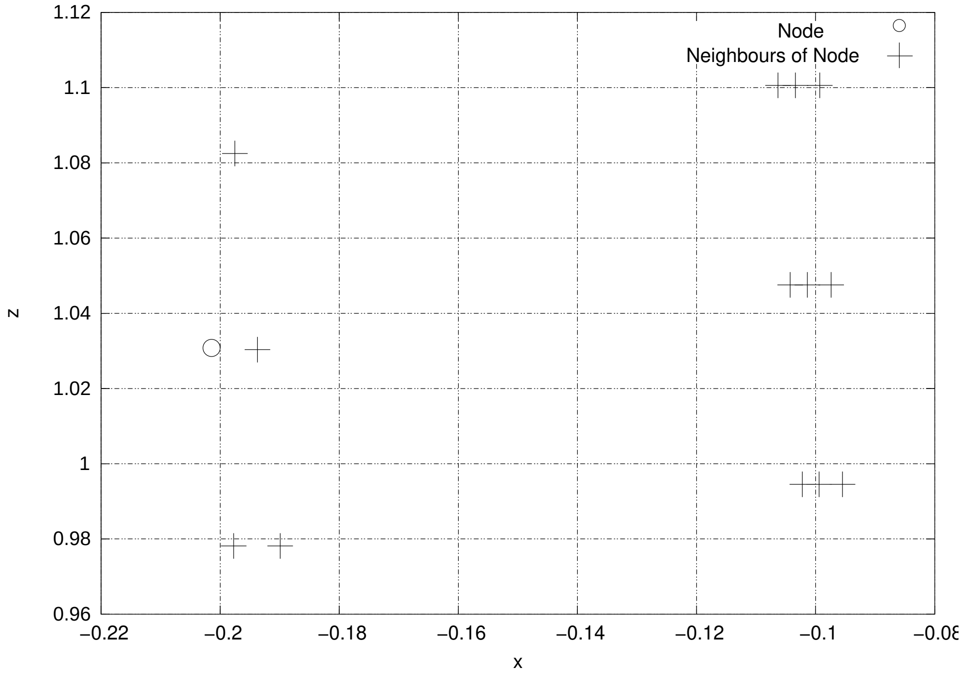


Figure 38: xz view of the node and its connectivity tabulated in Table 2

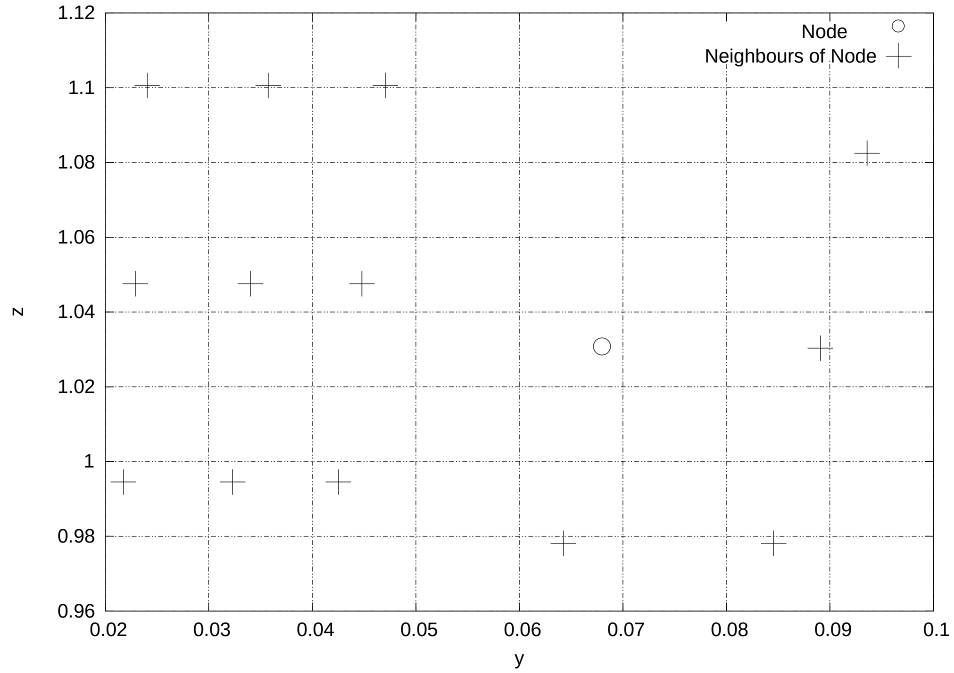


Figure 39: yz view of the node and its connectivity tabulated in Table 2

the above node is

$$A \quad = \begin{bmatrix} 0.091521 & -0.029930 & 0.014393 \\ -0.029930 & 0.012413 & -0.004047 \\ 0.014393 & -0.004047 & 0.027722 \end{bmatrix}$$

The coefficients defined above in the previous section for this particular node are given in the Table 3 :

Table 3

N	C_{xy}^N	C_{xz}^N	C_{yz}^N
I	0.000278	0.000217	0.001352
II	-0.019534	0.026017	-0.008508
III	-0.010872	-0.011228	0.003990
IV	0.000199	-0.000613	-0.000881

By Method I described above, we get the following weights :

$$w_I = 0.604729,\ w_{II} = 0.015023,\ w_{III} = 0.003005,\ w_{IV} = 0.796284$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 0.024843,\ w_{III} = 0.004969,\ w_{IV} = 1.316702$$

We see that in this case, all the weights calculated by first method are positive. The weighted least squares matrix now becomes :

$$A\,(w) \quad = \begin{bmatrix} 0.001150 & 0.000000 & 0.000000 \\ 0.000000 & 0.001027 & 0.000000 \\ 0.000000 & 0.000000 & 0.004069 \end{bmatrix}$$

Using Method II, we get the following weights for the same node :

$$w_I = 1.000000,\ w_{II} = 0.019232,\ w_{III} = 0.009266,\ w_{IV} = 1.000000$$

and the weighted least squares matrix now is :

$$A\,(w) \quad = \begin{bmatrix} 0.001659 & 0.000000 & 0.000000 \\ 0.000000 & 0.001587 & 0.000344 \\ 0.000000 & 0.000344 & 0.005806 \end{bmatrix}$$

In this example,we have seen that we have calculated two independent sets of positive weights for the node such that the x-derivative formula in higher dimensions reduces to the corresponding one dimensional formula. Now, some examples will be given when the weights calculated by one method do not satisfy the positivity condition.

2 Consider another node with its neighbours in 3-D having the coordinates as given in the Table 4 :

Table 4

Sr. No.	X Coordinate	Y Coordinate	Z Coordinate
Node: 0.	-0.507206	-0.907216	-0.110103
Nbr : 1.	-0.412599	-0.956996	-0.110103
Nbr : 2.	-0.498855	-0.892279	-0.219000
Nbr : 3.	-0.497261	-0.861281	-0.104528
Nbr : 4.	-0.489074	-0.847101	-0.207912
Nbr : 5.	-0.500000	-0.866024	0.000000
Nbr : 6.	-0.404508	-0.908541	-0.104528
Nbr : 7.	-0.420689	-1.005452	-0.115678
Nbr : 8.	-0.414871	-0.962268	0.000000
Nbr : 9.	-0.405805	-0.941240	-0.219000
Nbr :10.	-0.397848	-0.893582	-0.207912
Nbr :11.	-0.413762	-0.988898	-0.230089
Nbr :12.	-0.423006	-1.010990	0.000000
Nbr :13.	-0.406737	-0.913545	0.000000

The various views of the node and its connectivity tabulated in Table 4 are plotted in Figs.(40),(41),(42)and (43). The least squares matrix for the above node is

$$A = \begin{bmatrix} 0.084214 & -0.035425 & -0.004195 \\ -0.035425 & 0.041629 & -0.008187 \\ -0.004195 & -0.008187 & 0.105831 \end{bmatrix}$$

The coefficients defined above in the previous section for this particular node are given in the Table 5 :

Table 5

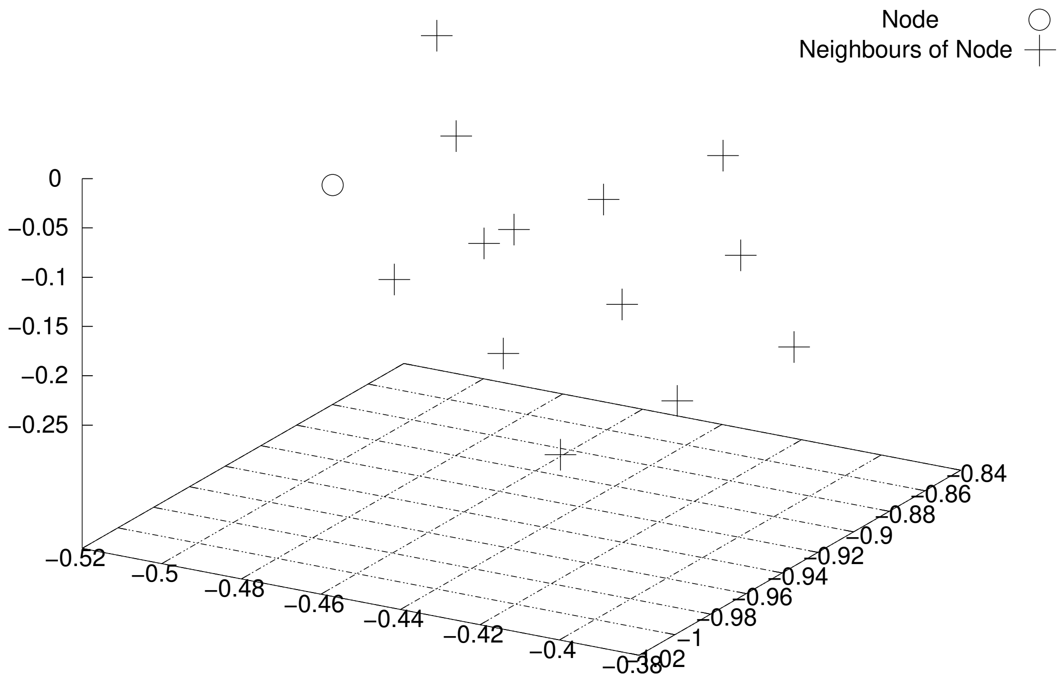


Figure 40: Isometric view of the node and its connectivity tabulated in Table 4

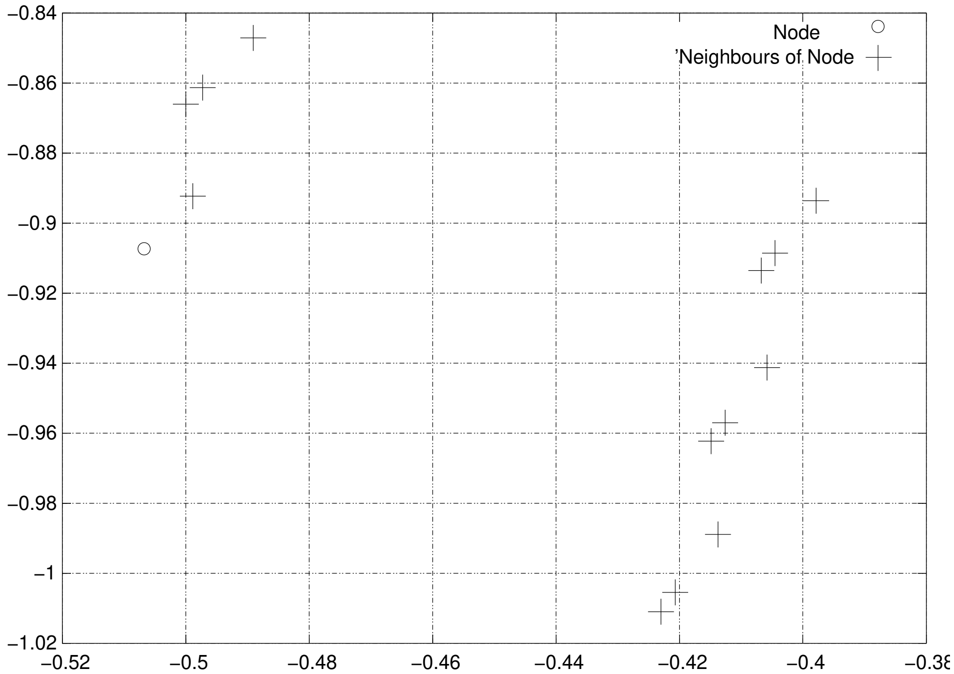


Figure 41: xy view of the node and its connectivity tabulated in Table 4

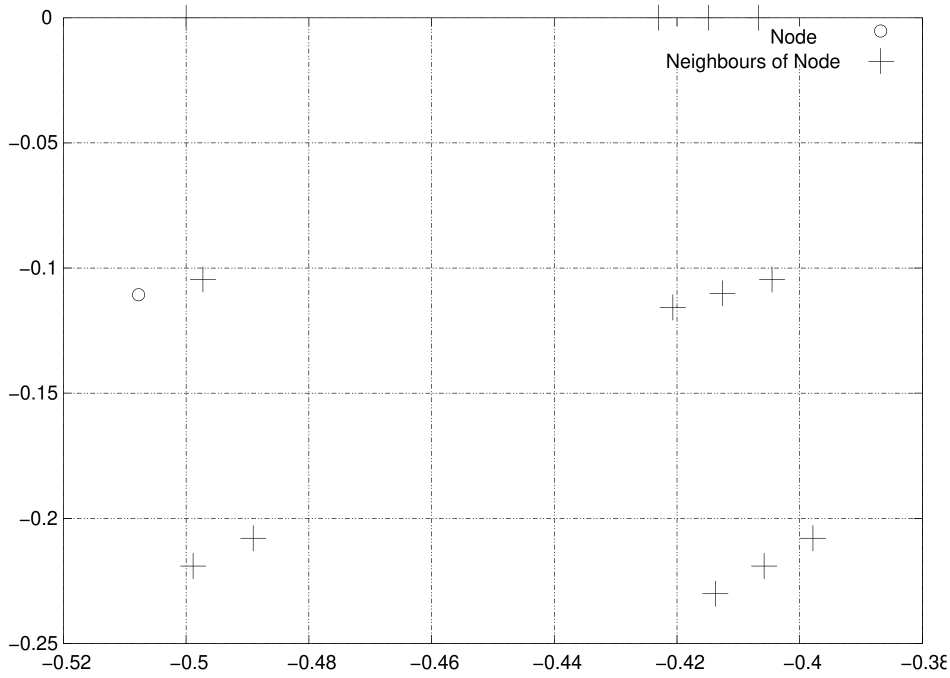


Figure 42: xz view of the node and its connectivity tabulated in Table 4

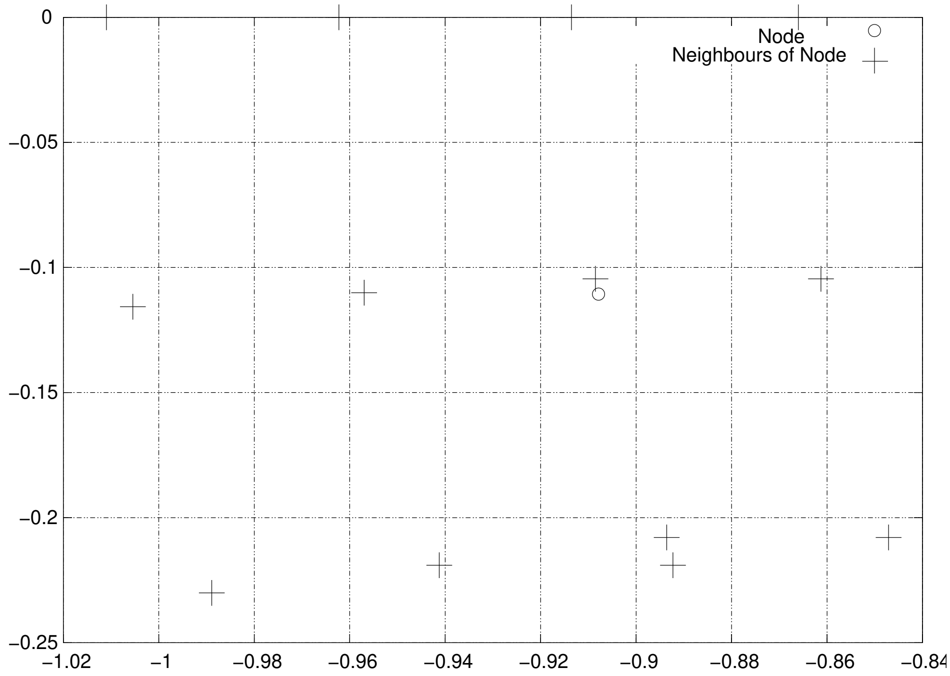


Figure 43: yz view of the node and its connectivity tabulated in Table 4

N	C_{xy}^N	C_{xz}^N	C_{yz}^N
I	0.000754	0.000849	0.004791
II	-0.019302	0.031072	-0.018191
III	-0.019582	-0.022737	0.014053
IV	0.002706	-0.013379	-0.008840

By Method I described above, we get the following weights

$$w_I = 0.949132,\ w_{II} = 0.093265,\ w_{III} = -0.013893,\ w_{IV} = 0.300427$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 0.098263,\ w_{III} = -0.014638,\ w_{IV} = 0.316528$$

We see that in this case, all the weights calculated by first method are not positive. But the weighted least squares matrix in this case too is diagonalized :

$$A(w) \quad = \begin{bmatrix} 0.007704 & 0.000000 & 0.000000 \\ 0.000000 & 0.006101 & 0.000000 \\ 0.000000 & 0.000000 & 0.023876 \end{bmatrix}$$

We have observed that using Method I, we were unsuccessful in calculating all positive weights for this node. So now using Method II, we get the following weights for the same node :

$$w_I = 0.999295,\ w_{II} = 0.002185,\ w_{III} = 0.037064,\ w_{IV} = 0.005489$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 0.002187,\ w_{III} = 0.037090,\ w_{IV} = 0.005493$$

It is observed that now all the weights calculated are positive. The weighted least squares matrix now is :

$$A(w) \quad = \begin{bmatrix} 0.001300 & 0.000000 & 0.000000 \\ 0.000000 & 0.004510 & 0.005221 \\ 0.000000 & 0.005221 & 0.013369 \end{bmatrix}$$

Using another variant of Method II, we can calculate a new set of all positive weights for the same node, thus verifying that no unique set of all positive weights exist for any node in the domain. Another set of all positive weights for the same node are:

$$w_I = 1.000000,\ w_{II} = 0.024875,\ w_{III} = 0.024875,\ w_{IV} = 0.078942$$

The weighted least squares matrix for this set of weights is :

$$A(w) = \begin{bmatrix} 0.002910 & 0.000000 & 0.000000 \\ 0.000000 & 0.004965 & 0.003991 \\ 0.000000 & 0.003991 & 0.016160 \end{bmatrix}$$

Sometimes, it may also happen that one variant of Method II does not give all positive weights. But its another variant may still give positive weights. This is shown by the third example below.

3 Consider another node with its neighbours in 3-D having the coordinates as given in the Table 6 :

Table 6

Sr. No.	X Coordinate	Y Coordinate	Z Coordinate
Node: 0.	-0.926712	0.100584	-0.428429
Nbr : 1.	-0.911454	0.200067	-0.428429
Nbr : 2.	-0.878507	0.095352	-0.526667
Nbr : 3.	-0.908541	0.095492	-0.406737
Nbr : 4.	-0.861281	0.090524	-0.500000
Nbr : 5.	-0.895732	0.100180	-0.553333
Nbr : 6.	-0.893582	0.189937	-0.406737
Nbr : 7.	-0.864043	0.189660	-0.526667
Nbr : 8.	-0.847101	0.180057	-0.500000
Nbr : 9.	-0.880985	0.199263	-0.553333
Nbr :10.	-0.913545	0.000000	-0.406737
Nbr :11.	-0.883346	0.000000	-0.526667
Nbr :12.	-0.886024	0.000000	-0.500000
Nbr :13.	-0.900666	0.000000	-0.553333

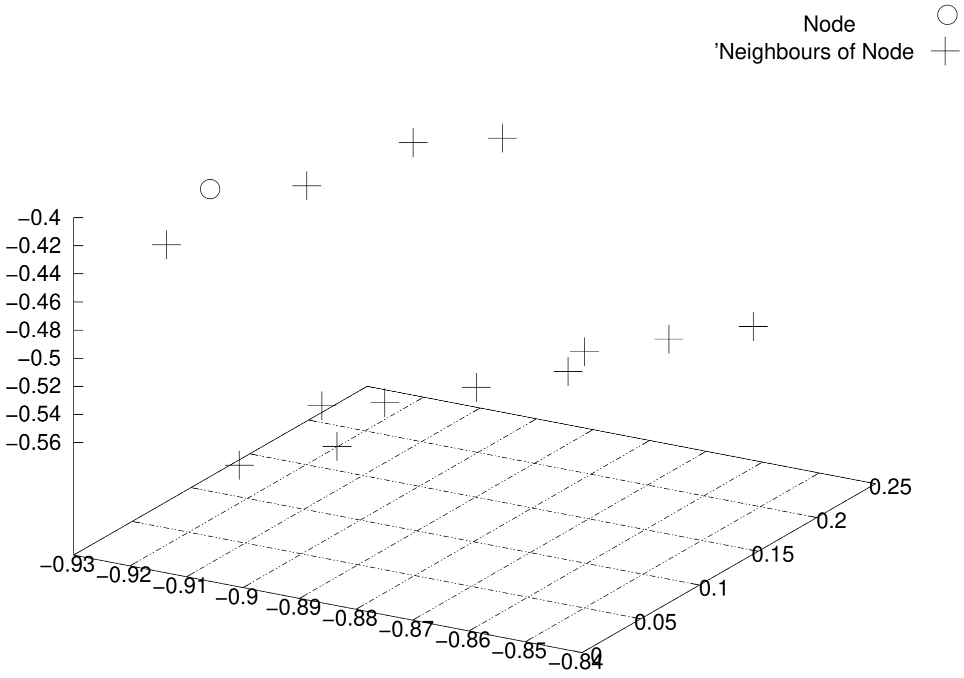


Figure 44: Isometric view of the node and its connectivity tabulated in Table 6

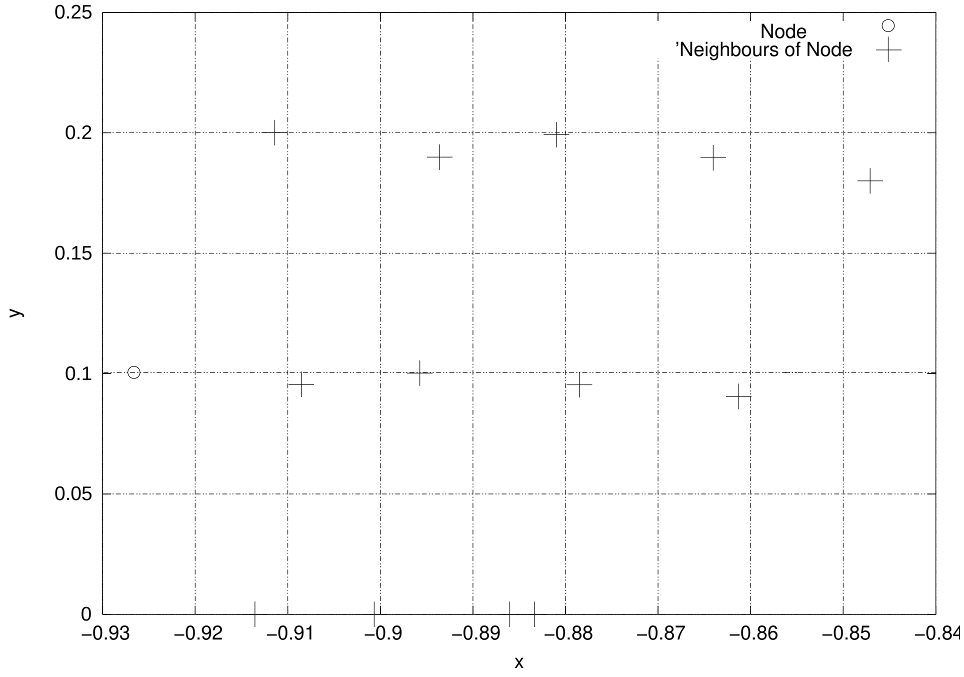


Figure 45: xy view of the node and its connectivity tabulated in Table 6

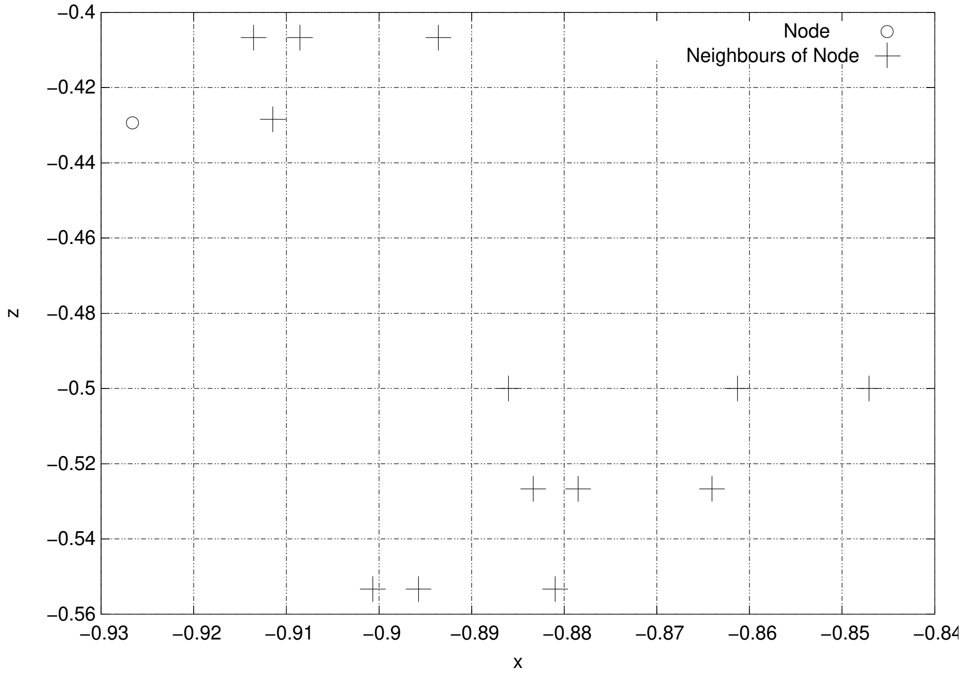


Figure 46: xz view of the node and its connectivity tabulated in Table 6

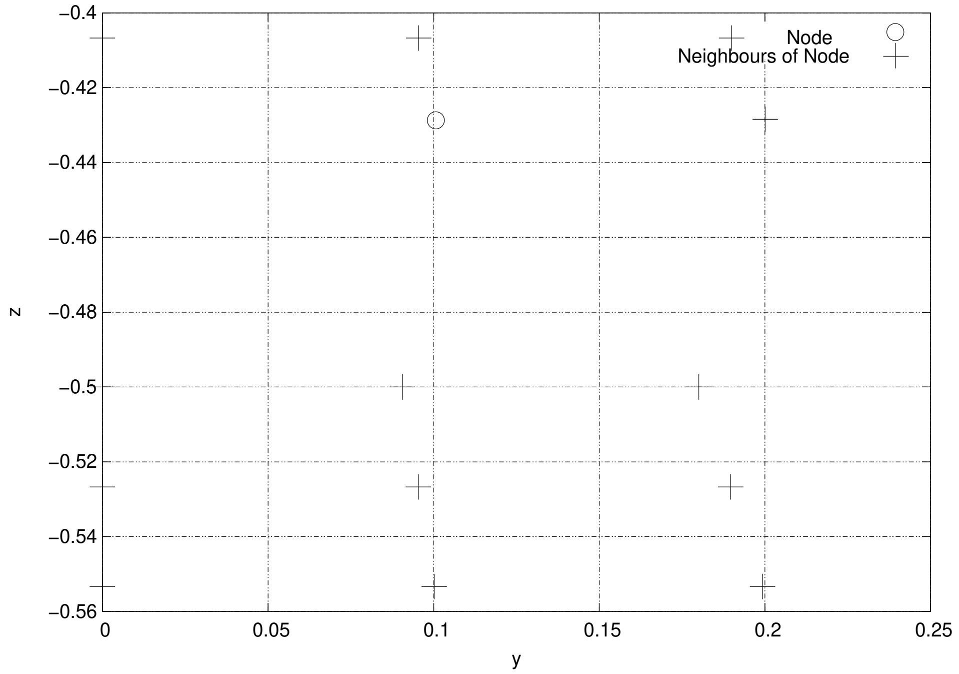


Figure 47: yz view of the node and its connectivity tabulated in Table 6

The various views of the node and its connectivity tabulated in Table 6 are plotted in Figs.(44),(45),(46)and (47). The least squares matrix for the above node is

$$A = \begin{bmatrix} 0.027997 & 0.005474 & -0.041312 \\ 0.005474 & 0.082492 & 0.003810 \\ -0.041312 & 0.003810 & 0.092534 \end{bmatrix}$$

The coefficients defined above in the previous section for this particular node are given in the Table 7:

Table 7

N	C^N_{xy}	C^N_{xz}	C^N_{yz}
I	0.004478	0.000719	0.001938
II	-0.001417	0.000680	-0.002292
III	-0.014009	-0.025145	0.030928
IV	0.016422	-0.017566	-0.026764

By Method I described above, we get the following weights

$$w_I = -0.418323,\ w_{II} = -0.907476,\ w_{III} = -0.038524,\ w_{IV} = 0.002913$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 2.169319,\ w_{III} = 0.092092,\ w_{IV} = -0.006964$$

We see that in this case, all the weights calculated by first method are not positive. But the weighted least squares matrix in this case too is diagonalized :

$$A = \begin{bmatrix} -0.001509 & 0.000000 & 0.000000 \\ 0.000000 & -0.017789 & 0.000000 \\ 0.000000 & 0.000000 & -0.003303 \end{bmatrix}$$

We have observed that using Method I, we were unsuccessful in calculating all positive weights for this node. So now using Method II, we get the following weights for the same node :

$$w_I = 0.979229,\ w_{II} = 0.013172,\ w_{III} = 0.134150,\ w_{IV} = -0.151460$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 0.013452,\ w_{III} = 0.136996,\ w_{IV} = -0.154672$$

It is observed that now also the weights calculated are not positive. The weighted least squares matrix now is :

$$A = \begin{bmatrix} 0.001290 & 0.000000 & 0.000000 \\ 0.000000 & 0.018099 & 0.010070 \\ 0.000000 & 0.010070 & 0.004022 \end{bmatrix}$$

Using another variant of Method II, we are able to calculate a set of all positive weights for the same node :

$$w_I = 0.305064,\ w_{II} = 1.000000,\ w_{III} = 0.021049,\ w_{IV} = 0.021049$$

Normalizing the weights obtained above with w_I, we get

$$w_I = 1.000000,\ w_{II} = 3.278001,\ w_{III} = 0.068999,\ w_{IV} = 0.068999$$

The weighted least squares matrix for this set of weights is :

$$A = \begin{bmatrix} 0.001460 & 0.000000 & 0.000000 \\ 0.000000 & 0.016744 & -0.001613 \\ 0.000000 & -0.001613 & 0.003003 \end{bmatrix}$$

Thus we have seen that there are a large number of positive weights with the help of which we can fully or partially digonalize the least squares matrix. The weighted least squares method gives us immense ability to manipulate the matrix $A(w)$ in any way we like. We have used the weighted least squares method to take advantage of the nice properties of the real, symmetrix matrix A obtained in the least squares method.

8 Conclusion

A new least squares formulation along eigenvectors has been developed. In fact, for any given x,y direction the weights in $A(w)$ can be suitably chosen so that the x,y directions are eigen directions of $A(w)$. Consequently, the least squares formulae for derivatives in multidimensions (2-D, 3-D) reduce to 1-D formulae thus considerably reducing the problem of code divergence faced by the standard LSKUM. In fact, the present work shows that bad connectivity has no absolute meaning, it is bad or good depending on the way matrix A is formed in least squares formulation. A connectivity $N(P_o)$ of node P_o which is bad for $A(w_1)$ (infact causing code divergence) can become good for $A(w_2)$ leading to code convergence. This approach can offer tremendous advantage in LSKUM in terms of robustness and accuracy and give the user a kind of "care free flexibility" while generating connectivity. There are many LSKUM and q-LSKUM codes being used now for computation of flows around practical configurations. The value addition to these codes by introducing the above idea (which is a very small coding effort!) can be immense and this potential needs to be exploited further.

References

[1] Anandhanarayanan, K. (2003) *Development and Applications of a Gridfree Kinetic Upwind Solver to Multibody Configurations*, PhD. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[2] Ghosh, A.K. (1996) *Robust Least Squares Kinetic Upwind Method for Inviscid Compressible Flows*, PhD. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[3] Ghosh, A.K. and Deshpande, S.M. (1995) *Least Squares Kinetic Upwind Method for Inviscid Compressible Flows*, AIAA Paper No. 95-1735.

[4] Jameson, Antony (1995) *Analysis and Design of Numerical Schemes for Gas Dynamics 1 Artificial Diffusion, Upwind Biasing, Limiters and Their Effect on Accuracy and Multigrid Convergence*, RIACS Technical Report 94.15, International Journal of Computational Fluid Dynamics, Vol. 4, 1995, pp. 171-218.

[5] Mandal, J.C. and Deshpande, S.M. (1994) *Kinetic Flux Vector Splitting for Euler Equations*, Computers and Fluids, Vol 23. No. 2, pp. 447-478.

[6] Mahendra, A.K. (2003) *Application of Least Squares Kinetic Upwind Method to Strongly Rotating Viscous Flows*, MSc. (Engg.) Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[7] Praveen, C. (2004) *Development and Applications of Kinetic Meshless Methods for Euler Equations*, PhD. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[8] Ramesh, V. (2001) *Least Squares Grid-Free Kinetic Upwind Method*, PhD. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[9] Sashi Kumar, G.N., Mahendra, A.K. and Deshpande, S.M. (2004) *Spectral Resolution and Order of Accuracy of Least Squares Scheme*, FM Report (under preparation), Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

[10] Strang, Gilbert *Linear Algebra And Its Applications*, Third Edition.

[11] Varma, Mohan U., Raghurama Rao, S.V. and Deshpande, S.M. (2003) *Point generation using Quadtree Data Structures for Meshless Solvers*, FM Report No. 2003 FM 08, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.