# EMPLOYING DAUBECHIES WAVELET IN SOLUTION –BASED GRID ADAPTATION

**Khalid M. Sultan, Basim A. Belgasim and Gamal G. S. Hashem**

Mechanical Engineering Department
Faculty of Engineering
Garyounis University Benghazi-Libya
Dr-Khalid-Sultan@hotmail.de

## الملخص

في كثير من حالات حساب التدفق التي تجري هذه الايام سواءً لأغراض عملية أو بحثية يسعى الباحثون و المهندسون إلى زيادة الدقة في النتائج المتحصل عليها. إحدى الطرق لزيادة الدقة في النتائج المحسوبة للتدفق هي إجراء عملية تنعيم للشبكة الحسابية أثناء عملية الحساب، و لكي تحدث هذه العملية بشكل تلقائي أثناء الحساب يجب أن يحتوي محلل التدفق على جزء إضافي خاص في برنامج المحلل تكون مهمته القيام بالبحث عن الخلايا ذات التغيرات الكبيرة في الشبكة الحسابية. يقوم هذا الجزء الإضافي الخاص بالبحث في خلايا الشبكة الحسابية كلها و يخضعها للإختبار عن طريق حساسات التنعيم التي يحتوي عليها و عند استشعار أي حساس من هذه الحساسات لحاجة الخلية للتنعيم فإن هذه الخلية تصبح مرشحة للتنعيم. في هذه الورقة تم توظيف مويجة داوبشي لمعالجة البيانات في محلل تدفق تم تطويره محليا و ذلك لتقليص نطاق البحث في خلايا الشبكة الحسابية بحيث لا يتم البحث في كامل خلايا الشبكة الحسابية بل يتم البحث فقط في الجزء المحتوي على التفاصيل التي تقدمها مويجة داوبشي و هذا بالتالي يقلص الوقت و الجهد كما بينت الحالات الدراسية في هذه الورقة.

## ABSTRACT

Many flow computation problems are performed in the present time for several real-life cases and/or research purposes. However, in most of these computations there is always a tendency from engineers and researchers to increase the accuracy of the solutions obtained. Increasing the accuracy of such computations could be obtained by many means, among them is to refine the computational grid during the solution process. In order to automate this process, an ancillary algorithm must be incorporated into the flow solver in such a way that a search process can be carried out to identify those regions of high gradients. This presumes that the search process will include all of the computational cells and accordingly investigating the predesigned adaptation sensors for a possible nomination of a particular cell for refinement. In the present work the Daubechies wavelet transform is incorporated in an in–house made unstructured Cartesian grid generator and flow solver. The details of the Daubechies wavelet algorithm are demonstrated and the application of the adaptation criteria is carried out only on the wavelet function instead of applying it on the whole sequence of adaptation sensors. Case studies for verification purposes are carried out.

**KEYWORDS:** Daubechies wavelets; Adaptation sensors; Grid adaptation; Unstructured cartesian grid.

## INTRODUCTION

It is well known that the accuracy of the obtained solution depends on the grid size, especially; in regions of high gradients. Usually an ancillary algorithm is included as an optional mechanism that can be switched on whenever needed. This algorithm is composed of two main processes, the first process is the computations of the adaptation sensors, and the second process is comparing each of the computed sensors with its respective threshold and consecutively either tagging the cell for refinement if a sensor's absolute value is greater than its respective threshold or leaving the cell without refinement if a sensor's absolute value is smaller than its respective threshold. As the reader can conclude, the second process is in essence a search process for those cells that must be refined. This search process is a time consuming process and lends itself to optimization through the employment of a wavelet transform. Wavelet transform compresses a given data set by splitting it up into a low resolution part (scale) and a high resolution part (detail). The wavelet function (the detail) is in our case the important criterion since it represents the amplitude of the difference between the given data. Hence the search process can be reduced by searching only the wavelet function instead of searching the whole data set. Form the many wavelet functions available, the Daubechies' wavelet of four data point window (D4 wavelet) was employed in this work since it is better suited to deal with general applications [1].

## MATHEMATICAL MODEL

For an inviscid, unsteady compressible, and two dimensional flows with both mass diffusion and thermal conductivity are neglected, the conservation laws are:

**Continuity equation:**

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho V)}{\partial y} = 0 \qquad (1)$$

**Momentum equation:**

x-direction

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2 + p)}{\partial x} + \frac{\partial(\rho u V)}{\partial y} = 0 \qquad (2)$$

y-direction

$$\frac{\partial(\rho V)}{\partial t} + \frac{\partial(\rho u V)}{\partial x} + \frac{\partial(\rho V^2 + p)}{\partial y} = 0 \qquad (3)$$

**Energy equation:**

$$\frac{\partial(\rho e)}{\partial t} + \frac{\partial\left[u(\rho e + p)\right]}{\partial x} + \frac{\partial\left[V(\rho e + p)\right]}{\partial y} = 0 \qquad (4)$$

Where $\rho$ is the fluid density, $u$ is the fluid velocity in x-direction, $V$ is the fluid velocity in y-direction, $p$ is the pressure and $e$ is the total energy. One more equation is needed in order to have a number of equations that is equal to the number of unknowns $(\rho, u, V, p, T, and\ e)$. Using the thermodynamic relation $C_v = R/(\gamma - 1)$ and the definition of the specific internal energy $\hat{u} = C_v T$, the equation of state can be reformulated to relate the pressure $p$ to the conservative variable $\rho e$ as:

$$p = (\gamma - 1)(\rho e - \frac{1}{2}\rho \vec{V}^{2}) \tag{5}$$

Equation (1) to (5) can be rewritten in a dimensionless compact form as:

$$\frac{\partial \vec{Q}}{\partial t^{*}} + \frac{\partial \vec{E}}{\partial x^{*}} + \frac{\partial \vec{F}}{\partial y^{*}} = 0 \tag{6}$$

where:

$$\vec{Q} = \begin{bmatrix} \rho^{*} \\ \rho^{*} u^{*} \\ \rho^{*} V^{*} \\ \rho^{*} e^{*} \end{bmatrix}, \quad \vec{E} = \begin{bmatrix} \rho^{*} u^{*} \\ \rho^{*} u^{*2} + p^{*} \\ \rho^{*} u^{*} V^{*} \\ u^{*}\left(\rho^{*} e^{*} + p^{*}\right) \end{bmatrix} \quad \text{and} \quad \vec{F} = \begin{bmatrix} \rho^{*} V^{*} \\ \rho^{*} V^{*} u^{*} \\ \rho^{*} V^{*2} + p^{*} \\ V^{*}\left(\rho^{*} e^{*} + p^{*}\right) \end{bmatrix}$$

The dimensionless variables are defined as shown in the nomenclature. It is worth noting here that the governing equations in the dimensionless form are similar to those that are in dimensional form, so for reason of simplicity the asterisk will be dropped from now and as this text goes on.

**Initial and boundary conditions**

The governing equation, equation (6), is a partial differential equation with both time and space derivatives and thus requires both initial and boundary conditions. Specifying the initial conditions is a trivial task since the flow can be assumed to be parallel flow at the start of calculation. This is a conventional approach which is used for the solution of inviscid flows, for instance see [2] and [3]. The boundary conditions, on the other side, are classified into two types: Far-field boundary conditions and fluid-body boundary conditions. The far-field boundary conditions are based on the method of characteristics, see [4], and summarized as follows:

- Subsonic flow
-

$$\left\{ \begin{aligned} \vec{Q}_{inflow} &= \left( \rho_{\infty},\ \rho_{\infty}\vec{V}_{\infty},\ \frac{p_{cd}}{\gamma - 1} + \frac{1}{2}\rho_{\infty}\vec{V}_{\infty}^{2} \right)^{T} \\ \vec{Q}_{outflow} &= \left( \rho_{cd},\ \rho_{cd}\vec{V}_{cd},\ \frac{p_{\infty}}{\gamma - 1} + \frac{1}{2}\rho_{cd}\vec{V}_{cd}^{2} \right)^{T} \end{aligned} \right\} \tag{7}$$

- Supersonic flow

$$\left\{\begin{aligned}\vec{Q}_{inflow} &= \left(\rho_\infty,\ \rho_\infty\vec{V}_\infty,\ \frac{p_\infty}{\gamma-1}+\frac{1}{2}\rho_\infty\vec{V}_\infty^2\right)^T \\ \vec{Q}_{outflow} &= \left(\rho_{cd},\ \rho_{cd}\vec{V}_{cd},\ \frac{p_{cd}}{\gamma-1}+\frac{1}{2}\rho_{cd}\vec{V}_{cd}^2\right)^T\end{aligned}\right\} \tag{8}$$

The fluid-body boundary conditions for an inviscid flow at the wall are extrapolated from the computational domain. In other words, the flow properties except the velocity are all extrapolated from the nearest flow cell. With respect to the velocity, for a nonporous wall, there can be no mass flow into or out of the wall this means that the flow velocity vector immediately adjacent to the wall must be tangent to the wall. If $\vec{n}$ is a unit vector at a point on the surface of the wall, the velocity at the wall can be given as:

$$\vec{V}_{sc} = \vec{V}_{fc} - \left(\vec{V}_{fc}\cdot\vec{n}\right)\cdot\vec{n} \tag{9}$$

Where $\vec{V}_{sc}$ is the velocity at the cut surface and $\vec{V}_{fc}$ is the velocity of the nearest flow cell.

## METHOD OF SOLUTION

The system of the conservation equations, in vector compact form equations (6), comprise a set of nonlinear partial differential equations containing both space and time derivatives. The spatial derivatives will be discretized using an upwind-high resolution scheme, namely the Advection Upstream Splitting Method (AUSM scheme), see [5]. The time derivatives, on the other hand, will be discretized using a five stage Runge-Kutta method.

### Discretization of the Convective Terms, the AUSM Scheme

The AUSM scheme is one of several available upwind schemes that have the advantage of simplicity in programming without loosing the high accuracy. In order to explain this scheme, the flux $\vec{E}$ in equation (6) will be taken as an example. The flux in the y-direction will be treated similarly. The first step in this scheme is to split the flux $\vec{E}$ into convective and pressure terms, and a cell interface velocity $u_{1/2}$ is introduced which is used later to identify the upstream (upwind) direction, that is;

$$\vec{E} = u_{1/2}\begin{bmatrix}\rho \\ \rho u \\ \rho V \\ \rho H\end{bmatrix}_{L/R} + \begin{bmatrix}0 \\ P_L^+ + P_R^- \\ 0 \\ 0\end{bmatrix} \tag{10}$$

The interface velocity is given by:

$$u_{1/2} = a_{L/R}\ M_{1/2} \tag{11}$$

Where $a_{L/R}$ is the sound speed at the left or right side with respect to the cell face. The decision to take either the left or the right value is taken according to the upstream direction. Mathematically, this can be written as:

$$(\bullet)_{L/R} = \begin{cases} (\bullet)_L & if\ u_{1/2} \le 0 \\ (\bullet)_R & if\ u_{1/2} > 0 \end{cases} \tag{12}$$

Substituting equation (11) into equation (10) leads to:

$$\vec{E} = M_{1/2} \begin{bmatrix} \rho a \\ \rho au \\ \rho aV \\ \rho aH \end{bmatrix}_{L/R} + \begin{bmatrix} 0 \\ P_L^+ + P_R^- \\ 0 \\ 0 \end{bmatrix} \tag{13}$$

The convective interface Mach number $M_{1/2}$ is defined by combining the wave speed of left and right running waves, that is;

$$M_{1/2} = M_L^+ + M_R^- \tag{14}$$

In defining the split Mach number, $M_L^+$ and $M_R^-$, Liou and Steffen [5] have provided a second degree polynomial to be used if the flow is subsonic and a first degree polynomial if the flow is supersonic, that is;

$$M_{L/R}^{\pm} = \begin{cases} \pm\frac{1}{4}\left(M_{L/R} \pm 1\right)^2 & if\ \left|M_{L/R}\right| \le 1 \\ \frac{1}{2}\left(M_{L/R} \pm \left|M_{L/R}\right|\right) & if\ \left|M_{L/R}\right| > 1 \end{cases} \tag{15}$$

The pressure part of the flux $\vec{E}$ is calculated similarly by two polynomials provided by [5] as:

$$P_{L/R}^{\pm} = \begin{cases} \pm\frac{P}{4}\left(M_{L/R} \pm 1\right)^2\left(2 \pm M_{L/R}\right) & if\left|M_{L/R}\right| \le 1 \\ \frac{P}{2}\left(M_{L/R} \pm \left|M_{L/R}\right|\right)/\, M_{L/R} & if\left|M_{L/R}\right| > 1 \end{cases} \tag{16}$$

It should be noted here that the pressure term of the flux does not undergo any upwinding as it is clear from equation (13). This is expected because pressure waves can travel in all directions.

**Discretization of the Unsteady Terms (Runge-Kutta method)**

The discretization of the unsteady terms can be performed using either an explicit or an implicit time integration scheme. In order to avoid the greater memory requirement of the implicit scheme an explicit five stage Runge-Kutta integration scheme of the following form is selected:

$$\vec{Q}^{(0)} = \vec{Q}^{(n)}$$

$$\vec{Q}^{(1)} = \vec{Q}^{(0)} + \alpha_1 \Delta t \cdot \mathrm{Re}\,s\left(\vec{Q}^{(0)}\right)$$

$$\vec{Q}^{(2)} = \vec{Q}^{(1)} + \alpha_2 \Delta t \cdot \mathrm{Re}\,s\left(\vec{Q}^{(1)}\right)$$

$$\vec{Q}^{(3)} = \vec{Q}^{(2)} + \alpha_3 \Delta t \cdot \mathrm{Re}\,s\left(\vec{Q}^{(2)}\right)$$

$$\vec{Q}^{(4)} = \vec{Q}^{(3)} + \alpha_4 \Delta t \cdot \mathrm{Re}\,s\left(\vec{Q}^{(3)}\right)$$

$$\vec{Q}^{(5)} = \vec{Q}^{(4)} + \alpha_5 \Delta t \cdot \mathrm{Re}\,s\left(\vec{Q}^{(4)}\right)$$

$$\vec{Q}^{(n+1)} = \vec{Q}^{(5)}$$

The values of the coefficients $\alpha_{1\,to\,5}$ are given, as in [1], by:

$\alpha_1 = 0.059$ , $\alpha_2 = 0.14$ , $\alpha_3 = 0.273$ , $\alpha_4 = 0.5$ , and $\alpha_5 = 1$.

The time step can be computed by the procedure given by [6] as:

$$\frac{1}{\Delta t} = \frac{1}{\Delta t_x} + \frac{1}{\Delta t_y}$$

where:

$$\Delta t_x = \frac{CFL * \Delta x}{|u| + a} \quad , \quad \text{and} \quad \Delta t_y = \frac{CFL * \Delta y}{|V| + a}$$

Here CFL is the Courant-Fredrich-Levy number that has a value of 1.0 which is normally recommended for explicit schemes.

**Adaptation sensors and Daubechies wavelet**

The adaptation sensors that are most commonly used in inviscid flow computations are: The relative change in the density, the relative change in the total pressure, and the relative change in the magnitude of the velocity vector, these sensors are summarized as:

$$s_1 = \frac{\Delta \rho}{\rho} \quad , \quad s_2 = \frac{\Delta P_t}{P_t} \quad , \quad s_3 = \frac{\Delta |V|}{|V|} \tag{17}$$

where the total pressure $P_t$ is defined as :

$$P_t = P\left[\frac{(\gamma - 1)}{2} M^2 + 1\right]^{\frac{\gamma}{\gamma - 1}} \tag{18}$$

A widely used approach to calculate the refinement threshold is to employ the standard deviation of the data together with the mean value. That is, to set the refinement threshold to some fraction of a standard deviation above the mean value of the data distribution [7-9]. Using the mean value of the data together with the standard deviation, the refinement threshold can be written as:

$$Threshold \quad = Mean \ - \psi .\sigma \tag{19}$$

Where *Mean* is the mean value of data, $\sigma$ is the standard deviation of the data, and $\psi$ is a factor whose value is given by [9] as 0.5. Another simpler yet faster approach to compute the threshold was used in this work that was originally introduced by [10] who employed the *Median* of the data. The formula suggested by [10] is:

$$Threshold \quad = \frac{Mean}{0.6745} \tag{20}$$

Where the Median is the statistical median of the data, and the factor 0.6745 rescales the numerator so that the threshold is also suitable estimator for the standard deviation. The D4 wavelet has the following form:

$$SCL_i = a_0h_0 + a_1h_1 + a_2h_2 + a_3h_3 \tag{21}$$

$$DTL_i = a_0g_0 + a_1g_1 + a_2g_2 + a_3g_3 \tag{22}$$

Where the $a_i$ , $i=0,...,3$ are the values of the $S_1$ for each of the four cells included in the wavelet window. The $SCL_i$ is the scale of the data at this data range, and $DTL_i$ is the detail or the wavelet function. The wavelet coefficients $h_{0,...,3}$ and $g_{0,...,3}$ are given as:

$$\begin{aligned}
h_0 &= \frac{1+\sqrt{3}}{4\sqrt{2}} & g_0 &= h_3 \\
h_1 &= \frac{3+\sqrt{3}}{4\sqrt{2}} & g_1 &= -h_2 \\
h_2 &= \frac{3-\sqrt{3}}{4\sqrt{2}} & g_2 &= h_1 \\
h_3 &= \frac{1-\sqrt{3}}{4\sqrt{2}} & g_3 &= -h_0
\end{aligned} \tag{23}$$

The numerical algorithm of the adaptation process proceeds as the follows: The variables in equation (17) are calculated and plugged into the wavelet algorithm, equations (21) and (22), that in turn compress the data by splitting it into a sequence of scales and sequence of details. This step is the main task performed by the wavelet transform algorithm. Only the sequence of the details is used in equation (20) to calculate the threshold for each sensor. Then, the detail of each sensor is compared with the corresponding threshold for every flow cell and the cell is tagged for refinement whenever a detail of the sensors is greater than its threshold.

## RESULTS AND DISCUSSIONS

In this paper, an open source Cartesian grid generator and flow solver[1] which is beefed up with an algorithm for performing solution–based grid adaptation was

[1] The **CFS**; (**C**)artesian (**F**)low (**S**)olver, is an open source Cartesian grid generator and flow solver created and developed by the first author.

carefully studied in order to gain an in–depth sight of the solution-based grid adaptation process. To verify and evaluate the efficiency of the algorithm two test cases were selected. These two test cases involve the inviscid flow past the well-known NACA0012 airfoil [11]. The program was run for both test cases with the solution-based grid adaptation option switched on. It should be noted here that the number of grid adaptations was restricted to two times only. This was done in order to keep the computational time in reasonable range, also the increase of the obtained accuracy-when more grid adaptations is performed- was not worthy the resulting very high number of emerging cut and flow cells.

The first test case is the computation of the flow past the previously mentioned airfoil with a free stream Mach number of 0.85 and an angle of attack of 1.0 degree. The second test case is similar to the first with the exception that its free stream Mach number is 1.2 and its angle of attack is zero. The first case involves two attached shock waves; a strong one on the upper side of the airfoil, and a weaker one on the lower side of the airfoil, while the second test case is characterized by a detached shock wave in front of the airfoil, and two attached shock waves emanating from the trailing edge of the airfoil.

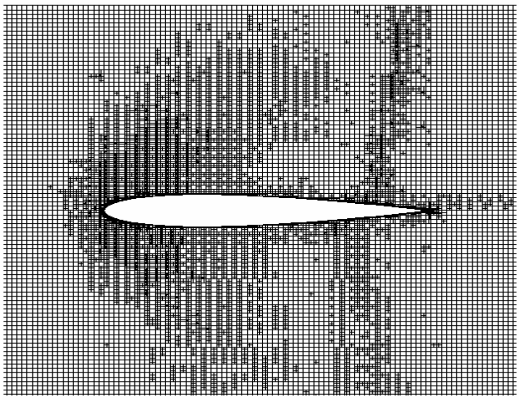

**Figure1: The solution-base adapted grid for the first test case**

As seen in Figure (1), the grid for the first test case was adapted at the regions of high gradients, that is; at the leading edge in vicinity of the stagnation point as well as at the two attached shock waves.

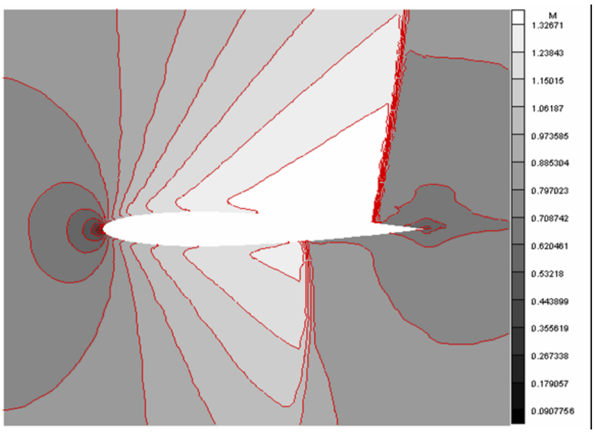

**Figure 2: The Mach number distribution for the first test case**

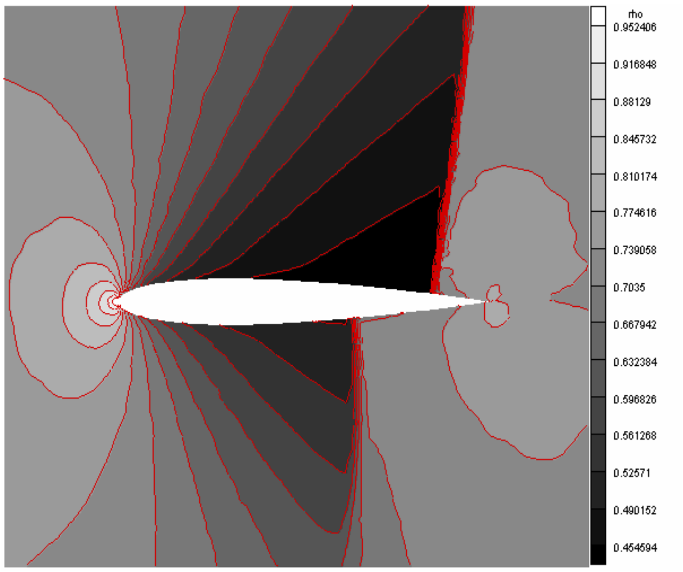

**Figure 3: The density distribution for the first test case**

Figures (2) and (3) show the Mach number distribution and the density distribution for the first test case, respectively. The high resolution obtained here, especially near the shock waves, is of course due to the grid adaptation (grid refinement) at these regions.

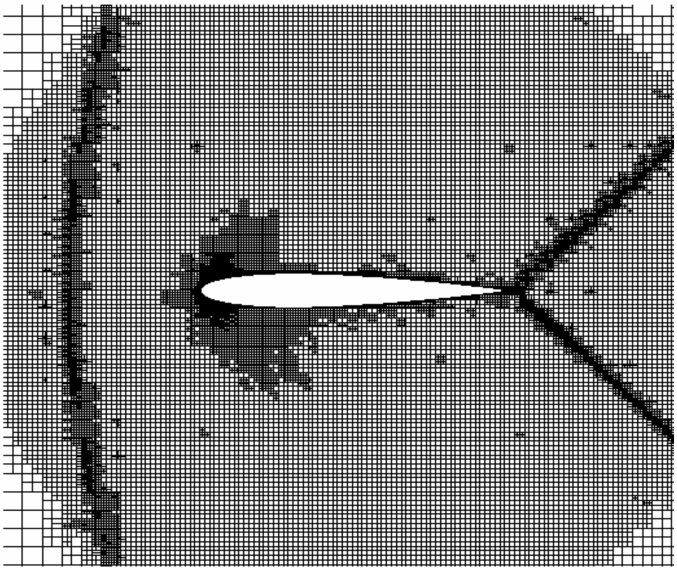

**Figure 4: The solution-base adapted grid for the second test case**

Figure (4) shows that the built -in sensors that were introduced in the introduction, have captured the important flow features of the second test case successfully. In other words, the sensors have successfully captured the detached shock wave and the two attached shock waves as well as the stagnation point.

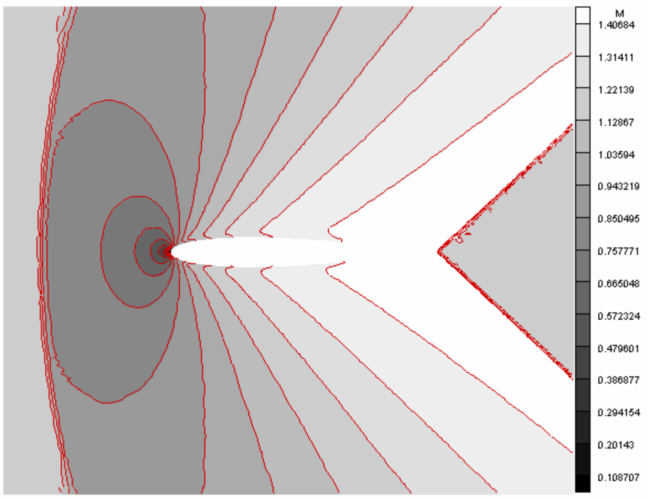

**Figure 5: The Mach number distribution for the second test case**

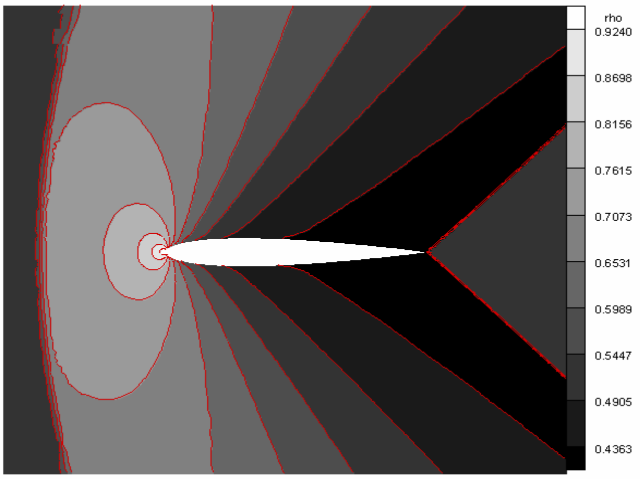

**Figure 6: The density distribution for the second test case**

Figures (5) and (6) show the Mach number and density distribution for the second test case, respectively. The very good resolution obtained, especially in the case of the detached shock wave is again related to grid adaptation. This high resolution was not obtained when the same test case was carried out by [12], since they have deactivated the solution -based grid adaptation option in their calculations.

## CONCLUSIONS

The Adaptation sensors that were incorporated in the solution based grid adaptation algorithm were successful in locating the high gradient regions in the computational domain; however, the used adaptation sensors were selected for computations of inviscid flow calculations. For viscous flow calculations, another sensor(s) should be considered. The Daubechies wavelet has reduced the search process to include only the detail of the given data instead of applying the search process on the whole range of the given data, and this reduction has not affected the accuracy of the search process.

## ACKNOWLEDGMENTS

The first author would like to thank Mr. Ian Kaplan for his comprehensive explanation on the wavelet which is documented and available for public free of charge at www.bearcave.com/misl/misl_tech /wavelet/

## REFERENCES

[1] Wang, J. Z., Wiederhold, G., Firschein, O. , and Wei, S. X. , Content-based Image Indexing and Searching Using Duebechies Wavelet , Int. J. of Digit. Lib. 1, pp. 311-328, 1997

[2] Fares, E. , Meinke, M. , and Schroeder, W., Numerical Simulation of the Interaction of Wingtip Vortices and Engine Jets in the Near Field , AIAA paper, pp. 2000-2222 , 2000.

[3] Opela, Mario, Grobstruktur-Simulation der Interaktion des Nachlaufs eines bewegten Kreiszylinders mit einer Turbinenschaufel , Ph.D. Dissertation (in German), Aerodynamisches Institut Aachen-RWTH AACHEN, 2003.

[4] Hirch, C., Numerical Computation of Internal and External Flow, Vol. 2, John Wiley and Sons, 1992.

[5] Liou, M, S. and Steffen, C. J., A New Flux Splitting Scheme , J. Comp. Phys., 107, pp. 23 -39 1993.

[6] Gaffney, R., Hassan, H., and Salas, M., Euler Calculations for Wings Using Cartesian Grids, AIAA paper, pp. 87 -0356, 1987.

[7] Warren, G. P. , Anderson, W. K. ,Thomas, J. L., and Krist, S. L., Grid Convergence for Adaptive Methods , AIAA paper, 91 -1592-cp, 1991.

[8] Aftosmis, M. J., Upwind Method for Simulation of viscous Flow on Adaptively refined meshes, AIAA J., 32(2), pp. 268-277, 1994.

[9] Lahur, P. R., Nakamure, Y. , Anisotropic Cartesian Grid Adaptation , AIAA paper, pp. 2000-2243, 2000.

[10] Percival, D, B. and Walden, A. T., Wavelet Methods for Time Series Analysis, Cambridge, England Cambridge Univ. Press. 2000.

[11] Pulliam, T. and Barton, J., Euler Computation of AGRAD Working Group 07 Test Cases, AIAA paper, 85 -0018, 1985.

[12] Sultan, Khalid M. and Belgasim, Basim A., Numerical Simulation of the Inviscid Flow Past Airfoils, Journal of engineering Research, vol. 7, April 2008.

## NOMENCLATURE

### Latin

| | |
|---|---|
| $a$ | speed of sound, m/s |
| $C_v$ | specific heat at constant volume, kJ/(kg K) |
| $c$ | Chord length, m |
| $CP$ | Coefficient of pressure |
| $e$ | Specific total energy, kJ/kg |
| $e^*$ | Dimensionless specific total energy $e / a_o^2$ |

| | |
|---|---|
| $L_x, L_y$ | Lengths of the computational domain in x and y directions, m |
| $M$ | Mach number |
| $n_x, n_y$ | Number of cells in x and y directions |
| $P$ | Pressure, kPa |
| $P^*$ | Dimensionless pressure $P/(\rho_o a_o^2)$ |
| $R$ | Gas constant, kJ/(kg K) |
| $T$ | Temperature, K |
| $T^*$ | Dimensionless temperature $T/T_o$ |
| $t$ | Time, s |
| $t^*$ | Dimensionless time $\frac{t \cdot a_o}{c}$ |
| $u$ | Component of velocity vector in x direction, m/s |
| $\hat{u}$ | Specific internal energy, kJ/kg |
| $u^*$ | Dimensionless component of velocity vector in x direction $u/a_o$ |
| $\vec{V}$ | Velocity vector |
| $V$ | Component of velocity vector in y direction, m/s |
| $V^*$ | Dimensionless component of velocity vector in y direction $V/a_o$ |
| $x$ | Horizontal coordinate, m |
| $y$ | Vertical coordinate, m |

**Greek**

| | |
|---|---|
| $\rho$ | Density, kg/m$^3$ |
| $\rho^*$ | Dimensionless density $\rho/\rho_o$ |
| $\theta$ | Angle of attack, radian |
| $\gamma$ | specific heat ratio |

**Subscripts**

| | |
|---|---|
| $o$ | Stagnation condition |
| $\infty$ | Free stream value |
| $fc$ | The nearest flow cell to the cut surface |
| $cd$ | Computational domain |
| $sc$ | Cut surface |

**Superscripts**

| | |
|---|---|
| * | Dimensionless variables |
| $n$ | Solution at time level n |
| $n+1$ | Solution at time level n+1 |