AIAA 92-0443

Simulation of Unsteady Inviscid Flow on an Adaptively Refined Cartesian Grid

Yu-Liang Chiang, Bram van Leer
and Kenneth G. Powell
Department of Aerospace Engineering
The University of Michigan
Ann Arbor, MI USA

30th Aerospace Sciences
Meeting & Exhibit
January 6-9, 1992 / Reno, NV

Simulation of Unsteady Inviscid Flow on an Adaptively Refined Cartesian Grid

Yu-Liang Chiang *

Bram van Leer †

Kenneth G. Powell ‡

The University of Michigan

Department of Aerospace Engineering

Ann Arbor, MI 48109-2140

Abstract

A method for adaptive refinement of a Cartesian mesh and corresponding time-step adaptation, for the solution of the unsteady Euler equations, is presented. In this work, a linear reconstruction of distributions inside cells, and Roe's approximate Riemann solver for interface fluxes, are used. The wave strengths and wave speeds needed for the flux calculation are reused in various ways. In particular, in order to prevent moving discontinuities from running out of fine cells during one global time-step, wave speeds and wave directions are used to predict the region traversed by the waves; these are then flagged for refinement. Moreover, the curvature of one-dimensional wave-strength distributions is introduced as the key quantity in refinement and recoarsening criteria. The numerical results presented show that this method can obtain the same accuracy on the adaptive grid as on a uniform grid with cells as fine as the finest cells of the adaptive grid, at large savings of computing time.

1 Introduction

When solving aerodynamic problems with computational methods we run into the problem of grid generation. Two major questions are:

1. How to create any grid in the presence of complex body shapes;

2. How to get sufficient resolution in places where the action is.

The adaptive Cartesian approach of De Zeeuw and Powell [1] answers these questions by allowing irregular cells, cut off by the body from Cartesian cells, and by embedding refined cells wherever needed to resolve geometric and/or flow details. The code developed in [1] is for two-dimensional steady flow.

The present work extends the approach of De Zeeuw and Powell to unsteady flow. A code for unsteady flow has been developed that responds to adaptive spatial refinement by time-step adaptation, i.e., by using many small time steps in refined regions in order *to* match a single large time step used in coarse cells. In this way, explicit time-marching can be used throughout the computational domain and temporal accuracy is preserved. unlike in implicit methods, where only *stability* is maintained. The present code achieves second-order accuracy in time, which is a non-trivial extension.

2 Time Discretization

It is generally agreed upon that second-order accuracy in space and time is a minimum requirement for an Euler (or Navier-Stokes) code to be useful in efficiently solving problems of transient flow. Some codes for transient (PPM) and for steady flows (e.g., CFL3D [2]) use spatial

*Member AIAA

†Professor,Member AIAA

‡Assistant Professor, Member AIAA

differencing techniques that would lead to third-order spatial accuracy if there were only one space dimension. Full third-order accuracy in multi-dimensionai space is rarely achieved (see, however, Barth [3]), and time-marching has never gone beyond second-order accuracy.

In the present work we have chosen for explicit multi-stage marching in time. This technique is routinely used for steady-state calculations and therefore has hardly been analyzed regarding temporal accuracy. We the efore include a discussion on the use of the Fourier transform in searching for time-marching schemes that are in some sense optimal. This technique is applied to a family of three-stage convection schemes, of at most third-order accuracy. Furthermore, we investigated carious differencing formulas with regard to their accuracy near an interface between regions of coarse and fine cells [4].

2.1 A Design Criterion for Time-Accurate Multi-Stage schemes

When selecting a multi-stage scheme for time marching it is best to start from a family of schemes with the same order of accuracy, and then select the most desirable one according to some design criterion. For instance. if second-order time accuracy is to be achieved, at least two stages are needed, and preferably three, for some freedom of choice. The design criterion must take into account u-hat class of problems the scheme will be applied to, in particular, whether the solutions sought will be smooth or discontinuous.

There is a very useful tool based on the Fourier transform, which actually takes into account the spectrum of the initial-value distribution. This method is described by Wesseling [5]; it measures the total L_2-error a linear convection scheme makes in convecting a spatial distribution with a specific frequency content. Through Parseval's theorem, the numerical error integrated over the space domain is transformed to an integral over the frequency domain. In many cases this integral can be obtained analytically, allowing analytical minimization. When optimizing a scheme for application to problems of *discontinuous* flow, the initial-value distribution for which the integral error is minimized, is chosen to be a *step function*. This means that the errors in frequency space are weighted with the inverse of the frequency. A scheme thus optimized will produce minimal spurious oscillations near a discontinuity.

According to Parseval's equality, the truncation error $u^{n+1}(x) - u^{n+1}_{\text{exact}}$ satisfies the following equation:

$$\int_{-\infty}^{\infty} |u^{n+1}(x) - u^{n+1}_{\text{exact}}|^2 dx = 2\pi \int_{-\infty}^{\infty} |\hat{u}^n(\beta)|^2 |g(\beta,\nu) - g_{\text{exact}}(\beta,\nu)|^2 d\beta; \quad (1)$$

here β denotes frequency, ν the Courant number, arid g the amplification factor of the scheme. Comparison of the scheme's amplification factor with the exact amplification factor, in order to get an idea about the accuracy of the scheme, is commonplace; Equation 1, though, suggests that the error in the amplification factor should be weighted with the spectrum of the initial values. Define the weighted norm $\|g(\beta,\nu) - g_{\text{exact}}(\beta,\nu)\|$ by

$$\|g(\beta,\nu) - g_{\text{exact}}(\beta,\nu)\|^2 = \int_{-\infty}^{\infty} \rho(\beta) |g(\beta,\nu) - g_{\text{exact}}(\beta,\nu)|^2 d\beta, \quad (2)$$

where the weight function ρ equals to the square of the modulus of the Fourier transform of the distribution at time t^n. For a step function the weight function will be $\rho(\beta) = \frac{1}{\beta^2}$. For the family of convection schemes studied in [5], based on a five-point stencil at the initial time level, the selection procedure based on minimizing this norm yields an *apwind-biased* scheme, confirming the reputation of such schemes to represent moving or steady discontinuities with reduced oscillations.

2.1.1 An example

Consider the first-order upwind-differencing operato:, with Fourier transform

$$z = -\nu(1 - e^{-i\beta}).$$

All three-stage schemes with second-order temporal accuracy can be represented by the amplification factor

$$g(\beta,\nu) \equiv g(z) = 1 + z + \frac{z^2}{2} + \alpha z^3,$$

where α is a free parameter to be determined by the minimization process, and

$$g_{\text{exact}}(\beta,\nu) = e^{-i\beta}.$$

The error norm is

$$\int^{\infty} |g(z) - e^{-i\beta}|^2 \frac{1}{\beta^2} d\beta$$

$$\int |1 + z + \frac{z^2}{2} + \alpha z^3 - e^{-i\beta}|^2 \frac{1}{\beta^2} d\beta$$

$$= 6\alpha^2\nu^6 + \alpha(-3\nu^5 + 3\nu^4 - 3\nu^3 + 3\nu^3|1-\nu| + \frac{\nu^4}{2} - \nu^3 + \nu^2 + (-\nu^2+\nu)|1-\nu|. \quad (3)$$

Plots of this integral against ν for two values of α, corn-puied with the above formula or by numerical integration, are shown in Figure 1.

It is seen in Figure 1 that $\nu = 1$ is a turning point for three-stage schemes; beyond this point the truncation

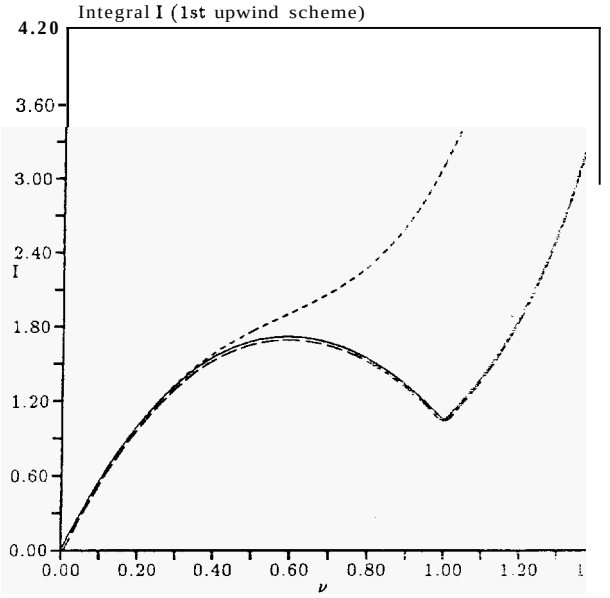


Figure 1: L_2-error committed after one time-srep when convecting a step-function. The scheme used includes first-order upwind differencing and third-order three-stage time-marching. The error is given in the form of an integral **I** over all frequencies, which depends on the Courant number ν. This simple case served to verify that the numerically evaluated integral (dashed line) was close to its exact value (solid line). For $\alpha = 0$ (dotted line; two-stage scheme) the error increases monotonically.

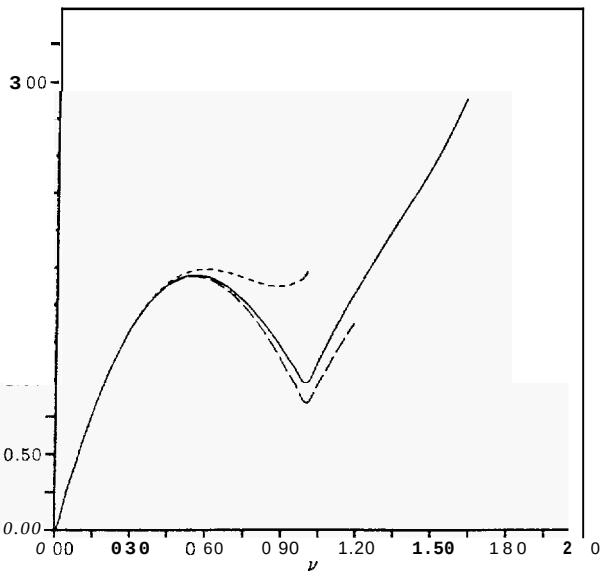


Figure 2: L_2-error as a function of Courant number, for three different three-stage schemes and third-order spatia! differencing. The error is plotted only for the stable range of these scheme. The value $\alpha = 0.072$(dashed line) yield the lowest average error; $\alpha = \frac{1}{6}$ (solid line) yields third-order time accuracy; $_{cy} = \frac{1}{3}$ (dotted line).

error is increasing rapidly. This suggests that, in practical applications, one should use $\nu \leq 1$. Note that the above analysis is for a scheme with only first-order spatial accuracy. The analytical evaluation of the integral I for third-order upwind-biased differencing (parameter value $\kappa = \frac{1}{3}$; see [6]) is much more tedious, but numerical integration gives a reliable result. Plots of the integral for various values of α are shown in Figure **2.**

To facilitate the comparison of schemes, we may eliminate the dependence of the integral on ν by integrating over the stable range of ν. That is, we minimize

$$I_{\text{ave}} = \frac{1}{\nu_{\max}} \int_0^{\nu_{\max}} \int_{-\infty}^{\infty} |\hat{u}^n|^2 |g(\beta,\nu) - g_{\text{exact}}(\beta,\nu)|^2 d\beta d\nu. \tag{4}$$

Plots of I_{ave} and $\nu_{\max}$ versus $_{cy}$ are shown in Figure *3.* The minimum error occurs for $_{cy} = 0.072$, but this is mainly so because, for this value of α, $\nu_{\max}$ is not much greater than 1. With increasing $_{cy}$ the stability region grows rapidly, causing a rise in I_{ave} up to $\alpha = 0.15$. For larger values of α the error decreases, which again is mostly due to $\nu_{\max}$ decreasing toward 1. Looking back at Figure 2 we conclude that the third-order scheme $(\alpha = \frac{1}{6})$ is preferable to the optimal scheme: its error is hardly greater than the error of the optimal scheme, it offers a much larger stability range, and therefore increased robustness, and it is formally third-order accurate.

In the present code a two-stage second-order algorithm is implemented, which, in combination with adaptation, already leads to a complex sequence of steps. In principle, the three-stage method can be programmed in the same manner; whether this is worth-while remains to be seen. It may be argued that the local mesh refinement is a more efficient way to increase resolution by the higher-order interpolation in space and time.

2.2 Time Discretization and Mesh Refinement Combined

The spatial embedding technique of De Zeeuw and Powell employs the quad-tree data structure: one parent cell generates four child cells (Figure **4).** The spatial discretization follows the reconstruction/evolution approach: in each cell gradients of flow quantities are formed (by evaluating a contour integral, see Figure 5); these are used to evaluate states at the cell boundaries. Interface fluxes are then computed from the two different states found on opposite sides of the interface, using Roe's approximate

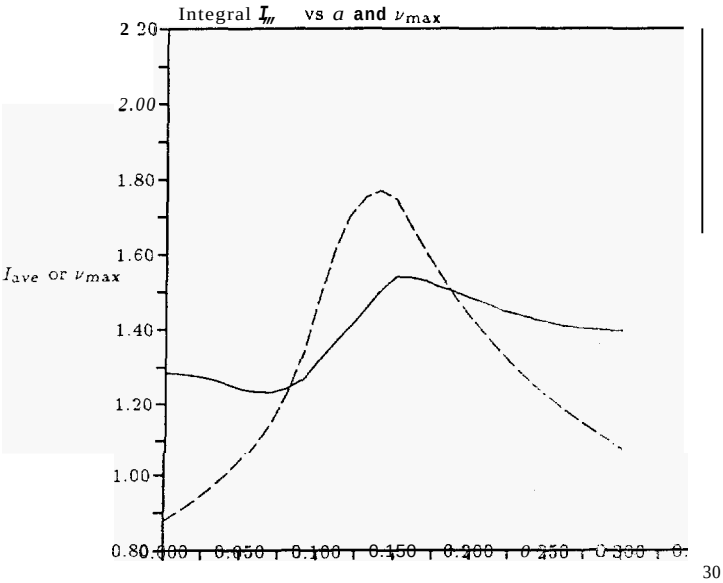


Figure 3: Maximum stable Courant number (dashed line) and average of L_2-error (I_1) over the stable Courant-number range (solid line), for three-stage schemes with free parameter α. Spatial differencing is third-order ($\kappa = \frac{1}{3}$); the amplification factor of the time-marching scheme is $1 + z + \frac{1}{2}z^2 + \alpha z^3$. Note that the average error increases sharply with the stability range.

Riemann solver; for details see [1].

The time-adaptation procedure introduces halving of the time step for every level of spatial refinement; in consequence, for each level of refinement the number of times a cell is updated doubles. Updating starts with the coarsest cells and cascades down to the finest cells. In order to simplify the spatia1 discretization and the update procedure, the spatial grid is constrained such as to always have at least two cells of the same size adjacent to each other in any direction (horizontal, vertical, diagonal).

The time-marching scheme is a simple two-stage procedure, based on the midpoint integration rule:

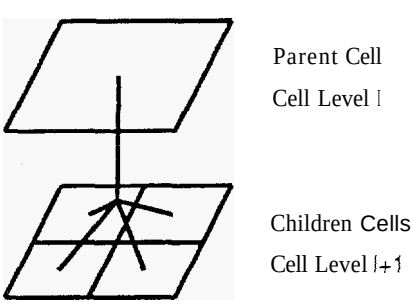


Figure 4: Parent/Children Relationship

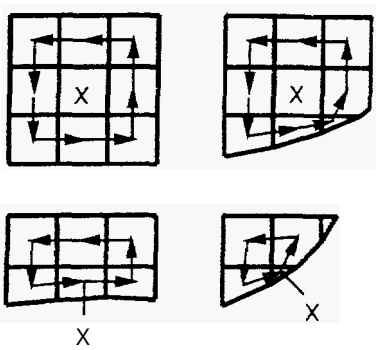


Figure 5: Normal And Special Paths

$$
\begin{aligned}
u^{(0)} &= u^n \\
u^{(1)} &= u^{(0)} + \frac{\Delta t}{2} Res(u^{(0)}) \\
u^{(2)} &= u^{(0)} + \Delta t Res(u^{(1)}) \\
u^{n+1} &= u^{(2)}.
\end{aligned} \tag{5}
$$

According to Shu and Osher [7], this time-marching scheme is not Total-Variation Diminishing (TVD); their two-stage TVD scheme is based on trapezoidal integration. When updating a fine cell adjacent to a coarse cell with the TVD scheme, the prediction step is problematic: by matching the time level in the coarse cell, the fine cell might go unstable. A compromise would be to use the mid-point rule in this coarse ceii; this has not yet been tested.

Scheme *(5)* is used at any level of refinement; the time-step used at cells of level l, refined $l-1$ times, is $\Delta t/2^{l-1}$, where Δt is the global time-step used. Now suppose we are about to update I-level cells by a time-step Δt_l. Owing to our restriction on the local variation in cell size, only two other levels of refinement will play a role in the update: $l+1$ and $1-1$.

2.2.1 Interaction at $(l, l+1)$ Boundary

At first, we consider the boundary between the cell of level l and a possible neighbor at level $l+1$; here, as at any other cell boundary, we *must* evaluate fluxes. In the two-stage method we need to first carry out the predictor stage, which achieves only first-order accuracy in time. The initial values, $u^{(0)}$, are used to evaluate the gradients in all cells, which, in turn, are used to reconstruct the "right" and "left" face values. Subsequently, one can compute the fluxes based on Roe's [8] approximate Riemann solver on all child-cell faces (level $l+1$) of the fine-coarse boundary. Conservation dictates we must sum these child-cell fluxes to get the flux across the full parent-cell boundary (level l). Now the predictor step at level l can be completed.

For the corrector-stage calculation, we need new fluxes at the cell boundary; therefore, we need new input values for the flux function, hence, new gradient values. In order to make it possible to evaluate the gradients not only in the I-level cells themselves but also in the abutted $(l-1)$-level cells, some $(l+1)$-levels near the boundary must be updated by a step $\Delta t_{l+1}(=\frac{1}{2}\Delta t_l)$. After obtaining the gradients in these cells we can, again, reconstruct the right and left face values, and compute the fluxes needed for the correction stage in the cells at level l. The whole procedure is shown in Figure 6 – Figure 9. These, however, are not the final fluxes; in order to achieve conservation in time, the fluxes in the l-level cells adjacent to the $(l, l+1)$ boundary must undergo a correction after the $(l+1)$-level cells have been fully updated.

2.2.2 Interaction at $(l, l-1)$ Boundary

To make it possible to update l-level cells, it is assumed that we have already obtained, in the cells at level $l-1$, first- and second-order-accurate solutions at Δt_{l-1} and first-order-accurate solutions at $\frac{1}{2}\Delta t_{l-1}$. This dictates the order in which the cells of different levels are treated. Having these solutions in the coarser cells one may, by interpolation, get first- and second-order accurate values at any time between t'' and $t^n+\Delta t_{l-1}$. In particular, by using linear interpolation one can get first-order-accurate solutions after time-steps $\frac{1}{4}\Delta t_{l-1}(=\frac{1}{2}\Delta t_l)$ and $\frac{3}{4}\Delta t_{l-1}(=\frac{3}{2}\Delta t_l)$; these are needed to compute the gradients for corrector-stage use in I-level calculations. By using quadratic interpolation, one can get second-order-accurate solutions at $t^n+\frac{1}{2}\Delta t_{l-1}(=At,)$, which can be used as the initial conditions for predictor-stage use in the second application of scheme 5 to l-level cells. These interpolation procedures are illustrated in Figure 10.

2.2.3 Conservation in Time

In the above update procedure, the fluxes actually used occur at the corrector stage rather than the predictor stage. At an $(l, l-1)$ boundary, however, the corrector-stage fluxes at $t''+\frac{1}{2}\Delta t_l$ and $t''+\frac{3}{2}\Delta t_l$, used in updating the l-level cell, are not used in the provisional update of the $(l-1)$-level cell. Conservation requires that the same fluxes be used in both cells; hence, from the detailed fluxes used for the l-level cells we must construct parent fluxes by averaging and apply these to the $(l-1)$-level cell. This amounts to a correction of the solution at $t^n+\Delta t_{l-1}$ in the coarse cell. The procedure is illustrated in Figure 11.

2.2.4 Computational Priority of Different Levels

As mentioned above, before we can start to integrate in l-level cells, we should already have first- and second-order-accurate solutions at Δt_{l-1}, and first-order-accurate solutions at $\frac{1}{2}\Delta t_{l-1}$; that is, $(l-1)$-level cells should already have completed the predictor-corrector stage. The update procedure then cascades from lower-level cells to higher-level cells. Once the predictor-corrector scheme has been completed in $(l+1)$-level cells, one returns to the l-level cells for the conservation correction, and so on.

3 Time-Marching in Cut Cells

If the geometry is just slightly complex or its dimensions are unfavorable, the square cells of an Cartesian giid may be cut by bodies. Instead of reducing the time step, we can combine the cut cell with an uncut neighbor to form a larger cell and determine its evolution as part of the evolution of the full cell. The full update procedure for cut cells is as follows:

1. compute the gradients in the cut cell just as in any uncut cell;
2. compute the area average of the gradient,

$$G_{\text{avg}} = \frac{\sum_{i=1}^{n} G_i A_i}{\sum_{i=1}^{n} A_i}, \tag{6}$$

   where n is the total number of merged cells and an uncut:
3. compute the residuals in the cut cells just as in any uncut cell;
4. compute the area average of the residual;

$$\text{Res}_{\text{avg}} = \frac{\sum_{i=1}^{n} (\frac{\partial \mathbf{U}}{\partial t})_i A_i}{\sum_{i=1}^{n} A_i}; \tag{7}$$

5. update the merged cells;
6. update the average gradient in the merged cells;
7. use the average gradient to reconstruct separate cell averages in the cut-cell and uncut-cell portion of the merged cell.

The whole procedure is illustrated in Figure 12.

4 Automatic Grid Adaptation

An adaptive grid may be refined or recoarsened 2s dictated by the amount of detail in the flow. The refinement/recoarsening criterion used in this work is based on the curvature of one-dimensional wave-strength distributions; the wave strengths are a by-product of evaluating Roe's flux function. In two dimensions there are four families of waves for each coordinate direction, leading to eight different curvature values in each cell. If a particular curvature in the cell is above a predetermined fraction of the

maximum for its own family, the cell is flagged for refinement, if it drops below another, lower threshold, it is flagged for recoarsening.

The wave information is used once more for the prediction of the area where cells need to be refined; for this prediction the wave *speeds*, including their sign, enter. Wave speeds and propagation directions are used to estimate how far strong waves (identified by the wave strengths) will propagate during the next global time step: the cells to be traversed by these waves are flagged for refinement. The prediction procedure, illustrated by Figures 13 - 15, is as follows:

1. use the wave speed and the wave direction *to* predict the number, (N_x, N_y), of fine cells (l-level) in x- and y-directions for the highest level (l-level) cells containing strong waves; $(0, 0)$ means *that* the waves in this cell are not strong enough to implement the fine-cell prediction technique.

2. use the number (N_x, N_y) to construct a triangle or a line (degenerating from the triangle if either $N_x = 0$ or $N_y = 0$) and then flag the cells covered by any part of the triangle;

3. refine all flagged cells and use the smoothing procedure, (see [4]) to eliminate undesirable features in the resulting mesh.

4. apply steps **2–3** recursively until l-level cells cover all the triangles.

In this way we avoid the loss of resolution of important flow features incurred when these run out of a refined region. For convenience, we name the prediction method "dynamic" and the method without prediction "static" in this paper. The dynamic refinement method has become one of the nicest features of the present adaptive-mesh code. It is very effective, as the next section will show, and surprisingly, uses only 1% of the total computing time.

In spite of its complexity, the prediction method has a flaw: it is based strictly on waves traveling in the grid directions. Thus, a steady oblique shock wave in supersonic flow is not detected as such, but is described by a combination of waves traveling in the grid directions at appreciable speeds. The prediction algorithm flags a band of cells for refinement, but during the next time step the wave does not move, and the refined cells are recoarsened. The resulting cycles of unnecessary refining and recoarsening may be avoided by the use of a multi-dimensional wave model such as that of Rumsey et al. [9, 10]; for efficiency this should also be used in the flux function. As this flux function still lacks the robustness of the flux function based on grid-aligned waves, we have chosen another method to detect steady waves: we check the value of the residual in each cell. If wave strengths are high but the residual is small, there must be cancellation of waves, and no pre-refinement is done.

5 Numerical Results

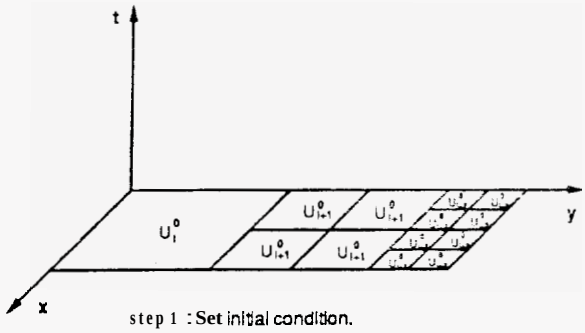
The shock-tube problem is a popular test-case for algorithms intended for solving unsteady flow problem. Figure 16 shows the solution of the one-dimensional shock-tube problem with the two-dimensional code. No prediction of wave motion was used in flagging cells for refinement. For comparison, the solution on a uniform grid of the finest cells is also shown; the accuracy of the solation on the adaptive grid is disappointing. The reason is readily discovered upon inspection of the grid: the important flow features to be resolved, such as the right-running shock, tend to move out of the refined grid in the coarse of the full time-step. *On* the basis of this result it was concluded that spatial refinement ought to be based on the predicted motion of flow features that need to be resolved Figure 17 shows the solution of the Riemann problem on a grid adapted in anticipation of the motion of flow features. The improvement in accuracy is dramatic: the solution is essentially the same as for a uniformly fine grid. Furthermore, the results indicate that according to our refinement criterion based on curvature, the solution inside the expansion wave is partly regarded as smooth: it need not be resolved by the highest-level cells.

The second test case is uniform supersonic flow led through a channel with a forward-facing step. Figure 18–20 show results at time= 0.5 and 4.0 for a uniform grid and an adaptive grid; the uniform-grid cells are everywhere as fine as the highest-level cells of the adaptive grid. There is very little difference between the two solutions, but only one-third CPU time is spent *on* grid-adaptive calculation.

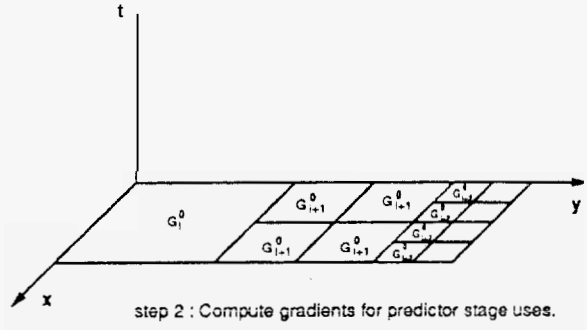
The next figures show how, in the two-dimensional calculation, the main shock moves out of the refined region (Figure 21) unless wave-speed-based refinement is used (Figure 22).

Nest, Figure 23 shows uniform- and adaptive-grid solutions for an axi-symmetric flow problem also solved on an adaptive grid by Quirk [11], namely, that of a shock wave leaving a barrel. Again, there is very little difference between the solutions, but the savings are huge. Figures 24 and 25 show that the fine-cell clusters agree well with the flow features.

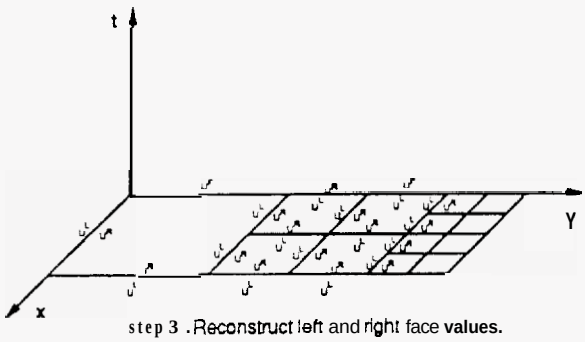
For validating our treatment of cut cells, the case of uniform supersonic flow through a channel with a forward-facing step is reused. In order to compare the results with the previous ones, only the cells along the vertical side of the step are cut by the body; the cells on the top side of the step is still aligned with the body. Two cases of extremely different cut-cell area ratio: defined by the area ratio of a cut cell to an uncut cell of the same level, are chosen: $A_{ratio} = 0.5$ and $A_{ratio} = 0.01$. We reduced the time step to obtain solutions in the first case without merging of cut cells, *for* comparison of the solution with the cut-cell-merging method. Figures 26 and 27 show the results obtained with cell-merging and with a reduced time-step.



step 1 : Set initial condition.

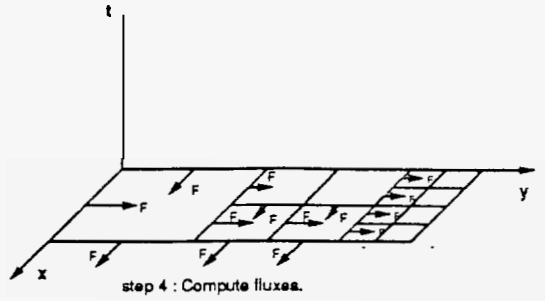


step 2 : Compute gradients for predictor stage uses.

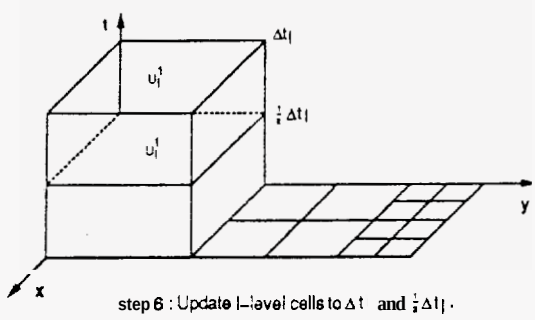


step 3 . Reconstruct left and right face values.

Figure 6: Procedure for lowerer-level cells nest to boundary.

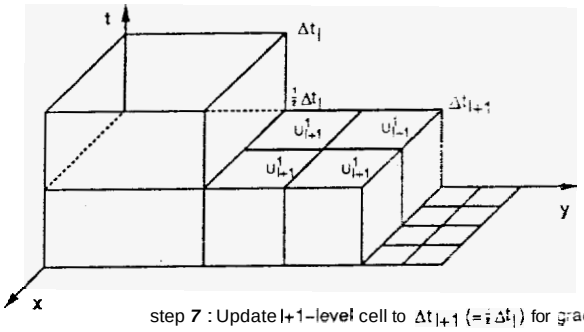


step 4 : Compute fluxes.

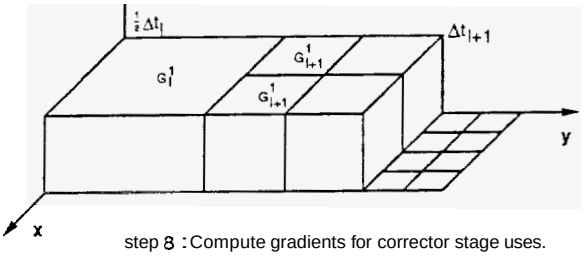


step 6 : Update l–level cells to Δt and $\frac{1}{2}\Delta t_l$.

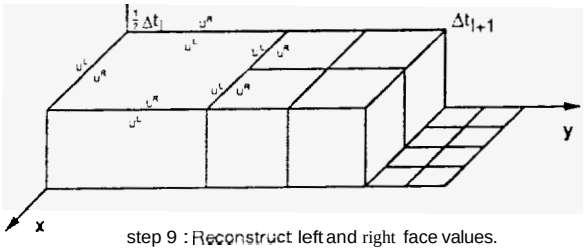
Figure 7: Continued.



step 7 : Update l+1–level cell to Δt_{l+1} $(=\frac{1}{2}\Delta t_l)$ for gradient calculations.

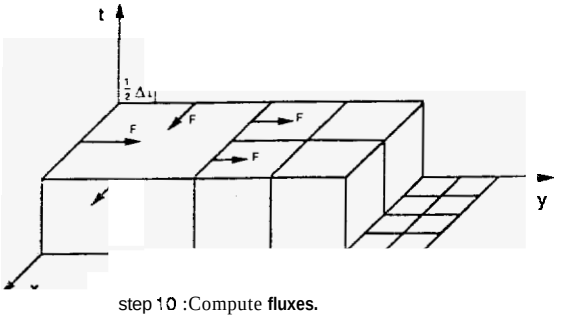


step 8 : Compute gradients for corrector stage uses.

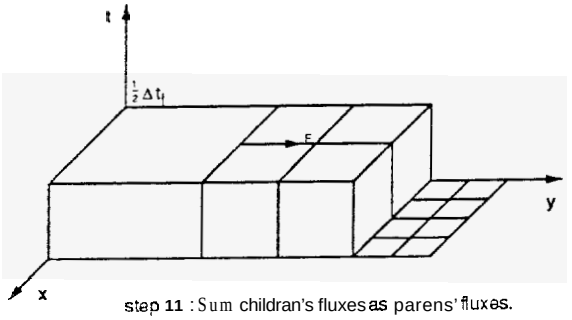


step 9 : Reconstruct left and right face values.

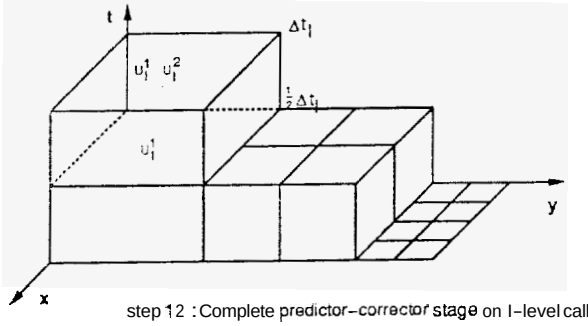
Figure 8: Continued.



step 10 : Compute **fluxes.**



step 11 : Sum children's fluxes as parens' fluxes.



step 12 : Complete predictor–corrector stage on l–level calls.

Figure 9: Conclusion.

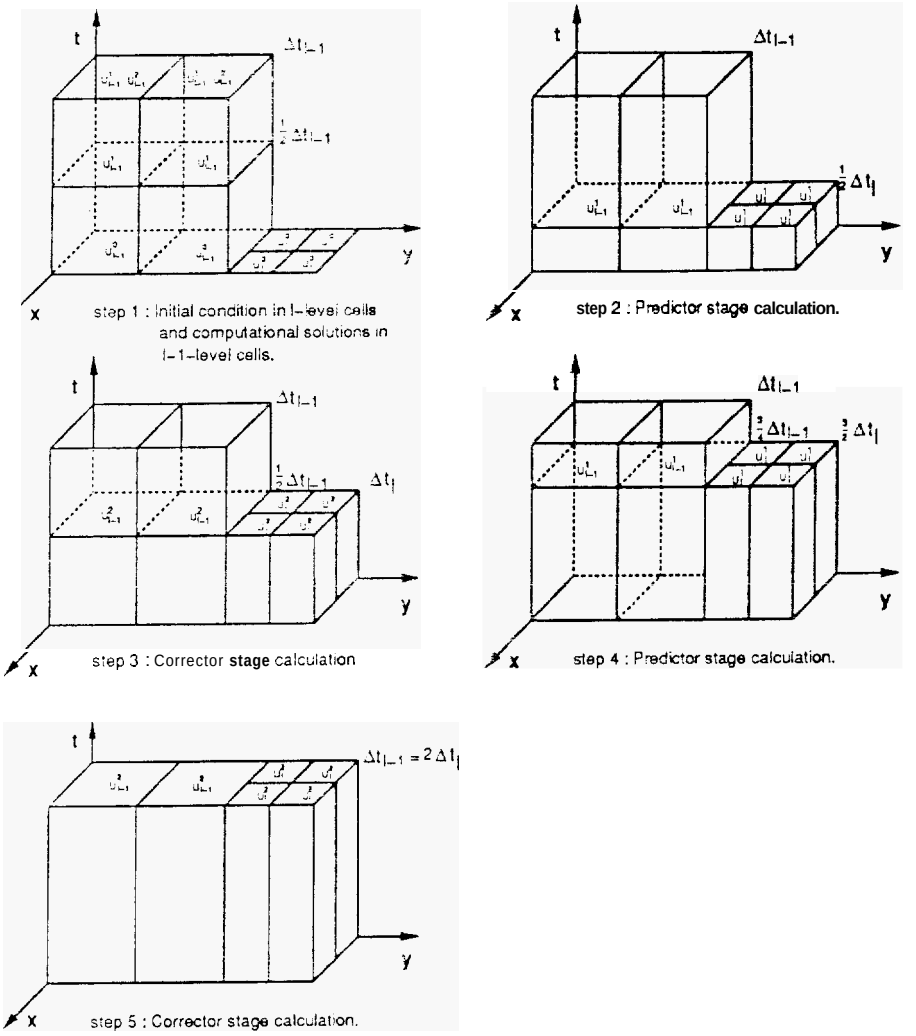


Figure 10: Procedure for higher-level cells next to boundary.

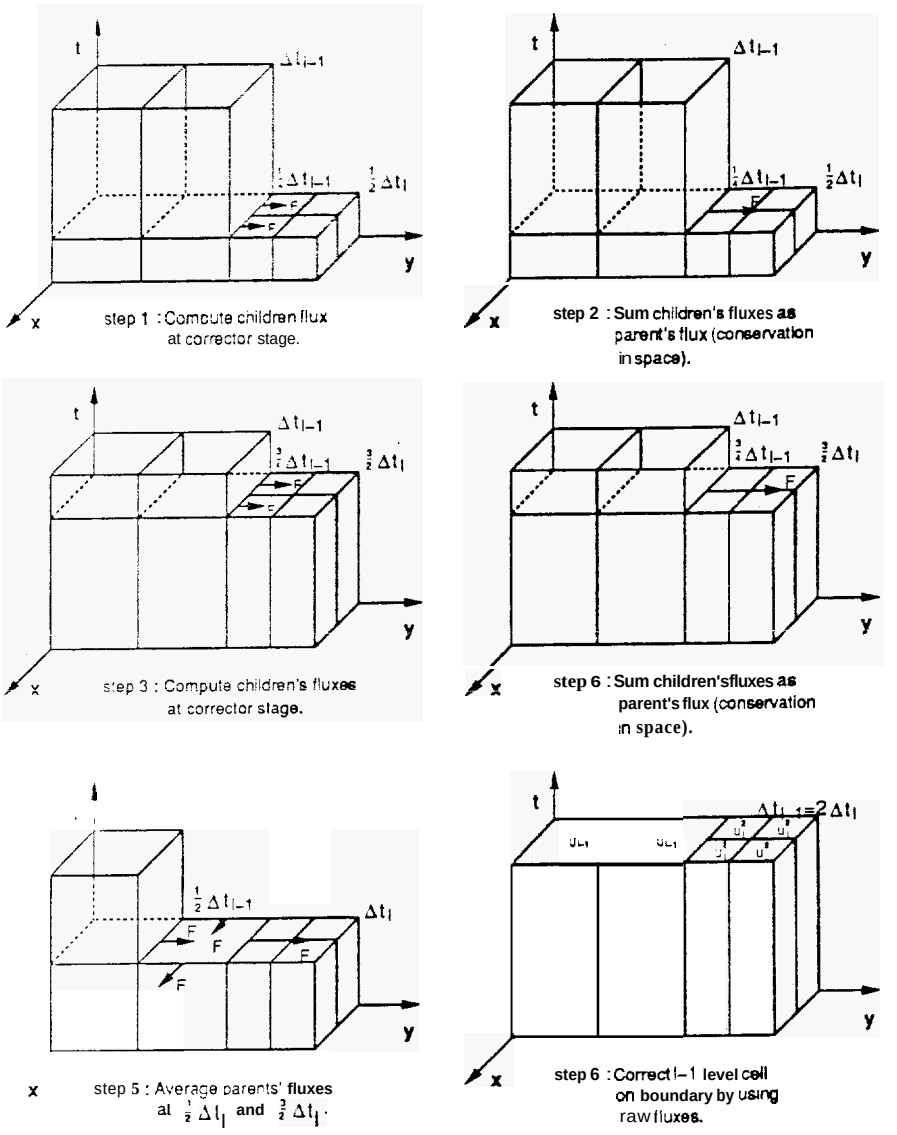


Figure 11: Conservation in time in the boundary cells.

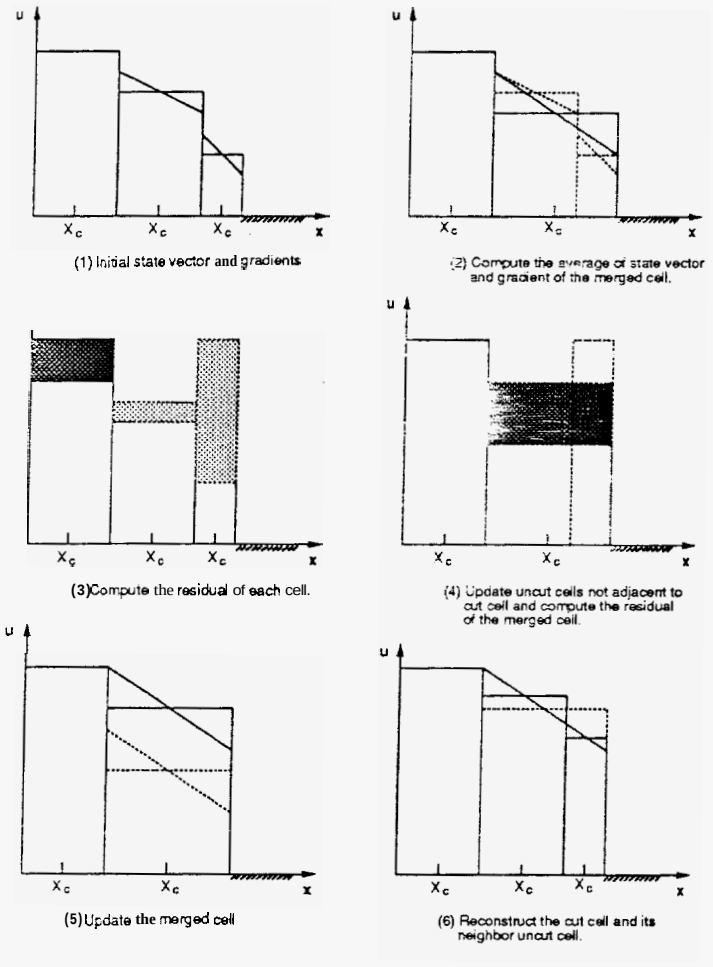


Figure 12: The procedure of updating a merged cell.

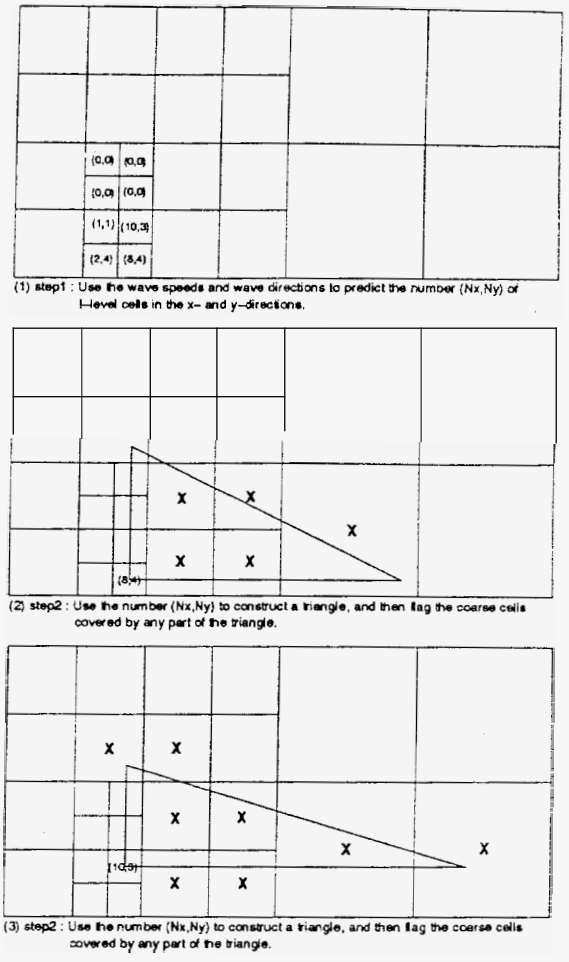


Figure 13: An example of predicting cells of l-level by wave speeds and wave directions

The figures show little difference. Finally, using the cell-merging method with regular time steps in the case of very tiny cut cells ($A_{ratio} = 0.01$) did not create any stability problem, as seen from the result in Figure 28. More numerical resclts are presented in [4].

6 Conclusions

In this paper, we study two major approaches to efficiently and accurately resolve the flow features in unsteady-flow calculation:

1. creating some grid capable of resolving small details of the flow;

2. implementing a high-order-accurate scheme in time and space on the unstructured grid.

The first approach is to apply the technique of self-adaptive mesh refinement to a Cartesian mesh, and use a corresponding time-step adaptation. A process has been developed whereby the computational grid automatically adapts to the flow features that require high resolution. Furthermore, the strategy of time-adaptation, i.e., using many small time-steps on the finer-level grid as compared to one large step on the coarsest grid level, makes the technique of the adaptive mesh refinement achieve a high efficiency.

Regarding the second approach, a second-order-accurate method based on upwindbiased differencing has been applied in our calculations. For time marching, for achieving second-order-accurate solutions in time, a two-stage method is employed. The wave strengths and wave speeds needed for the flux calculation are reused in various ways. In particular, the curvature of one-dimensional wave-strength distributions is introduced as the key quantity in refinement and recoarsening criteria. Furthermore, in order to prevent moving discontinuities from running out of fine cells during one global time-step, wave speeds and wave directions are used to predict the region traversed by the waves; these are then flagged for refinement. This dynamic refinement method is very efficient and is

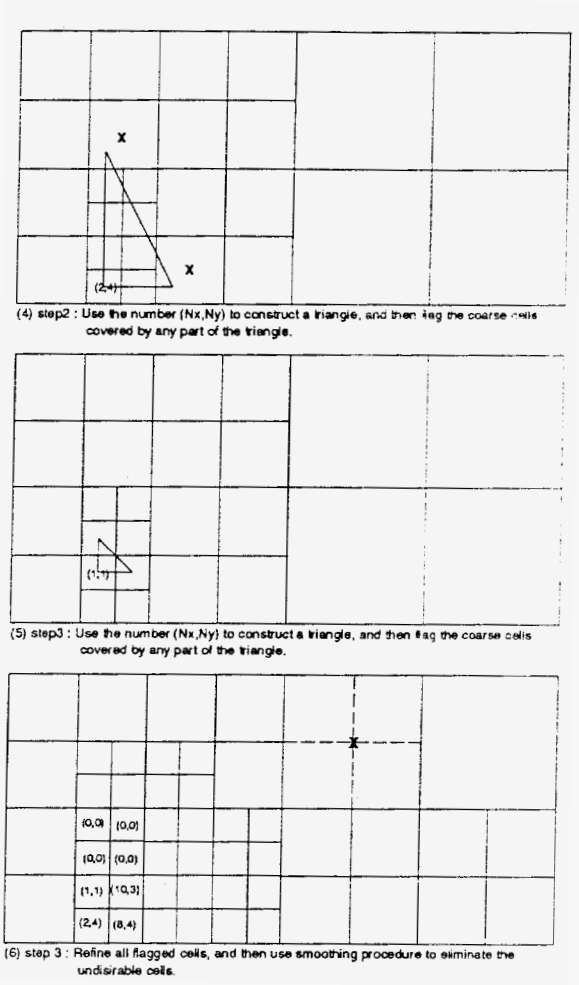


Figure 14: An example of predicting cells of l-level by wave speeds and wave directions (continued).

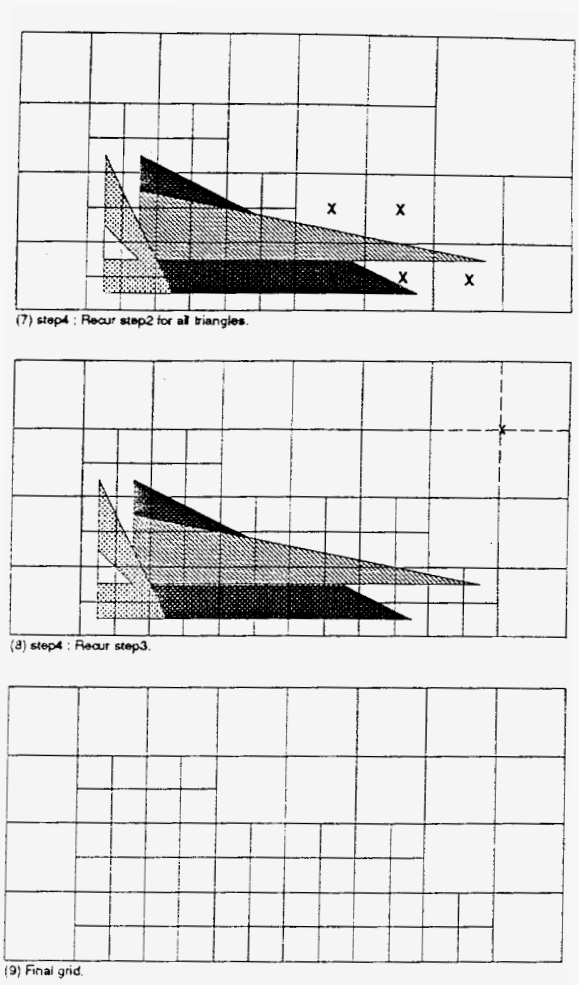


Figure 15: An example of predicting cells of I-level by wave speeds and wave directions (conclusion).

one of the most attractive features of our approach

The combination of techniques can accurately resolve flow features not only in regions of smooth flow for example, inside an expansion fan, but also in a region full of discontinuities. The numerical results show that our method can achieve the same accuracy on the adaptive grid as on a uniform grid with cells as fine as the finest cells of the adaptive grid, at large savings of computing time.

For full details the reader is referred to the first author's thesis [4].

References

[1] D. De Zeeuw and K. G. Powell, "An adaptively-refined Cartesian mesh solver for the Euler equations." **AIAA** 10th Computational Fluid Dynamics Conference. 1991.

[2] W. K. Anderson, J. L. Thomas, and B. van Leer, "A comparison of finite volume flux vector splittings for the Euler equations," *AIAA Journal*, vol. **24,** 1985.

[3] T. J. Barth and P. O. Frederickson, "Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction," AIAA Paper 90-0013, 1990.

[4] Y.L. Chiang, *Simulation of Unsteady Inviscid Flow on an Adaptively Refined Cartesian Grid.* PhD thesis. University of Michigan, 1992.

[5] P. Wesseling, "On the construction of accurate difference schemes for hyperbolic partial differential equations," *Journal of Engineering Mechanics,* vol. 7, 1973.

[6] B. van Leer, "Cpwind-difference methods for aerodynamic problems governed by the Euler equations, in *Large-Scale Computatzons in Fluid Mechanics, Lectures in Applied Mathematics*, vol. 22, 1935.

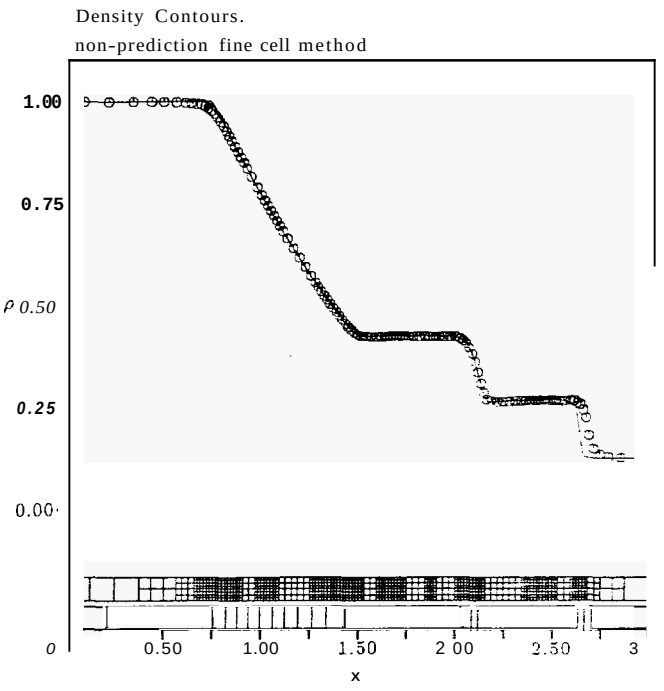


Figure 16: Solution of a shock-tube problem using space- and time-adaptation. The spatial refinement is carried out according *to* the "static" criterion.

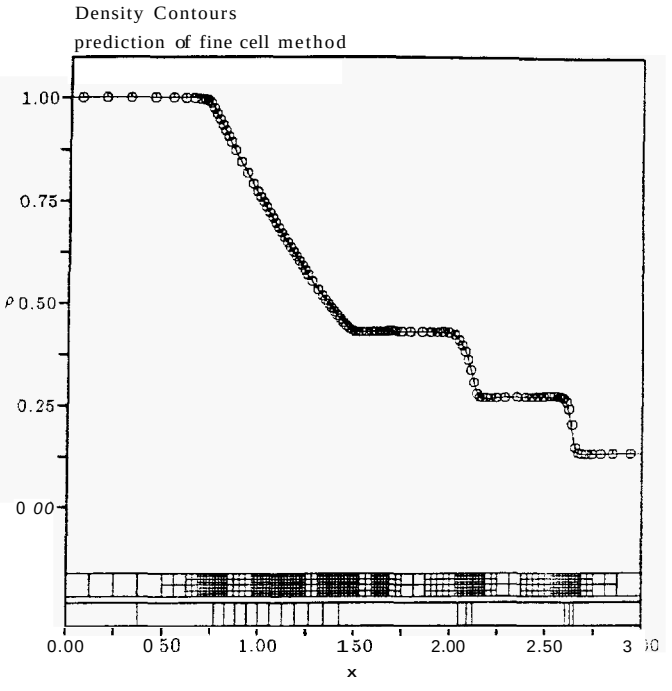


Figure 17: Solution of a shock-tube problem using space- and time-adaptation. The spatial refinement is carried out according to "dvnamic" criterion.

[7] C. W. Shu and S. Osher, "Efficient implementationof essentially non-oscillatory shock capturing schemes." ICASE Report 87-33, 1987.

[8] P. L. Roe, "The use of the Riemann probiem in finite-difference schemes," *Lecture Notes in Physics,* vol. 141, 1980.

[9] C. L. Rumsey, B. van Leer, and P. L. Roe, "A grid-independent approximate Riemann solver with applications to the Euler and Navier-Stokes equations." To be Presented at the 29th AIAA Aerospace Sciences Meeting, Reno, Nevada, 1991.

[10] C. L. Rumsey, B. van Leer, and P. L. Roe, "Effect of a multi-dimensional flux function on the monotonicity of Euler and Navier-Stokes computations." Abstract Submitted to the 10th Computational Fluid Dynamics Conference, 1991.

[11] J. Quirk, *An Adaptive Grid Algorithm* for *Computational Shock Hydrodynamics.* PhD thesis, Cranfield Institute of Technology, 1991.

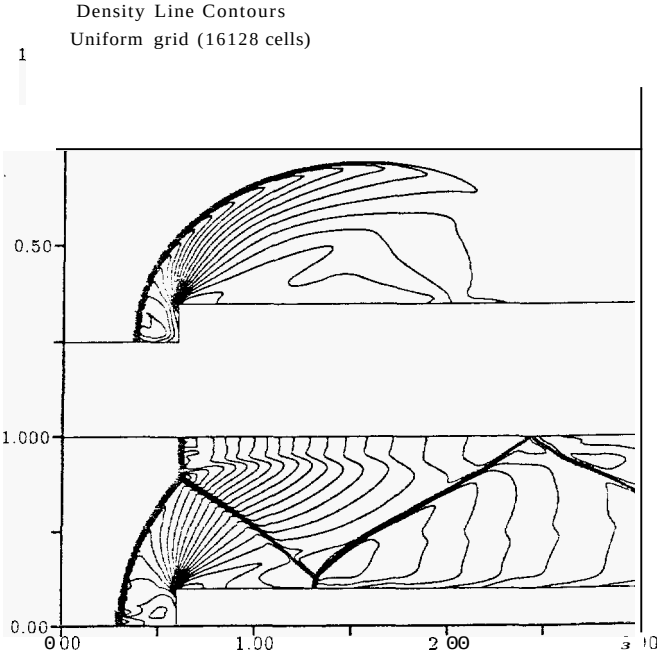


Figure 18: Uniform-grid solution.

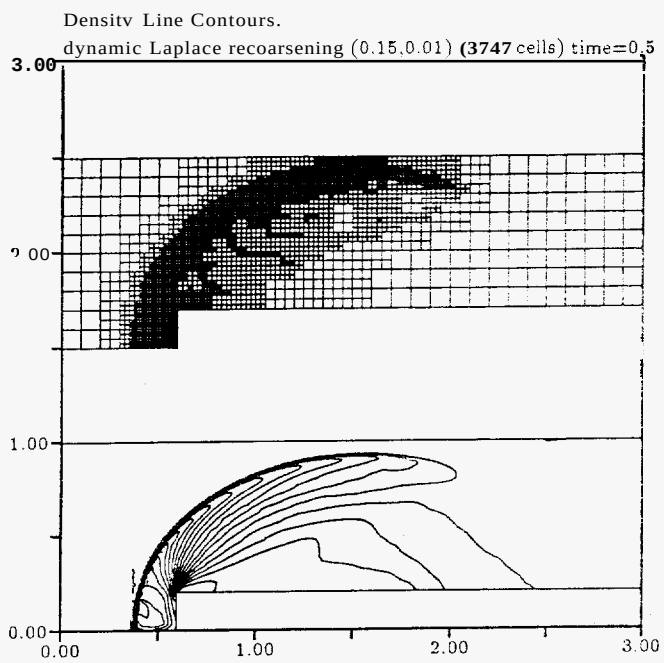


Figure 19: Adaptively-refined-grid solution; time=0.5

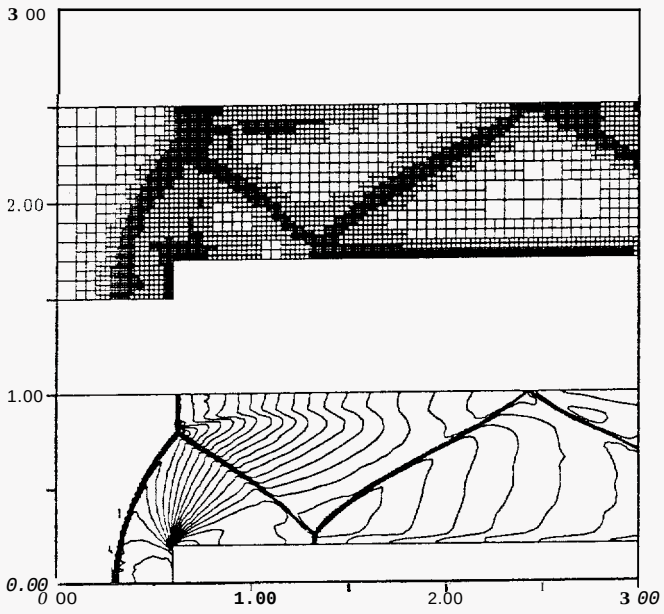


Figure 20: Adaptively-refined-grid solution; time=4.0

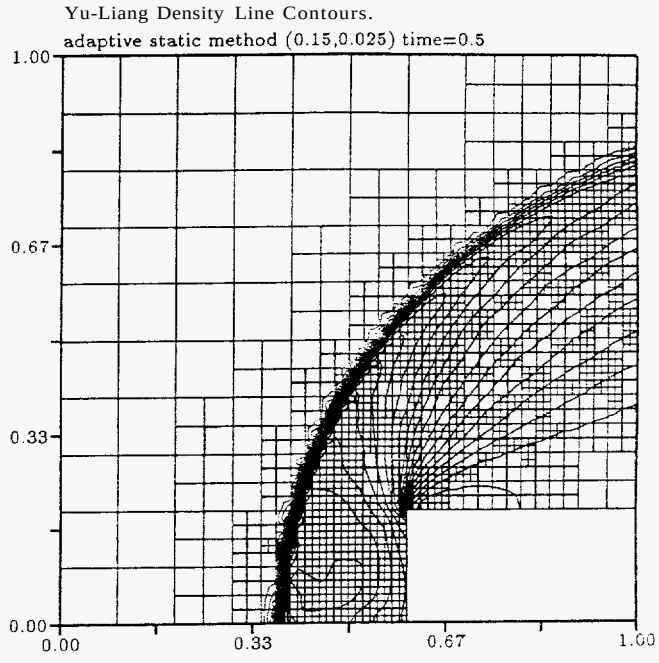


Figure 21: No refinement based on predicted wave motion

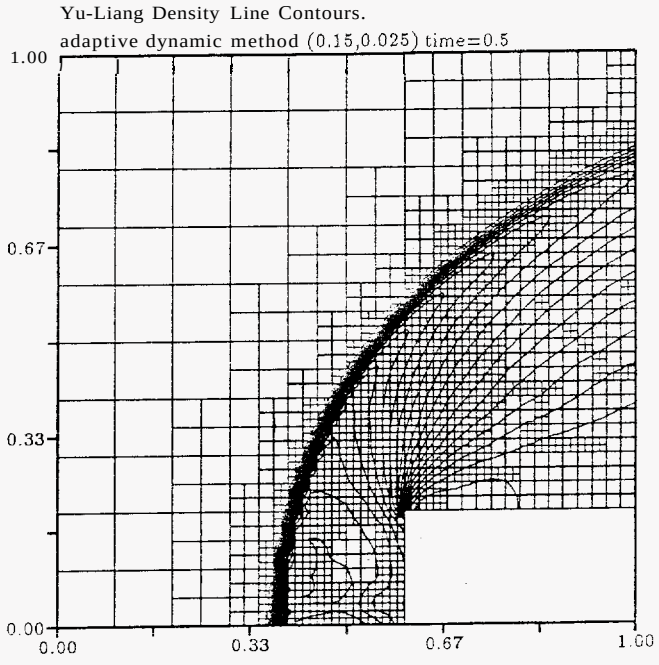
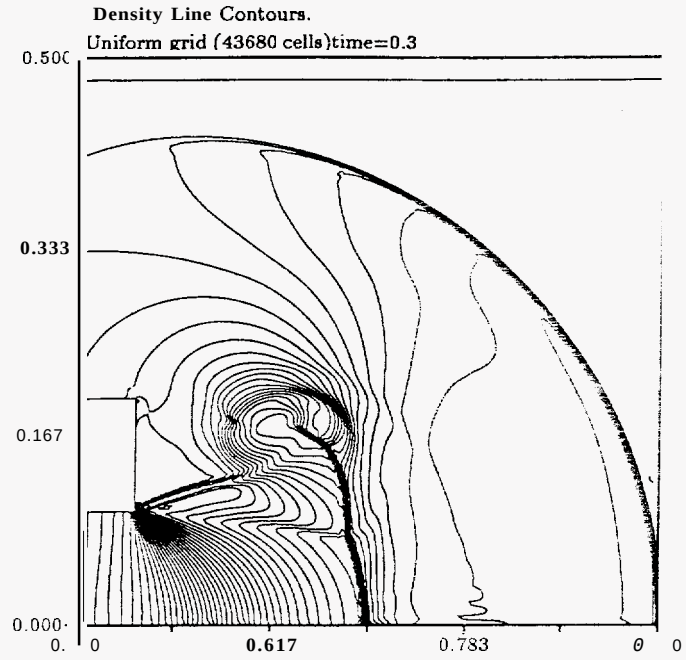
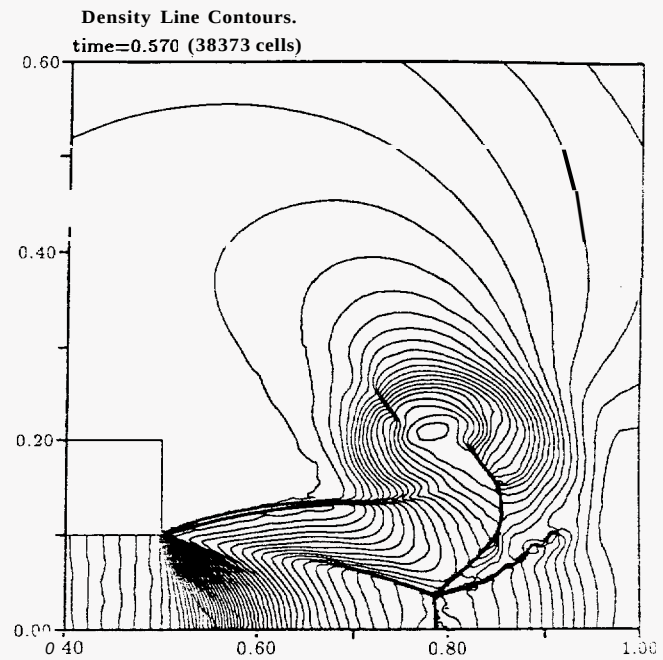
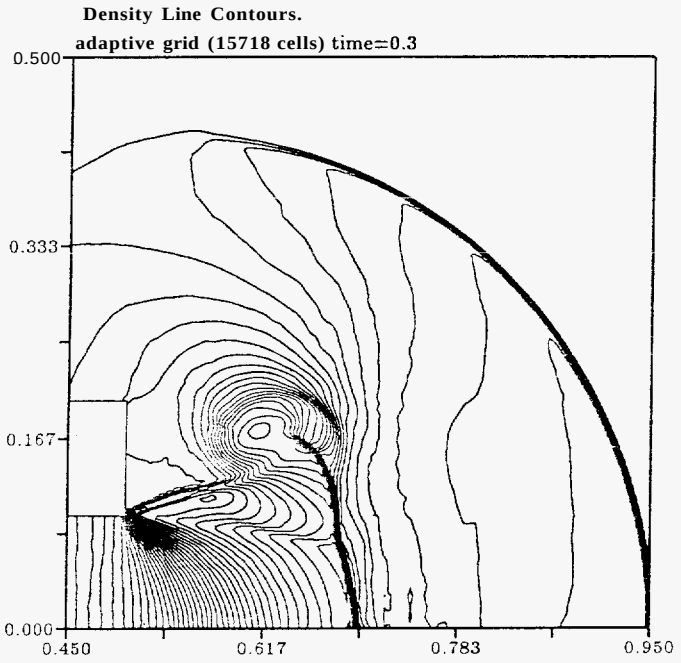


Figure 22: Refinement based on predicted wave motion.







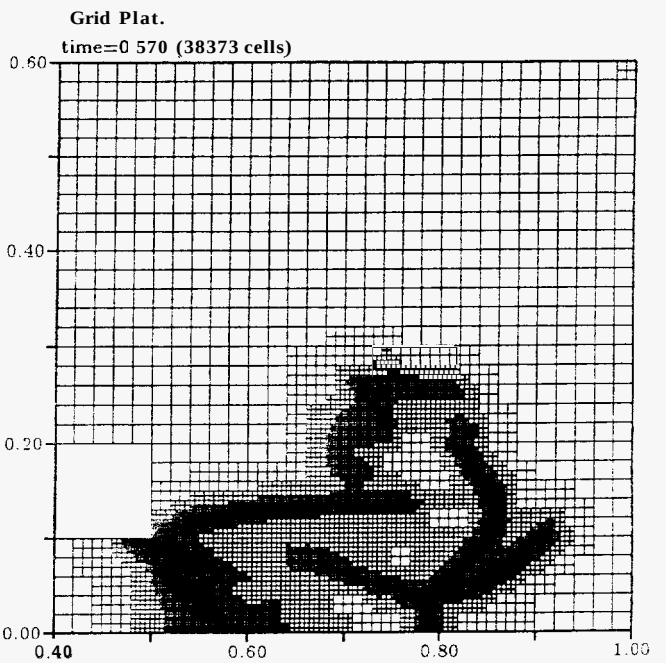


Figure **23:** Blast-wave problem: density distribution on uniform (top) and adaptive (bottom) grid; time=0.3.

Figure 24: Density distribution on the adaptive grid, time=0.570.

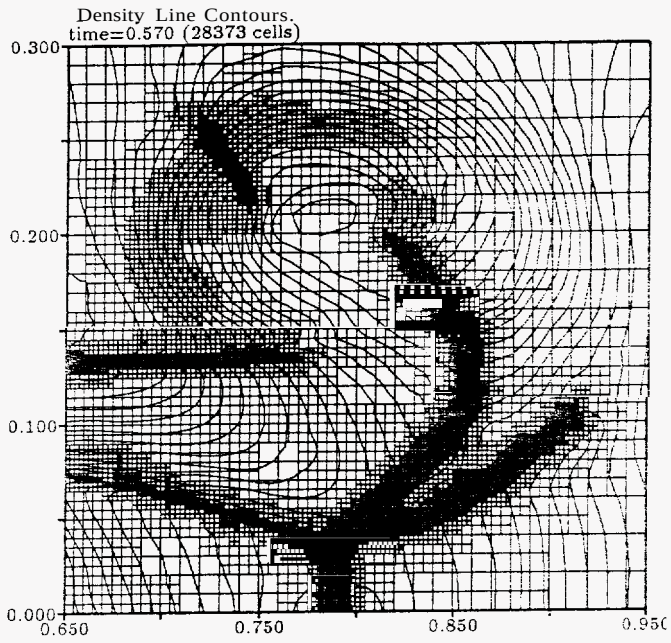


Figure 25: Density distribution on the adaptive grid; time=0.570.

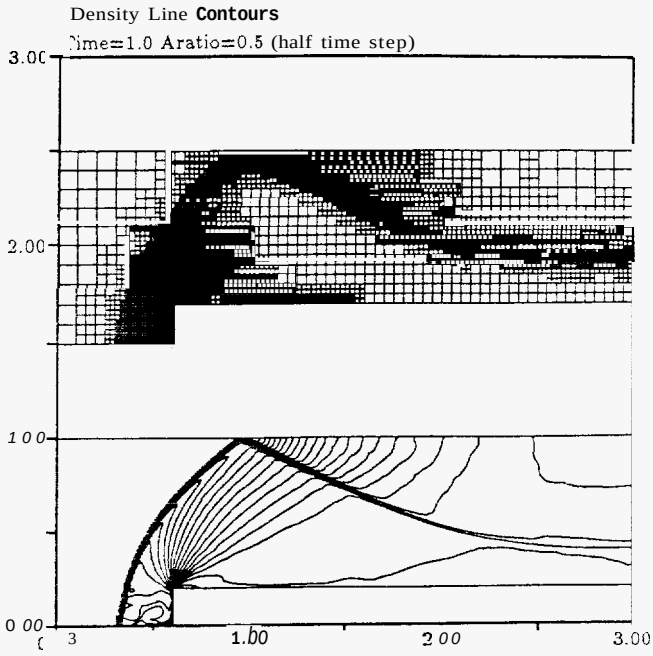


Figure 27: Density distribution and grid; time=1.0; reduced-time-step method.

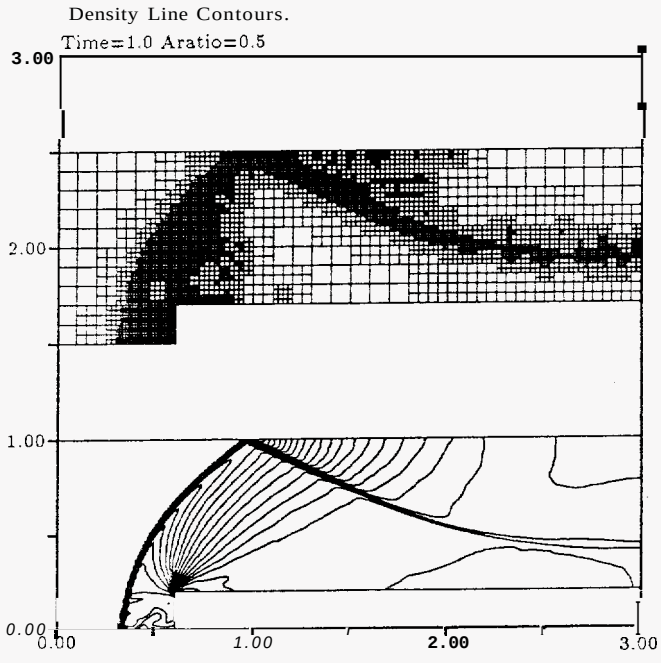


Figure 26: Density distribution and grid; time=1.0; cell-merging method.

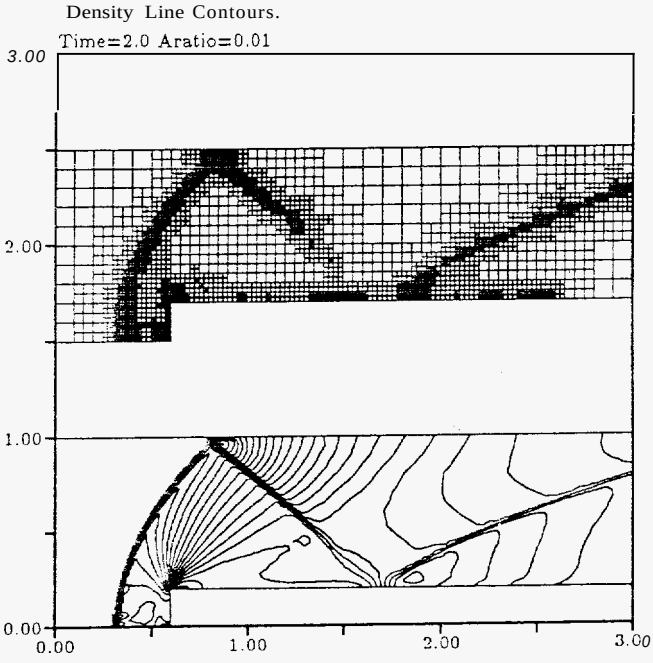


Figure 28: Density distiibution and grid obtained with cell-merging method; time= 2.0 and $A_{ratio} = 0.01$.