

**PORCTIONS  
OF THIS  
DOCUMENT  
ARE  
ILLEGIBLE**

MASLEN

CONF-820447--1

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

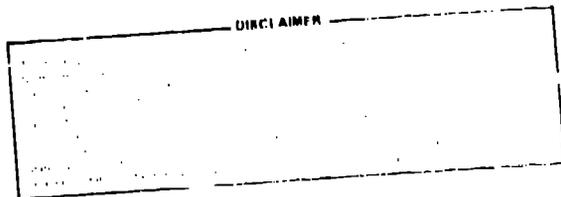
LA-UR--82-1044

DE82 014058

TITLE: APPLICATIONS AND GENERALIZATIONS OF VARIATIONAL METHODS FOR GENERATING ADAPTIVE MESHES

AUTHOR(S): Jeffrey Saltzman and Jeremiah Brackbill

SUBMITTED TO: Symposium of Numerical Generation of Curvilinear Coordinate Systems, Nashville, TN, April 16, 1982



NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE. It has been reproduced from the best available copy to permit the broadest possible availability.

LAN ONLY

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution to allow others to do so, for U.S. Government purposes.

The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy.

Los Alamos Los Alamos National Laboratory Los Alamos, New Mexico 87545

Applications and Generalizations of Variational

Methods for Generating Adaptive Meshes

Jeffrey Saltzman and Jeremiah Brackbill

Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, NM 87545

INTRODUCTION

Generating computation meshes for irregular regions have been of interest to a lot of people in many areas of research for a long time. One technique that has met with success over the long run has been to generate meshes using an elliptic equation or a system of elliptic equations.

The technique in its simplest form, uses a system of Laplace equations which are solved by direct or iterative methods. As people gained more experience with this method, source terms were added to the Laplace equations to gain additional control of the mesh. In addition, variable coefficients of the derivatives were added for further flexibility.

In this paper we work with a method that systematically generates a set of elliptic equations without having to explicitly perturb a set of Laplace equations with source terms and variable coefficients. This technique uses the variational methods often associated with elliptic equations.

Following this introduction, we will briefly discuss the variational formulation in two-dimensional cartesian geometry. Then the formulation will be generalized to three dimensions. Next, several three-dimensional test problems will be shown. After displaying these three-dimensional results, we will then exhibit an application of the mesh generation technique in two dimensions. This application involves generating an adaptive mesh for a supersonic flow past a step in a wind tunnel.

RESEARCH REPORT THE VARIATIONAL FORMULATION IN TWO DIMENSIONS

Finite difference schemes on nonuniform meshes all have the obvious characteristic that the independent variables in the calculation must be kept track of as well as the dependent variables. This is usually done by introducing an indexing scheme that tags the independent variables in the same way as the dependent variables. For example, in two dimensions, this is done by assigning two indices to a variable such as the fortran dimension statement

```
dimension x(30,40), y(30,40)
```

This is quite suggestive of a mapping. This mapping is one from a rectangular array of integers to a set of real coordinates. By filling in between the points using an interpolation scheme of some sort we now have a continuous mapping of a rectangular region into some two-dimensional region. This mapping is illustrated in figure 1. To be more systematic we will call the collection of points formed from the indices of the grid points the parameter space while we will call the collection of points formed from the grid points the physical space. With this mapping, we now have a tool to describe qualities of the given mesh in a quantitative manner.

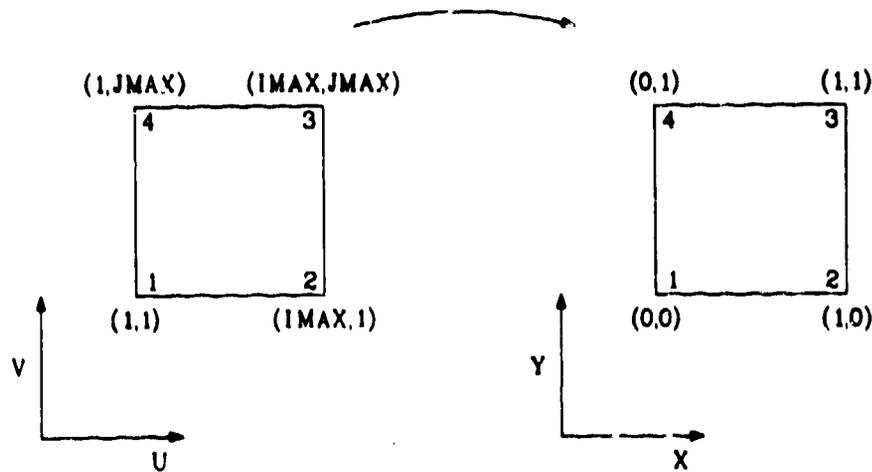


Fig. 1

FIG. 1-1

As illustrated in the first figure we consider a mapping from the two-dimensional parameter space  $(u,v)$  to the space  $(x,y)$ . We can quantify a mapping between these two spaces using the following functionals.

$$I_s = \iint (\nabla_{x,y} u)^2 + (\nabla_{x,y} v)^2 dx dy \quad (1)$$

$$I_o = \iint (\nabla_{x,y} u \cdot \nabla_{x,y} v)^2 dx dy \quad (2)$$

$$I_v = \iint w(x,y) J dx dy \quad (3)$$

where

$$J = \frac{\partial(x,y)}{\partial(u,v)} \quad (4)$$

and  $w$  is a given function of  $x$  and  $y$ .

We now describe the meaning of each functional.

The integral in equation (1) measures the smoothness of the mapping from  $(u,v)$  to  $(x,y)$ . In particular, the gradients in the integrand measures the spacing of the constant  $u$  and  $v$  lines. It seems plausible that a mesh that has smooth changes in spacing would have a functional value less than a jaggedly spaced mesh. We will call this integral the smoothness functional. The integral in equation (2) measures the orthogonality of constant  $u$  and  $v$  lines. If the mesh were perfectly orthogonal then the integral would be zero. We will call this integral the orthogonality functional. The integral in equation (3) measures how well the volume elements are conforming to a given weight function  $w(x,y)$ . If we were to minimize this integral, we would predict that where  $w$  is large  $J$  should be small and conversely where  $J$  is large  $w$  should be relatively small. Further, if  $J$  is small in a neighborhood of some point  $P$  then the grid should have many points close together in a neighborhood of the point  $P$ . We will call this last integral the volume weighting functional.

The functionals in the first three equations all measure useful qualities of a mesh. Both smoothness and orthogonality of a mesh are important to maintain accurate differencing. The volume weighting functional is useful in measuring

LINES TO  
B

3  
2  
1  
0

REF. 11111

Now a mesh is adapting to a given function. In addition to measuring mesh qualities, these functionals can be used to generate meshes as well. This is done by deriving a system of elliptic equations from the functionals. This process is broken into several steps.

The first step in this process is to write the integrals using (u,v) as the independent variables. This is useful later on when we will difference some equations. Next we take a linear combination of the integrals. The lambdas are all chosen positive and their relative size determines the importance given to each integral. With a single sum defined, we can minimize this integral using the methods of the calculus of variations.

$$I = \lambda_g I_g + \lambda_o I_o + \lambda_v I_v \tag{5}$$

$$\left[ \frac{\partial}{\partial x} - \frac{\partial}{\partial u} \frac{\partial}{\partial x_u} - \frac{\partial}{\partial v} \frac{\partial}{\partial x_v} \right] F = 0 \tag{6}$$

$$\left[ \frac{\partial}{\partial y} - \frac{\partial}{\partial u} \frac{\partial}{\partial y_u} - \frac{\partial}{\partial v} \frac{\partial}{\partial y_v} \right] F = 0 \tag{7}$$

where F is the integrand of the right hand side of equation (5).

To do so we calculate the Euler derivative of the integrand of the sum in equation (5). These expressions are listed in equations (6) and (7). Notice that having written the integrals in terms of (u,v) we can now difference the corresponding elliptic equations using symmetric differences on a rectangular region. We have chosen to solve the equations using an iterative scheme. This scheme is the classical Gauss-Jacobi iteration which is very amenable to vectorization. Now that we have given a sketch of the two dimensional equations, we move onto three dimensions. References [1,2] cover the two dimensional equations in more detail. One particular detail that should not be left to the references is that in practice the orthogonality term is multiplied by the term J<sup>3</sup> to counter problems with rounding errors. This new term slightly alters the effect of the functional in that regions where J is large are orthogonalized more than regions where J is small.

5.11.11

1  
2  
3  
4

GENERALIZATION OF THE VARIATIONAL FORMULATION TO THREE DIMENSIONS

The variational formulation easily generalizes to three dimensions. The first step in generalizing the equations is to define the appropriate mapping. This is simply done by adding one space dimension to both (u,v) and (x,y) to get (u,v,w) and (x,y,z). Equations (8), (9), (10) and (11) are the appropriate generalizations of equations (1) thru (4).

$$I_s = \iiint (\nabla_{x,y,z} u)^2 + (\nabla_{x,y,z} v)^2 + (\nabla_{x,y,z} w)^2 dx dy dz \quad (8)$$

$$I_o = \iiint (\nabla_{x,y,z} u \cdot \nabla_{x,y,z} v)^2 + (\nabla_{x,y,z} u \cdot \nabla_{x,y,z} w)^2 + (\nabla_{x,y,z} v \cdot \nabla_{x,y,z} w)^2 dx dy dz \quad (9)$$

$$I_v = \iiint w(x,y,z) J dx dy dz \quad (10)$$

where

$$J = \frac{\partial(x,y,z)}{\partial(u,v,w)} \quad (11)$$

The only real additional complexity is found in the orthogonality functional where two additional terms must be added to insure that orthogonality of all coordinate lines are measured. Again linear combinations of the integrals are taken and the Euler derivative of the integrand of the sum of these integrals is calculated. Equations (12) thru (15) list these expressions.

$$I = \lambda_s I_s + \lambda_o I_o + \lambda_v I_v \quad (12)$$

$$\left( \frac{\partial}{\partial x} - \frac{\partial}{\partial u} \frac{\partial}{\partial x_u} - \frac{\partial}{\partial v} \frac{\partial}{\partial x_v} - \frac{\partial}{\partial w} \frac{\partial}{\partial x_w} \right) F = 0 \quad (13)$$

$$\left( \frac{\partial}{\partial y} - \frac{\partial}{\partial u} \frac{\partial}{\partial y_u} - \frac{\partial}{\partial v} \frac{\partial}{\partial y_v} - \frac{\partial}{\partial w} \frac{\partial}{\partial y_w} \right) F = 0 \quad (14)$$

$$\left[ \frac{\partial}{\partial z} - \frac{\partial}{\partial u} \frac{\partial}{\partial z_u} - \frac{\partial}{\partial v} \frac{\partial}{\partial z_v} - \frac{\partial}{\partial w} \frac{\partial}{\partial z_w} \right] F = 0 \quad (15)$$

Again, in practice, the integrand of the orthogonality term is multiplied by  $J^3$  to reduce problems with rounding errors.

To demonstrate how these variational principles work in three dimensions several model problems will be presented. The first problem has a cubic physical region. Within this region a spherical exponential weight function is defined about the center of the cube. This problem will show how the weight functional influences the mesh. The second problem has a physical region that looks like an inverted pyramid with the tip cut off. This frustum problem will use the orthogonality functional to show its effect on the mesh. The last model problem uses the smoothness functional to generate a mesh around a cylindrical fuselage and an attached wing.

Figure 2 shows the mapping for the first model problem. The figure is a bit misleading since the cube actually rests at the origin. There are 20 points in each coordinate direction yielding a total of 8000 points. However only  $18 \times 18 \times 18$  (5832) points are actually iterated on since the others are fixed at the boundary. We use the following weight function. In this problem the

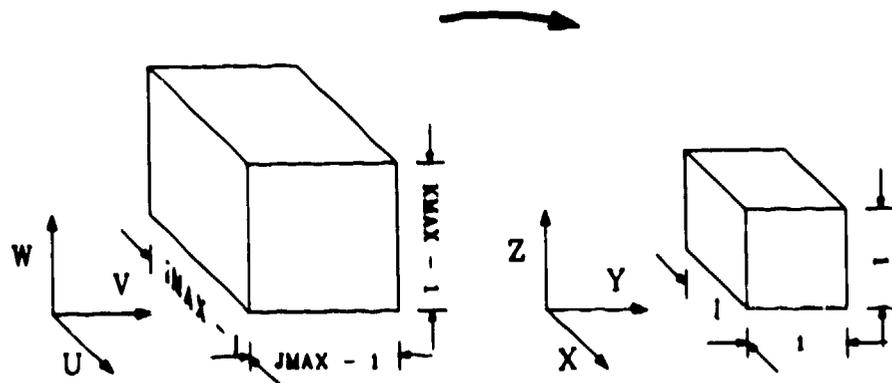


Fig. 2

weight  $\lambda_s$  for the smoothing functional will be set to one while the weight of the orthogonality functional will be set to zero. The weight for the volume weighting functional will vary between zero and one. To be able to show the full variation in the effect of the weight function while varying the parameter between zero and one a geometry dependent normalization was introduced for both the weighting functional and the orthogonality functional. These normalizations are introduced by dividing the lambdas by the lambda primes introduced in equations (16), (17) and (18).

$$\lambda_o' = \left(\frac{1}{Y}\right)^7 \quad (16)$$

$$\lambda_v' = \left(\frac{1}{Y}\right)^5 \hat{w} \quad (17)$$

$$\hat{w} = \frac{1}{V} \iiint w(x,y,z) dx dy dz \quad (18)$$

where

V is the physical volume region.

l is the physical length scale.

Y is the parameter length scale (number of points in a direction).

These normalizations were arrived at by using dimensional analysis. The "dimensions" of the weighting functional and the orthogonality functional were normalized to those of the smoothing functional. Finally the weighting function  $w(x,y,z)$  is defined in equation (19).

$$w(x,y,z) = 1000 \exp[(0.25 - (x^2 + y^2 + z^2)^{1/2})^2 / 0.05] \quad (19)$$

BEGAN HERE

Figure 3 shows the behavior of the weighting functional as a function of lambda. The behavior of the curve is what one expects since as more weight is given to the functional its influence should be felt more.

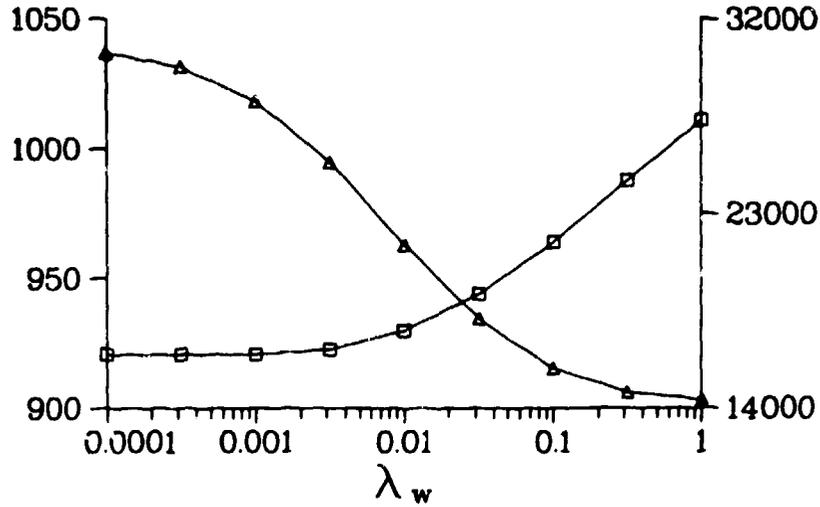


Fig. 3

The triangles mark values (right axis) of the volume weighting functional and the squares mark values (left axis) of the smoothing functional.

Figure 4 shows various cross-sections of the mesh when  $\lambda_v$  is set to one.

Figure 5 illustrates the frustum used in the second model problem. The coordinates of the bottom plane of the frustum are  $(x,y,z) = (0.25,0.25,0.5), (0.25,0.75,0.5), (0.75,0.75,0.5)$  and  $(0.75,0.25,0.5)$ . The coordinates of the top plane are  $(x,y,z) = (0.0,1), (1,0,1), (1,1,1)$  and  $(0,1,1)$ . In this problem the weighting for the smoothness functional is set to one while the weight of the orthogonality functional varies between 0 and 1000. The volume weighting functional is not used and subsequently has weight zero. Figure 6 shows the behavior of the orthogonality functional as a function of  $\lambda_o$  for the given range 0 to 1000. Again the curve exhibits a predictable behavior. Figure 7 shows several cuts made into the frustum. The cuts display how the mesh generation algorithm pushes the grid planes upward to make as many points as orthogonal as possible.

MOVES TO  
REVISION

7  
6  
5  
4  
3  
2  
1

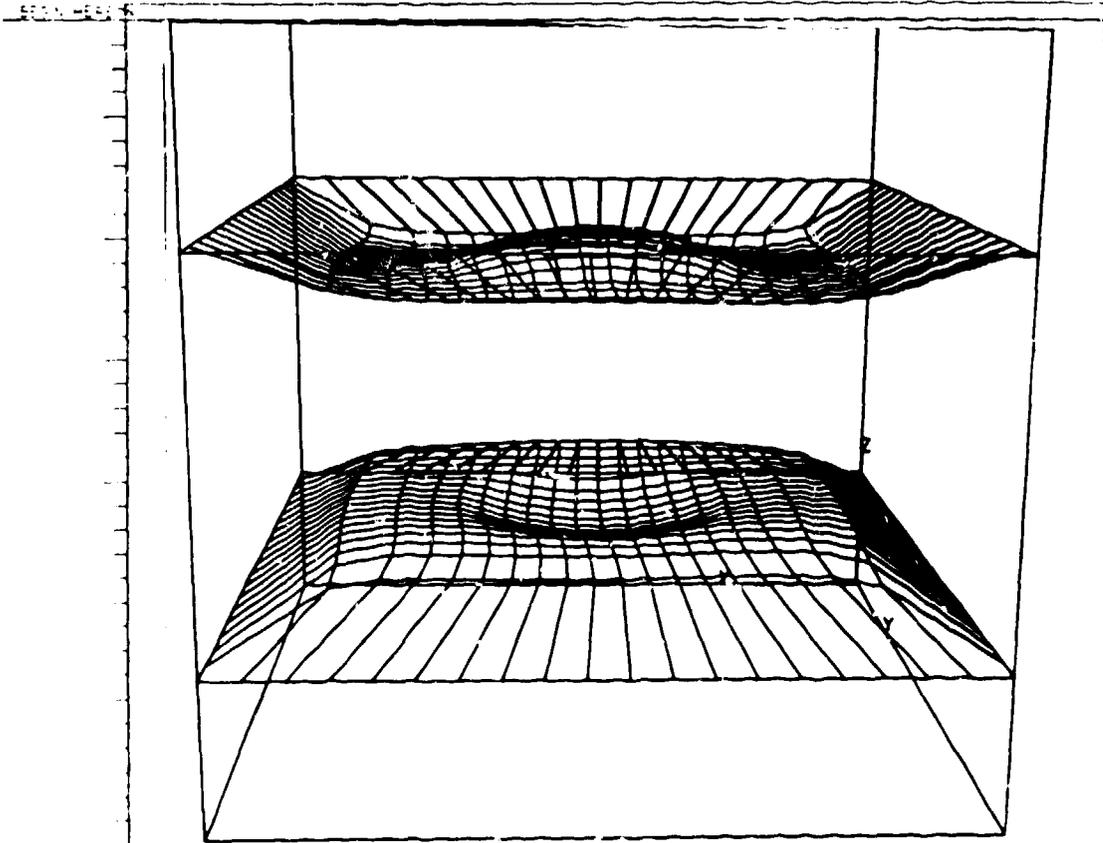


Fig. 4

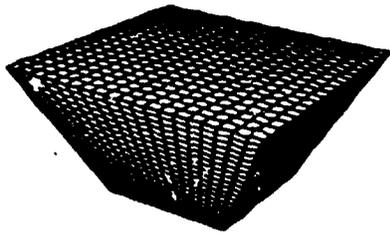


Fig. 5

6  
5  
4  
3

REG. HERE

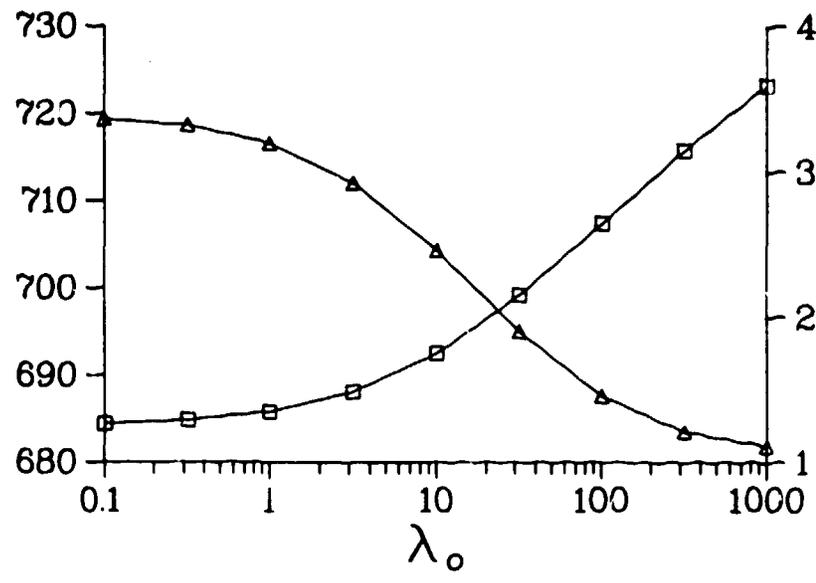


Fig. 6

The triangles mark values (right axis) of the orthogonality functional and the squares mark values (left axis) of the smoothing functional.

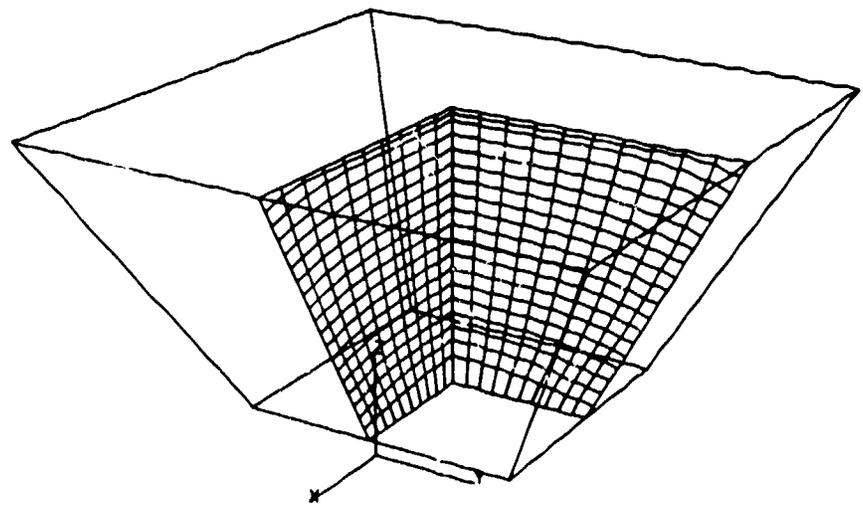


Fig. 7

LINES TO  
BOTTOM

7  
6  
5  
4  
3  
2  
1

Figure 8 displays the last model problem used to demonstrate the mesh generator. This figure shows a cylindrical fuselage with a wing attached.

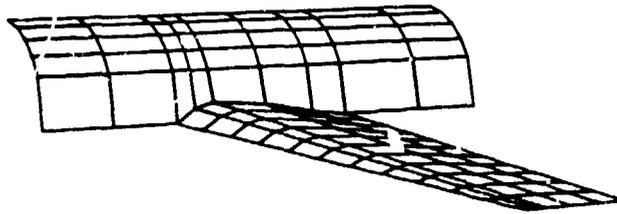


Fig. 8

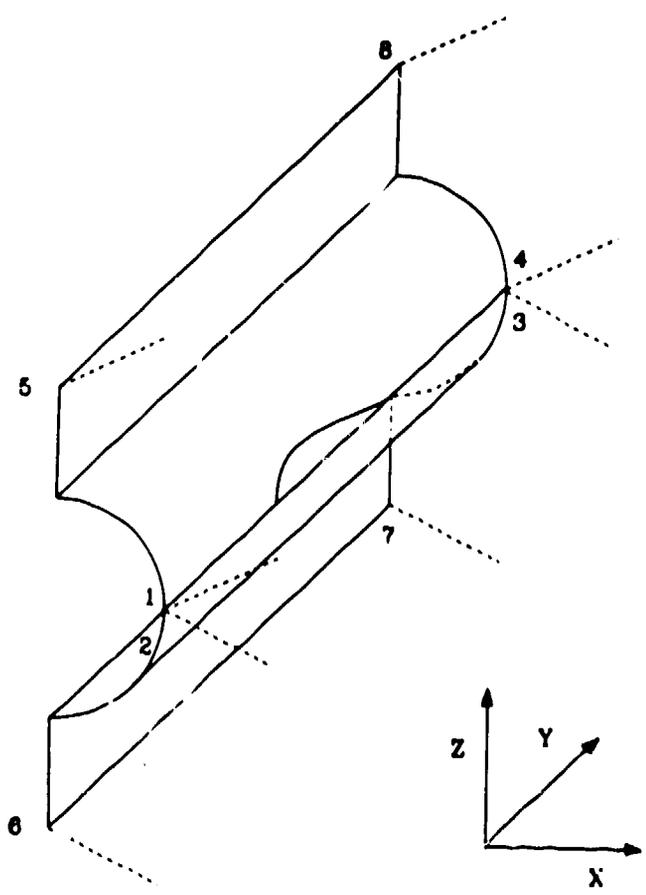
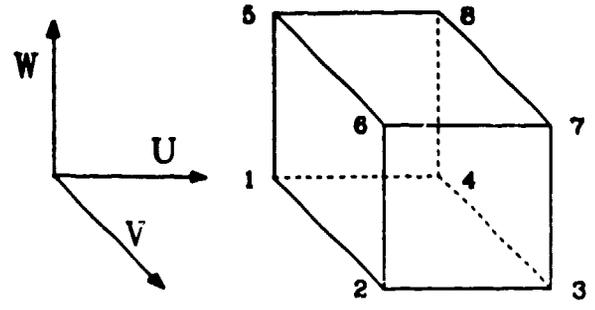
The fuselage is extended so that the cylinder extends  $-90$  degrees from the horizontal. In addition, a wall is constructed perpendicular to the wing. With these extensions a rectangular grid is wrapped around the wing as illustrated in figure 9. In this problem the smoothness functional will first be used without either of the other functionals to generate a regular mesh. In figure 10 a cut, seen from the front of the fuselage, of the grid is shown after 50 iterations. The mesh appears to be regular but has one troubling characteristic. One notices that the mesh is tightly spaced around the edge of the wing. After examining the equations derived from the smoothing functional it becomes clear by using electrostatic analogies that the mesh lines should bunch around the sharp wing edge much as potential lines concentrate about a lightning rod.

One possible way to fix the problem at the edge is to use the weighting functional to redistribute the grid points away from the edge by choosing a weight function appropriately. Figure 11 shows the region where the weight function is large. Elsewhere the weight function is very small or zero. The form of the function is unimportant. However, its location is away from the edge making it a good candidate to pull the mesh lines away from the wing edge. Finally figure 12 shows a cut made in the same manner as in figure 10.

FIG. 8

7  
6  
5  
4  
3  
2  
1  
0

REQ. HESS



LINES TO  
BOTTOM

Fig. 9

REF. 1111

Fig. 10

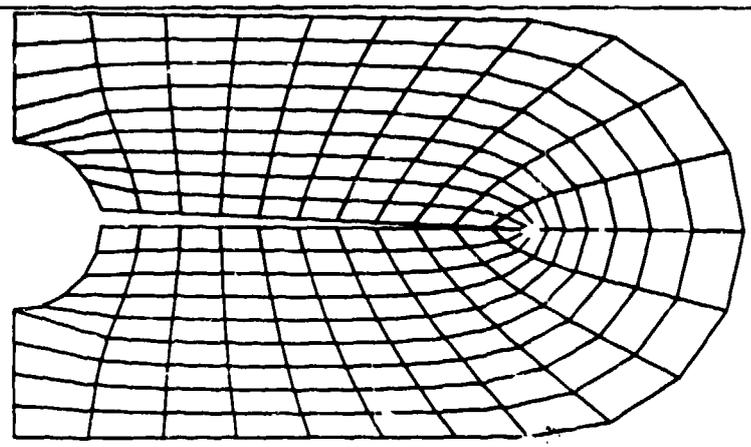


Fig. 11

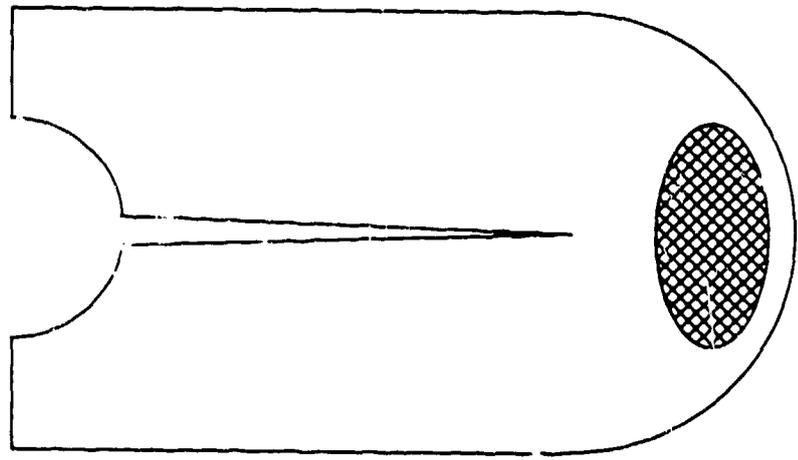
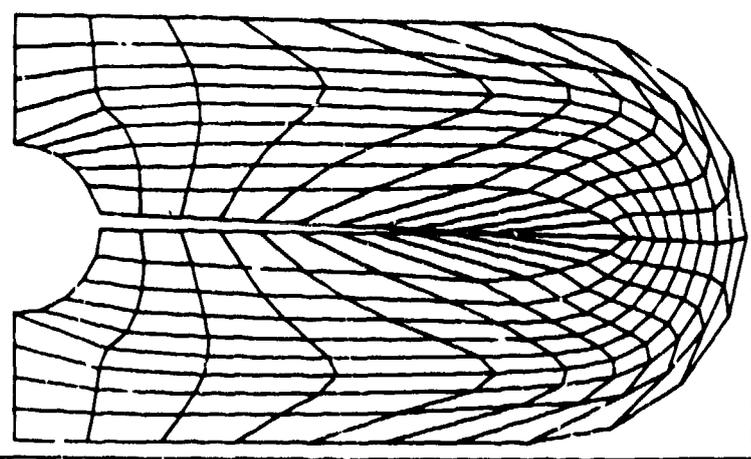


Fig. 12



LINES TO  
BE DRAWN

Fig. 10  
Fig. 11  
Fig. 12

BEGIN HERE

AN APPLICATION IN TWO DIMENSIONS

In this section we will show how the mesh generator can be combined with a two dimensional cartesian hydrodynamics code. All three functionals will be used to make the cartesian code adaptive. The step in a wind tunnel problem will be solved numerically to demonstrate the power of the method. This problem has been studied by many people and more information about its background can be found in references [2,3]. We first discuss the hydrodynamics code.

An appropriate model for the wind tunnel problem is the compressible inviscid fluid equations in two dimensional-cartesian geometry. These equations are listed below using a lagrangean formulation and an ideal equation of state.

$$\frac{dp}{dt} + \rho \nabla \cdot \vec{u} = 0 \quad (20)$$

$$\rho \frac{d\vec{u}}{dt} + \nabla p = 0 \quad (21)$$

$$\rho \frac{dI}{dt} + p \nabla \cdot \vec{u} = 0 \quad (22)$$

$$p = (\gamma - 1) \rho I \quad (23)$$

The standard variable names are used.  $\rho$  is the density,  $p$  is the pressure (which could include viscous contributions),  $\vec{u}$  is the velocity,  $I$  is the internal energy,  $t$  is time and  $\gamma$  is the ratio of specific heats. Although the fluid equations are inviscid, a viscous pressure will be added to prevent ringing at shock fronts.

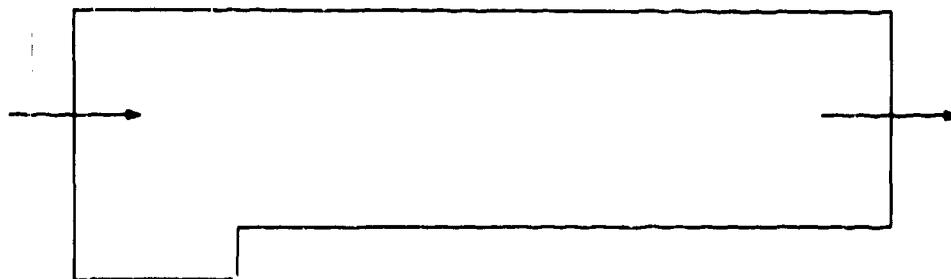
LINES TO  
BOTTOM

7  
6  
5  
4  
3  
2

REG. 4555

The equations will be differenced in two steps. The first step, called the Lagrangean phase, will approximate the above equations. Following the Lagrangean phase a remap phase will follow allowing an arbitrary mesh to be used. Because of length constraints on this paper, we cannot go into detail on the differencing. However, some general comments should be made about the differencing.

In the Lagrangean phase a conservative differencing scheme is used on a staggered quadrilateral grid. The velocities and masses are vertex centered while pressure and internal energy are cell centered. Differencing is explicit in time and stability is maintained using a Courant limitation on the time step. More details can be found in reference [2]. The remap phase of the calculation is also conservative making the entire scheme conservative. The central idea employed in the remap phase is the utilization of the fully two-dimensional FCT algorithm of Zalesak generalized to an arbitrary



### INFLOW PARAMETERS

$$\begin{aligned} \rho &= 1.4 \\ P &= 1.0 \\ u &= 3.0 \\ v &= 0.0 \\ \gamma &= 1.4 \end{aligned}$$

Fig. 13

LINES TO  
BOTTOM

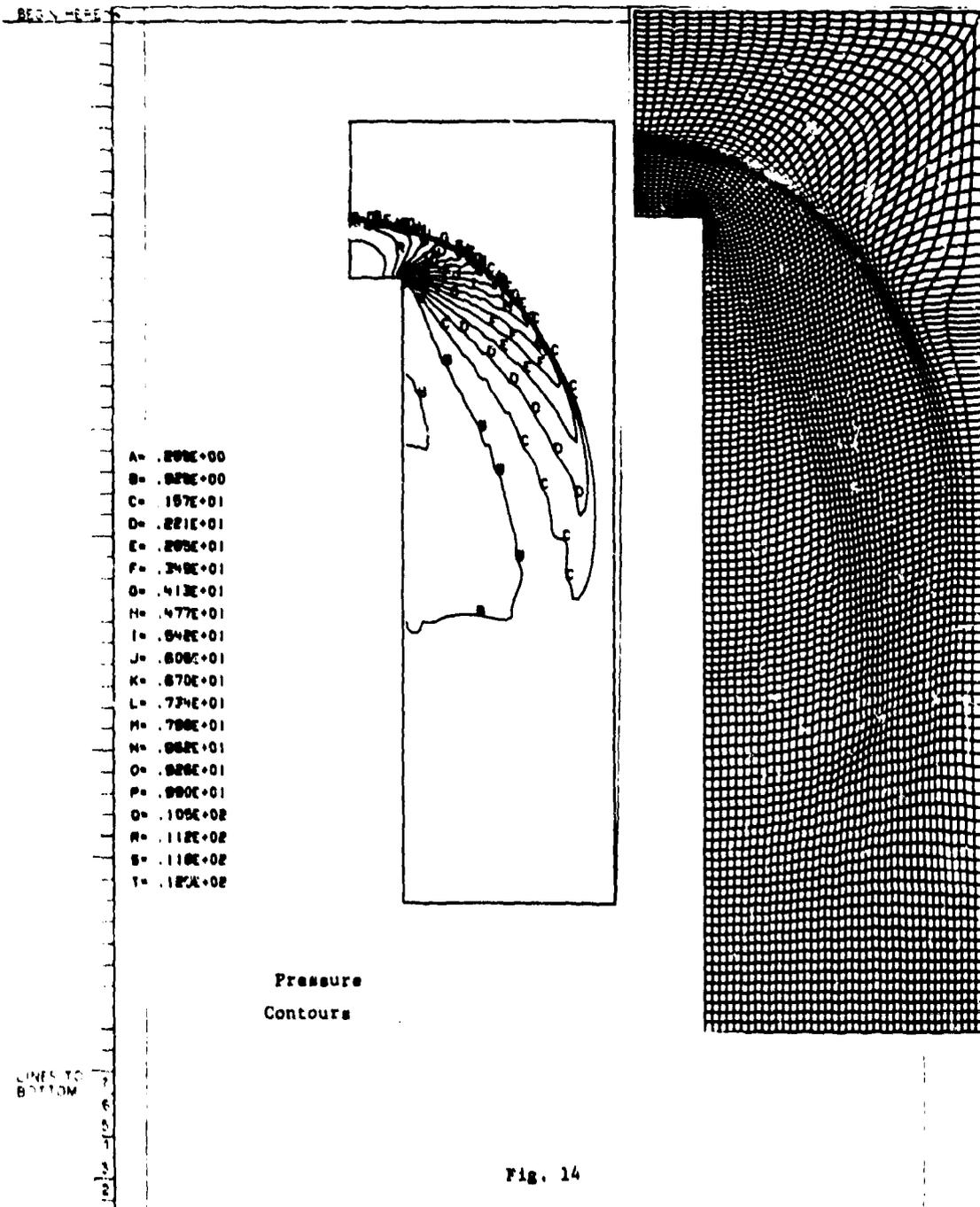
quadrilateral mesh (references [2,4]). The remap phase is dissipative assuring stability.

Figure 13 is a schematic of the initial conditions and boundary of the wind tunnel problem. Free slip boundary conditions are used on the top and bottom boundaries since the model is inviscid. At the corners of the step, the velocities are constrained to a direction parallel to a chord formed using adjacent boundary points. The outflow boundary conditions simply set all normal derivatives to zero. As it turns out, the flow is always supersonic at the outflow boundary making the boundary conditions irrelevant. The inflow boundary is set to make the flow Mach 3. The interior initially also has a uniform Mach 3 flow in the horizontal direction. Next, the initial mesh (120 x 40 cells) is created using the smoothness functional. Finally we introduce the weight function used in the volume weighting functional. It is well known for this problem that multiple shock structures develop throughout the region. To resolve these structures we have chosen the gradient length of pressure listed in equation (24).

$$w(x,y) = \frac{|\nabla p|^2}{p} \quad (24)$$

Figures 14 and 15 are snapshots of the computation at times 0.5 and 2.0 respectively. The primary purpose of these illustrations is to show that the grid changes drastically over time in order that it may follow the structure of the flow. Further comparisons can be made at fixed times to show that the grid is concentrated around gradients in the pressure. Another property of the computation mesh that is observed is how it aligns itself with the gradients of the pressure. The cells contract in a direction parallel to the gradients which enhances the resolution more than if the cells shrunk in a uniform manner. Computations were carried out without using the weighting functional as well. The primary shock structures were still recognized in this computation, but shock thicknesses were doubled. Also finer structures that appeared near the shocks were lost completely. In this particular problem, very little extra computation time was expended in running with an adaptive mesh for two reasons. The first reason is that the adaptive algorithm adds

ENDS TO  
BOTTOM



- A= .299E+00
- B= .287E+00
- C= .187E+01
- D= .221E+01
- E= .203E+01
- F= .249E+01
- G= .413E+01
- H= .477E+01
- I= .542E+01
- J= .608E+01
- K= .670E+01
- L= .734E+01
- M= .798E+01
- N= .862E+01
- O= .926E+01
- P= .990E+01
- Q= .105E+02
- R= .112E+02
- S= .118E+02
- T= .125E+02

LINES TO  
BOTTOM

BEGIN HERE

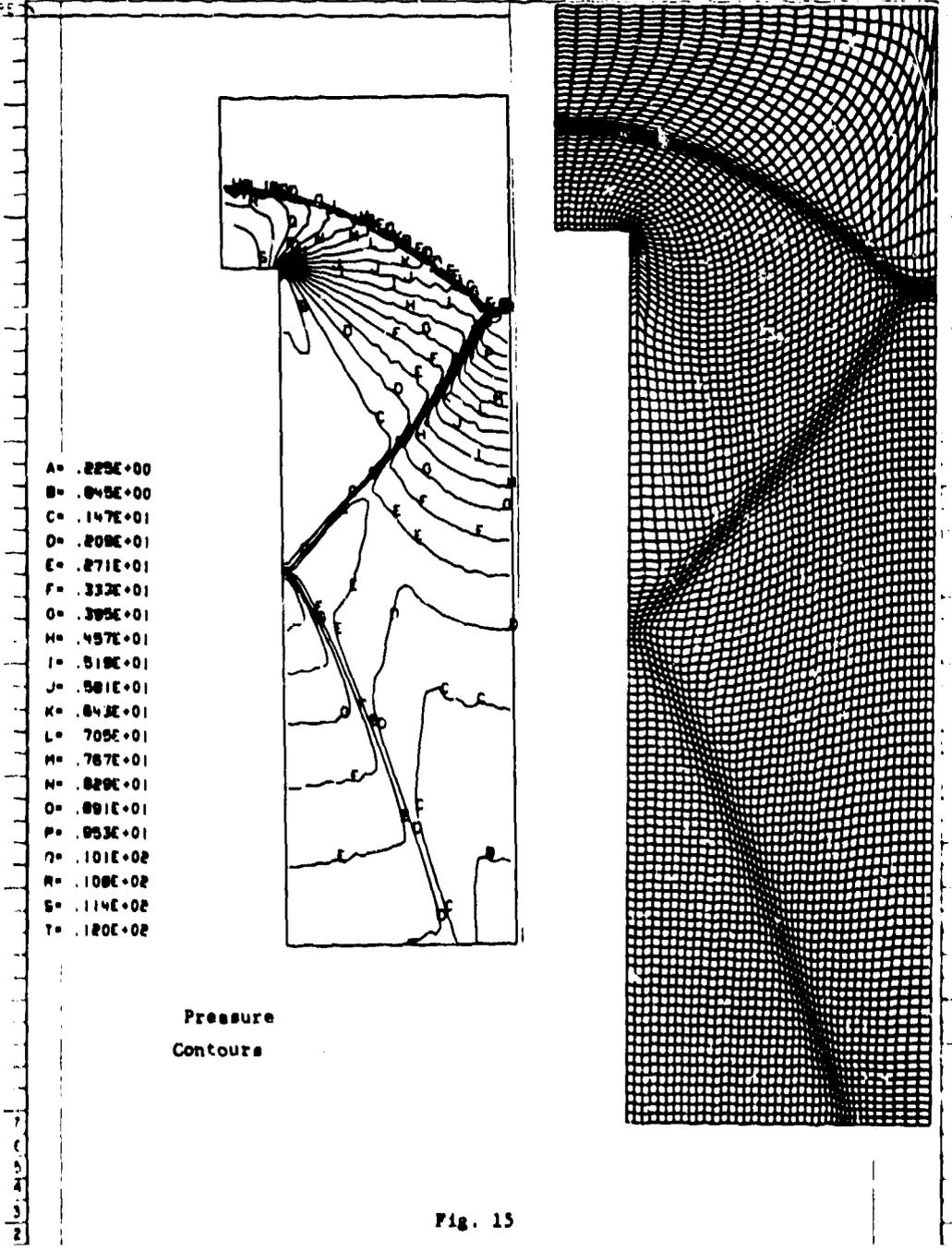


Fig. 15

BEG. MERE

little expense since few iterations were needed to update the mesh each time step. A more important factor was that the Courant condition for this problem was most limiting at the step corner. As a result both problems ran with almost the same number of time steps for a fixed interval of time.

#### CONCLUSIONS

To conclude, we summarize our results. We have shown that the variational formulation for generating meshes can easily be extended to three dimensions. Further the mesh generation equations behave in an easily predictable manner as illustrated with the three model problems given. We have also outlined a successful two-dimensional application of the mesh generator to a problem with moving multiple structures. The mesh generator moved the computation grid with the shock fronts and enhanced the resolution of the difference scheme significantly.

LINES TO

7

REF ID: A66877  
REFERENCES

- [1] J. Brackbill and J. Saltzman, Adaptive Zoning for Singular Problems in Two-Dimensions, To be published in the Journal of Computational Physics.
- [2] J. Saltzman, "A Variational Method for Generating Multidimensional Grids", Thesis, New York University, October 1981
- [3] B. van Leer, Towards the Ultimate Conservative Difference Scheme V. A Second Order Sequel to Godunov's Method, Journal of Computational Physics 32, 1979, pp. 101-136
- [4] S. Zalesak, Fully Multidimensional Flux Corrected Transport, NRL Memorandum, Report 3716.