Local Uniform Mesh Refinement for Elliptic Partial Differential Equations

William D. Gropp

Research Report YALEU/DCS/RR-278
July 1983

The work presented in this paper was supported in part by the Office of Naval Research under contract N000-14-82-K-0184 and by the National Science Foundation under grant MCS-8106181.

Abstract

Local Uniform Mesh Refinement is an efficient form of local refinement which has sucessfully been applied to time dependent problems. Its main advantages are its low overhead and suitability to vector and parallel machines. We show how to apply this approach to finite difference approximations of elliptic problems. By using the theory of M-matrices, we show that the mesh refinement approximation converges and that iterative methods based on splittings of the matrix problem suitable to parallel processors converge. A number of computational examples are provided.

1. Introduction

Many adaptive mesh methods have been proposed for the numerical solution of partial differential equations. The advantages are obvious: fewer points will reduce storage requirements and may reduce CPU time. The disadvantages are more subtle: adaptive methods are more difficult to program, the adaptivity often makes the code "unvectorizable", and there is often little theoretical justification for the form of refinement. Local Uniform Mesh Refinement [4, 5, 7] is one adaptive technique without these drawbacks which has been applied to time-dependent problems.

The advantages of local uniform mesh refinement lie in its local uniformity. By using locally uniform grids (rectangles), code can be written which takes advantage of this structure in much the same way as for a uniform grid. Further, the data structure overhead is much lower, since the basic item is a rectangle rather than a single point. Finally, because there are only a few fine grids, the theory for this form of adaptive griding is more tractable.

We will show in this paper how these ideas may be applied to the adaptive solution of elliptic partial differential equations. We first define our notation and then introduce the alogrithm. Next, we show that the algorithm gives a convergent approximation and that the resulting matrix problem can be solved by a simple iterative process which is provably convergent. We will then briefly discuss the data structures necessary to efficiently implement the algorithm. Finally, we will show some computational examples illustrating the method.

2. Notation

We wish to find the solution of an elliptic partial differential equation on a region R_1 with Dirichlet boundary conditions. We will assume that this region is rectilinear for simplicity, though that is not necessary for our results. It will also be clear from the discussion that more general boundary conditions (such as Robbins boundary conditions) can be handled with this technique. We place on this region a discrete grid G_1 of points $\{p_i\}$. Again, we will assume that these points are of the form (ih, jh), where h is the mesh width. G_1 does not include points along the boundary of R_1 (but see the remarks after Theorem 1). Our difference scheme for a point p_i is defined by a stencil $\{p_i, p_{i_1}, p_{i_2}, \ldots, p_{i_m}\}$. Thus, in constructing the matrix representing the difference scheme over G_1, the i^{th} row has non-zeros in columns i, i_1, i_2, ..., i_m.

To refine the grid G_1, we chose a region $R_2 \subset R_1$ which is rectilinear and whose boundary vertex points belong to G_1. We place on this region a grid G_2 of points $\{q_i\}$. Again, G_2 contains no points along the boundary of R_1.

We also define ∂G_2 as those points in G_2 which are in ∂R_2, the boundary of R_2, and $\partial_B G_1$ as those points $p_i \in G_1$ whose stencils would include points on ∂R_1. Define G^c as the complement of G.

We now divide G_1 into four disjoint sets of points:

$$G_{1E} = G_1 \bigcap G_2^c$$
$$G_{1B} = G_1 \bigcap \partial G_2$$
$$G_{1I} = \left\{ p_i \,|\, p_i \in (G_1 \bigcap G_2 \bigcap G_{1B}^c) \text{ and } p_i \in \text{stencil}(G_{1B} \bigcup G_{1E}) \right\}$$
$$G_{1S} = G_1 \bigcap (G_{1E} \bigcup G_{1B} \bigcup G_{1I})^c$$

G_{1E} is the part of the coarse grid exterior to the fine grid, G_{1B} is the part of the coarse grid on the boundary of the fine grid, G_{1I} is the part of the coarse grid inside of G_2 which contributes to the coarse grid stencil of some coarse grid point exterior or on the boundary of the fine grid, and G_{1S} is the part of the coarse grid which is no longer needed (values at those points in R_1 will be taken from the fine grid, and the points are not needed in any computation on G_1).

Similarly, the fine grid is divided into two disjoint sets of points:

$$G_{2B} = \partial G_2$$
$$G_{2I} = G_2 \bigcap G_{2B}^c$$

Each point $p_i \in G_{1I} \bigcup G_{1S} \bigcup G_{1B}$ has the same location in R_1 as some point $q_{f(i)} \in G_2$.

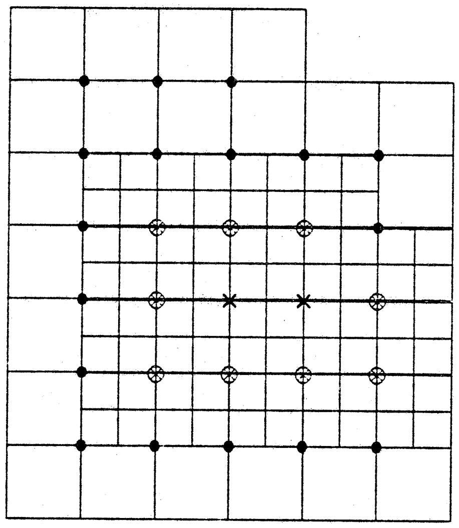

Figure 1: Relationships between the coarse and fine grids for a five point stensil. • are points in $G_{1E} \bigcup G_{1B}$, $\otimes$ points in G_{1I}, and $\times$ are points in G_{1S}.

We will also need a bit of matrix notation. If A is a matrix with elements $\{a_{ij}\}$, $A \geq 0$ means $a_{ij} \geq 0$ for all i and j. A matrix A is an M-matrix if $A^{-1} \geq 0$ and $a_{ij} \leq 0$, $i \neq j$. If A, M, and N are matrices, $A = M - N$ is a regular splitting of A if $M^{-1} \geq 0$ and $N \geq 0$ [11].

Our algorithm consists of specifying rules for determining equations for each of these sets of points.

3. Algorithm

The algorithm for constructing the matrix equation is:

1. for points in $G_{1E} \bigcup G_{1B}$, use the difference scheme on the coarse grid.
2. for each point $p_i \in G_{1I}$, use the equation $a_{ii} = 1$, $a_{if(j)} = -1$. This serves to inject the values computed on the fine grid values onto the coarse grid.
3. ignore points in G_{1S}.
4. for points in G_{2I}, use the difference scheme on the fine grid.
5. for each point $q_i \in G_{2B}$, use interpolation from points in G_{1B}.

The practical advantage of this algorithm lies in its local uniformity. We shall also see that it leads to a provably convergent approximation.

We will not discuss here how to decide where to refine. We shall see that the appropriate choice is to refine where the local truncation error becomes too large. Deciding where the local error is too large depends on the problem and the difference approximation used; possible choices for finite differences include direct approximation to the truncation error and Richardson-like extrapolation (see [8]). For finite element methods, *a posteriori* error estimates have been developed [1, 2, 12].

4. Convergence of Approximation

We will prove that this method converges as the local truncation error goes to zero for a significant class of problems. We do this by showing that the matrix for this method is an M-matrix under weak assumptions on the difference method used in the interior of the mesh and then applying well-known results about approximations which give M-matrices.

Theorem 1: For any connected region, let the matrix $A_I = \{a_{ij}\}$ for the interior scheme have the following properties:

(a) A_I is irreducible diagonally dominant;

(b) A_I can be written as $A_I = D - B$, where D is diagonal, $D \geq 0$ and $B \geq 0$; and

(c) let the set of points G_{1I} be non-empty.

Let the interpolation method (step 5 in the algorithm) for a point $q_i \in G_{2B}$ have non-zero matrix entries a_{ii}, a_{ij_1}, a_{ij_2}, ..., a_{ij_m}, where the points corresponding to columns j_1, j_2, ..., j_m are in G_{1B}. Further, let $a_{ii} > 0$ and a_{ij_1}, ..., a_{i,j_m} all be negative, and

$$\sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \leq |a_{ii}|.$$

Finally, for each row i corresponding to points $p_i \in \partial_B G_1$ and for each row i corresponding to points $q_i \in \partial_B G_2$, we have

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| < |a_{ii}|$$

(this is natural for Dirichlet boundary conditions). Then the matrix A (*not* just A_I) resulting from the algorithm is an M-matrix.

Proof:

We show this for a single fine grid. The proof can be applied repeatedly for more grids. Let A_1 be the matrix A_I for the coarse grid G_{1E} and A_2 the matrix A_I for the interior of the fine grid G_{2I}. Assume that the region $G_1 - G_2$ is connected. Then there is a permutation such that A has the form of the block matrix

$$A = \left[\begin{array}{c|c|c|c} I & 0 & 0 & C_{13} \\ \hline C_{11} & A_1 & 0 & 0 \\ \hline 0 & C_{12} & I & 0 \\ \hline 0 & 0 & C_{22} & A_2 \end{array}\right]$$

where the first row is for points in G_{1I}, the second row for points in $G_{1E} \bigcup G_{1B}$, the third row for points in G_{2B}, and the fourth row for the points in G_{2I}. The matrix C_{11} contains at least one non-zero per column by the definition of G_{1I}. The matrix C_{22} contains at least one non-zero per column by the definition of G_{2B}. The matrix C_{12} contains at least one non-zero per row by the definition of the interpolation method. Finally, the matrix C_{13} contains one non-zero per row by the definition of G_{2I}.

To show that this matrix is irreducible, we show how a chain $\{a_{i,i_1}, a_{i_1,i_2}, \ldots, a_{i_m,j}\}$ can be constructed for all i and j. First, note the A_1 and A_2 are irreducible by hypothesis. Let the partition of A be $1 \leq n_1 \leq n_2 \leq n_3 \leq n$ (the diagonal blocks are square).

1) $1 \leq i \leq n_1$, $n_3 < j \leq n$. Take first $\{a_{ii}\}$. There is a j_1 in $n_3 < j_1 \leq n$ such that $\{a_{ij_1}\} \neq 0$ (since C_{13} has one -1 entry per row). From here, there is a chain to j since A_2 is irreducible.
2) $n_3 < i \leq n$, $n_2 < j \leq n_3$. There is an i_1 in $n_3 < i \leq n$ such that $\{a_{i_1,j}\} \neq 0$ since C_{22} has at least one entry per column. Since A_2 is irreducible, there is a chain from i to i_1.
3) Similarly for $n_2 < i \leq n_3$, $n_1 < j \leq n_2$ and $n_1 < i \leq n_2$, $1 \leq j \leq n_1$. Since we can compose chains, we have shown that A is irreducible as long as $n_1 \geq 1$.

To show that A is irreducibly diagonally dominant, we need only to show that some row is strictly diagonally dominant. Since there either is a coarse grid point adjacent to a boundary which is not refined or there is a fine grid point adjacent to a boundary, the row corresponding to the stensil for that point will have a column "missing," and hence that row will be strictly diagonally dominant. Both of these satisfy the requirements, so we are done. Finally, we note that $a_{ij} \leq 0$ for $i \neq j$ by the construction of A, so that A is an M-matrix.

Now, if A_1 is not irreducible (ie., the fine grid divides the coarse grid into two or more disconnected regions), we can still show that a chain can be formed, because by hypothesis, C_{11} will have enough non-zeros, since each component connected component of G_1 must have points whose stencils include points in

G_{1I}. Otherwise, that component can be solved for seperately, which contradicts our hypothesis that G_{1I} is non-empty for each connected component.

∎

Remarks

1 The restriction that G_1 and G_2 not include any points from the boundary of R_1 is necessary only to allow us to use M-matrices. This could happen, for example, if the matrix includes diagonal terms for the points in G_1 and G_2 on ∂R_1 (for simplicity in coding). Inclusion of these points does not change the properties of the matrix which we are interested in: the existence of a non-negative inverse. This is easy to see by explicit construction of the inverse for the block 2×2 matrix

$$\left[\begin{array}{c|c} I & 0 \\ \hline C & M \end{array}\right]$$

where M is an M-matrix, and $C \leq 0$.

2 If $G_{1I} = \emptyset$, then C_{13} is empty and the matrix is block lower triangular. It is easy to see then that the solution on the coarse grid G_1 is independent of the solution on the fine grid G_2. We can then solve the coarse grid approximation first and then use the solution to specify the values at the boundary of R_2. In this case, the accuracy of the solution can be bounded in the usual way by the sum of the error at the boundary and the local truncation error. Since the error at the boundary of R_2 depends on the accuracy of the solution on G_1, solutions of this kind are usually not very useful.

We now state a theorem on the convergence of approximations to a class of elliptic equations. See [9] for the proof.

Theorem 2: Consider the elliptic boundary value problem

$$\mathrm{L}u \equiv -a_{11}\, u_{xx} - 2a_{12}\, u_{xy} - a_{22}\, u_{yy} - b_1\, u_x - b_2\, u_y$$

where

$$\begin{aligned} \mathrm{L}u &= q(x,y), \quad (x,y) \in R_1, \\ u(x,y) &= \psi(x,y), \quad (x,y) \in \partial R_1, \end{aligned}$$

and

$$a_{11},\, a_{12},\, a_{22},\, b_1,\, b_2 \in \mathrm{C}^{\infty},$$

$$q(x,y),\, \psi(x,y) \in \mathrm{C}^0, \text{ and}$$

$$a_{11} > 0, \qquad a_{11}a_{22} - a_{12}^2 > 0.$$

Approximate $\mathrm{L}u = q$ with a consistent difference scheme, using the exact values at the boundary. If the difference scheme results in an M-matrix, then the approximation converges in the discrete infinity norm. Further, if the approximate solution is w and the exact solution is u, then

$$\|Au - Aw\|_\infty \geq K\|u - w\|_\infty,$$

where K is independent of the stepsize.

Remarks

This is by no means the most general theorem possible. We could quickly extend this result to semi-linear L's, approximations at the boundary ∂R_1, and more general boundary conditions of the form $u + \alpha \partial u/\partial n = \beta$. This also gives us criteria for refining, since $\|Au - Aw\|_\infty = \|Au - q\|_\infty = \|Au - Lu\|_\infty$ measures the maximum of the local truncation error over the region.

5. Convergence of Iterative Solution

Now that we have a positive definite, non-symmetric matrix for our approximation, we naturally want to be able to select a solution method which can be guarenteed to converge.

We show the splitting as made up of 4×4 block matrices since each each row and column of the 4×4 matrix can be related to our decomposition of the grids in section 2. However, our splittings are all in terms of 2×2 partitionings, with each block in the 2×2 partition made up of a 2×2 piece of the 4×4 partitioning.

Perhaps the most natural iteration is to do each grid seperately and then combine the results. This splitting is suitable for parallel processors, as each grid is done independently. Also, this splitting is simply a (2×2) block Jacobi method. For the case above (with one fine grid and $G_1 - G_2$ a connected set), we split A into

$$A = \left[\begin{array}{c|c|c|c} I & 0 & 0 & 0 \\ \hline C_{11} & A_1 & 0 & 0 \\ \hline 0 & 0 & I & 0 \\ \hline 0 & 0 & C_{22} & A_2 \end{array}\right] + \left[\begin{array}{c|c|c|c} 0 & 0 & 0 & C_{13} \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & C_{12} & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right].$$

or $A = N_J - P_J$. The natural iteration is

$$N_J x^{k+1} = P_J x^k + b.$$

We can show that this converges by showing that this splitting is regular and then applying the regular splitting theorem [11].

Theorem 3: $N_J - P_J$ is a regular splitting for the matrix A in Theorem 1

Proof:

There are two conditions to check: $P_J \geq 0$ and $N_J^{-1} \geq 0$. $P_J \geq 0$ since all elements of C_{13} and C_{12} are negative or zero. To check the second condition, we explicitly construct the inverse of N_J:

$$N_J^{-1} = \left[\begin{array}{c|c|c|c} I & 0 & 0 & 0 \\ \hline -A_1^{-1}C_{11} & A_1^{-1} & 0 & 0 \\ \hline 0 & 0 & I & 0 \\ \hline 0 & 0 & -A_2^{-1}C_{22} & A_2^{-1} \end{array}\right].$$

Now, since A_1 and A_2 are M-matrices, they have non-negative inverses. Since $C_{11} \leq 0$ and $C_{22} \leq 0$, $N_J^{-1} \geq 0$.

∎

Another splitting is to solve the problem on the coarse grid, then use that solution in solving on the fine grid. The splitting for this is

$$A = \left[\begin{array}{c|c|c|c} I & 0 & 0 & 0 \\ \hline C_{11} & A_1 & 0 & 0 \\ \hline 0 & C_{12} & I & 0 \\ \hline 0 & 0 & C_{22} & A_2 \end{array}\right] + \left[\begin{array}{c|c|c|c} 0 & 0 & 0 & C_{13} \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right].$$

or $A = N_G - P_G$. The natural iteration is

$$N_G x^{k+1} = P_G x^k + b.$$

Again, we show that this iteration converges by showing that the splitting is a regular splitting.

Theorem 4: $N_G - P_G$ is a regular splitting for the matrix A in Theorem 1

Proof:

$P_G \geq 0$ since all elements of C_{13} are negative or zero. Again, we explicitly construct the inverse of N_G:

$$N_G^{-1} = \left[\begin{array}{c|c|c|c} I & 0 & 0 & 0 \\ \hline -A_1^{-1}C_{11} & A_1^{-1} & 0 & 0 \\ \hline 0 & -C_{12} & I & 0 \\ \hline 0 & A_2^{-1}C_{22}C_{12} & -A_2^{-1}C_{22} & A_2^{-1} \end{array}\right].$$

Now, since A_1 and A_2 are M-matrices, they have non-negative inverses. The product $C_{22}C_{12} \geq 0$, since $C_{22} \leq 0$ and $C_{12} \leq 0$. Since $C_{11} \leq 0$ and $C_{22} \leq 0$, $N_G^{-1} \geq 0$.

∎

Since $N_G - P_G$ is a (2×2) block Gauss-Seidel method, and the block matrix is consistently ordered and 2-cyclic, we know [11] that the second splitting converges twice as fast as the first.

We can also consider a (2×2) block SOR method with the splitting:

$$\omega A = \left[\begin{array}{c|c|c|c} I & 0 & 0 & 0 \\ \hline C_{11} & A_1 & 0 & 0 \\ \hline 0 & \omega C_{12} & I & 0 \\ \hline 0 & 0 & C_{22} & A_2 \end{array}\right] + \left[\begin{array}{c|c|c|c} (\omega-1)I & 0 & 0 & \omega C_{13} \\ \hline (\omega-1)C_{11} & (\omega-1)A_1 & 0 & 0 \\ \hline 0 & 0 & (\omega-1)I & 0 \\ \hline 0 & 0 & (\omega-1)C_{22} & (\omega-1)A_2 \end{array}\right].$$

The optimal choice of ω is not obvious. However, we do know some things about the related block Jacobi matrix $J = N_J^{-1}P_J$. First, $J \geq 0$. Further, J can be written in block diagonal form, with the first block irreducible and the second identically 0. In this case, we know from the Perron-Frobenius theory that J has a positive real eigenvalue equal to its spectral radius, and that the associated eigenvector is positive. We can use this as the basis of a heuristic to get an estimate of the optimal ω by computing Jx for a positive x, and using that to estimate the spectral radius of J. We then use that in the usual formula

$$\omega_c = \frac{2}{1 + \sqrt{1 - \rho^2(J)}}$$

to get an estimate of the optimal ω. However, since J may have complex eigenvalues, the optimal choice of ω is more complicated; see [13] for details. Other estimates of ω_c are of course possible.

6. Data Structure

The principle advantage of Local Uniform Mesh Refinement is the computational efficiency of the uniform structure of the refined grids. To realize this efficiency we must make a careful choice of data structure. We discuss here the choice which we have implemented; this is a generalization of the data structure in [7].

The basic item in our data structure is a *rectangle*. This is a regular grid of points of the form $\{x_0+ih_x, y_0+jh_y\}$ for $1 \le i \le n_x$, $1 \le j \le n_y$. Note that in contrast to the rectangles of [3], these rectangles are lined up in the coordinate directions. This choice agrees with the theory developed in section 4 as well as significantly simplifing the code. There is a cost in terms of more grids and mesh points.

The values of the dependent variables on a rectangle are stored in a contiguous block of memory to allow for easy access to nearby values (this allows vectorization of many operations). The hard part of this data structure is the joint where two rectangles abut. Instead of trying to patch two adjacent grids together, we allow them to overlap at the joint. In figure 2, we see that points 7, 8, and 9 of rectangle 1 overlap points 11, 12, and 13 of rectangle 2.

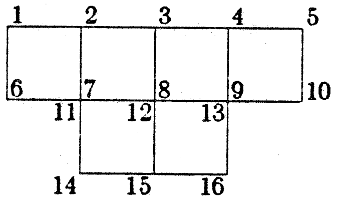

Figure 2: Rectangle/rectangle joint.

Linked lists are used to describe the joints and other boundaries (such as coarse/fine grid boundaries and boundaries of the problem). These lists contain information on the relationship between the grids (e.g., point 11 in rectangle 2 is at the same (x, y) location as point 7 in rectangle 1), length of the boundary in grid points, as well as other data useful in setting up the lists.

The grids for each level of refinement are linked together in another list. Constructing the matrix is then done by simply moving through this list. Since we allow points to overlap at a grid/grid joint, we don't have to worry about the numbering of the points inside of a grid depending on the exact placement of nearby grids. The lists describing the boundaries of the grids give us what little information we need in carrying a difference scheme across a grid/grid joint. Because of the uniformity in the grids, the matrix can be constructed in a format suitable for vector machines (for example, the matrix can be constructed with long diagonals suitable for fast matrix-vector products).

7. Computational Examples

In this section we show effective this algorithm is on three selected problems. The first problem is linear and has a smooth solution. The error is localized in the center of the domain. This example demonstrates the theoretical results of sections 4 and 5. The second problem has a singular derivative at a point on the boundary; this problem shows the application of this technique when the theory is not valid. The third problem is nonlinear and shows the advantage of combining the iterative methods of section 5 with the solution of the nonlinear problem.

All tests were run on a VAX 11/780 running the VMS operating system; all computations were done in single precision. The linear systems were all solved with orthomin and generalized ILU preconditioning [6]. In all examples, iterations were stopped when the residual became less than 10^{-5}.

The first problem is

$$\nabla^2 u = 4\sigma\left(\sigma r^2 \exp(-\sigma r^2)\right)$$

$$\text{where} \qquad r = \sqrt{(x-\frac{1}{2})^2 + (y-\frac{1}{2})^2}$$

on the unit square. The boundary data was chosen so that the exact solution was

$$u = \exp(-\sigma r^2)$$

which is smooth. The truncation error is concentrated around $(\frac{1}{2}, \frac{1}{2})$. In all of our tests, $\sigma = 20$, which gives the peak a half width of about 0.19.

Table 1 shows the result of a uniform fine grid computation. The error is measured at the coarse grid points. The error does include the error in solving the difference equations, but this error is a small contribution to the total error. Note that the error decreases by about a factor of 4 for each halving of the step size, as we expect. A graph of the computed solution for a uniform grid with step size 0.0125 is shown in figure 3, and the error in that solution in figure 4.

h_c	error_∞	time (secs)
.1	5.32E-2	.44
.05	1.27E-2	1.9
.025	3.10E-3	15.
.0125	6.94E-4	178.

Table 1: Results for a uniform grid for problem 1.

Now, in order to test the theory, we use the errors achieved on the uniform fine grids to set the permisible error for mesh refinement calculations. Further, we will compute the exact error in deciding where to refine. While this is certainly not realistic for a real calculation, it seperates the issue of error estimation from the theory we wish to demonstrate. This does affect the timing results, but in many cases the error estimation used will not be much more expensive than the calculation of the exact error which we used. In addition, we compute the error which would be committed by linear interpolation along a refined grid boundary, and require that error also be less than the specified amount. This is necessary to assure an accurate solution. For this first set of data, only one iteration of the (2×2) block Gauss-Seidel method was used.

The first calculation uses two levels of refinement and a ratio of grid sizes of 2, so that the finest grid is $\frac{1}{4}$ the mesh width of the coarsest grid. The grids were refined where the local error exceeded the "error tol" given in Table 2. A graph of the error in the solution with $h_c = 0.05$ is shown in figure 5, and the grid for that solution in figure 6. The results in Table 2 show clearly the advantage of this approach. For example, a uniform grid calculation which gave an error of 6.94E-4 took 178 seconds, while a 3 level mesh refinement calculation which gave a smaller error took only 75 seconds, less than one half as much time. This inspite of the fact that a great deal of the grid was refined (see figure 6). For even relatively coarse grids, the mesh refinement calculation is more efficient than a uniform grid calculation with the same maximum error.

The next results use one level of refinement and a ratio grid sizes of 4, so that the finest grid is also $\frac{1}{4}$ the mesh width of the coarsest grid. The results in Table 2 show that, at least on a sequential machine, the additional overhead in the second level of refinement is more than outweighed by the reduced number of mesh points. This is hardly surprising, since the number of grids in both cases is fairly small.

h_c	levels	ratio	error_∞	error tol	time (secs)
.1	3	2	3.04E-3	3.10E-3	13
.05	3	2	6.83E-4	6.94E-4	75
.1	2	4	3.04E-3	3.10E-3	16
.05	2	4	5.77E-4	6.94E-4	158

Table 2: Results for mesh refinement with 2 and 3 levels for problem 1. h_c is the mesh width of the coarsest grid.

We now illustrate the results of section 5. In each of the following cases, we iterated until the l_2 norm of the difference in two successive solutions was less than 10^{-5}.

h_c	levels	ratio	error_∞	error tol	time (secs)	iterations
.1	3	2	3.80E-3	3.10E-3	108	11
.05	3	2	4.43E-4	6.94E-4	401	7
.1	2	4	3.30E-3	3.10E-3	72	5
.05	2	4	8.17E-4	6.94E-4	714	5

Table 3: Results for mesh refinement with iterations between coarse and fine grids for problem 1.

The increase in the error over the single iteration results in 3 of the 4 cases is an artifact of the computation. In fact, the graphs of the error in the solution show that the error in the solution with mesh refinement becomes more similar to the error with a uniform grid with each iteration (note the differences between Figures 4 and 5). In all cases, the error is less than the bound from the theory (though since $u_{xxxx} = u_{yyyy} = 4800$ at $x = y = \frac{1}{2}$, the bound is quite large).

The second problem is $\nabla^2 u = 1$ on the L-shaped region in Figure 7.

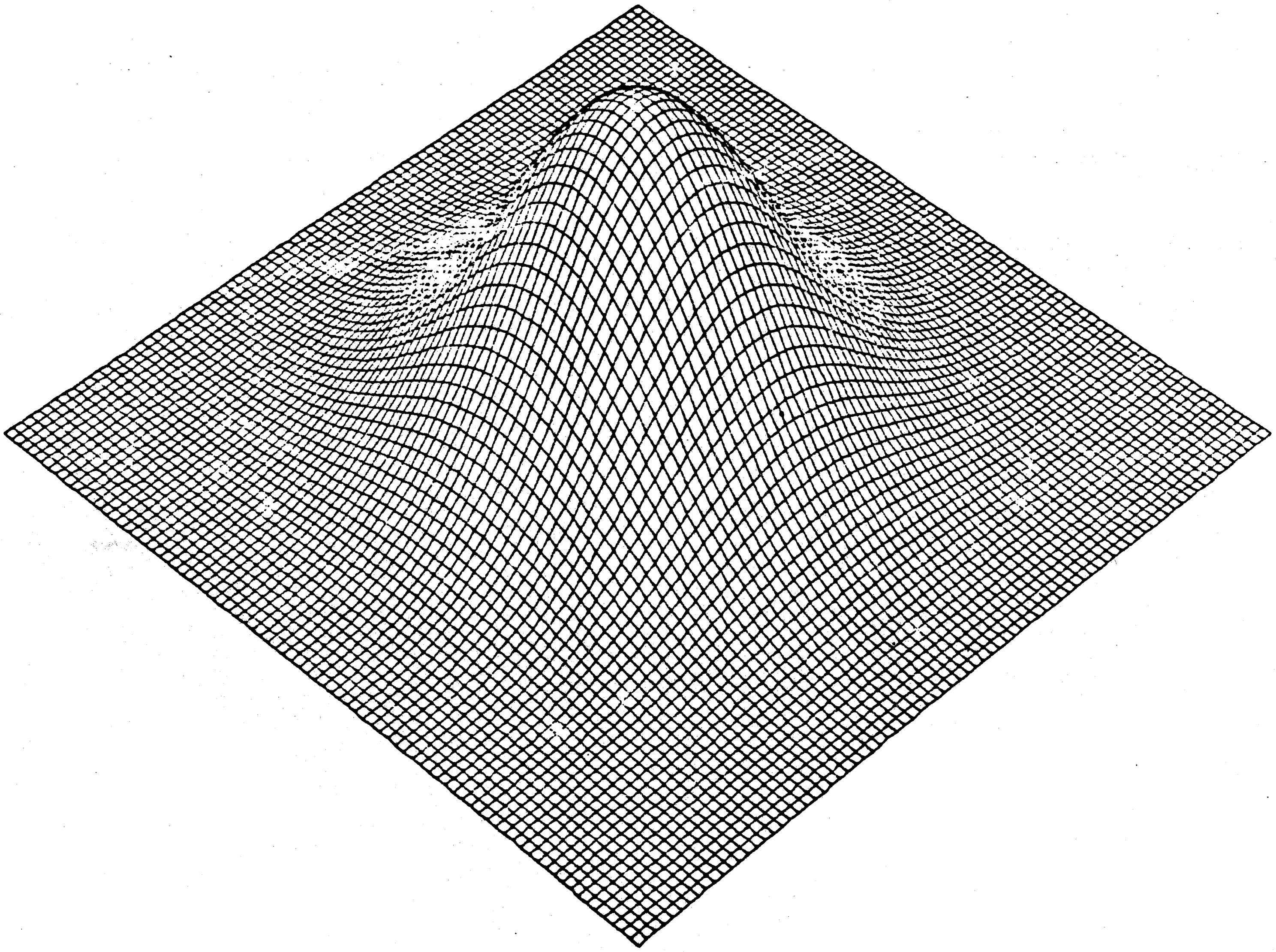

Figure 3: Solution to problem 1 with a uniform fine grid with step size 0.0125.

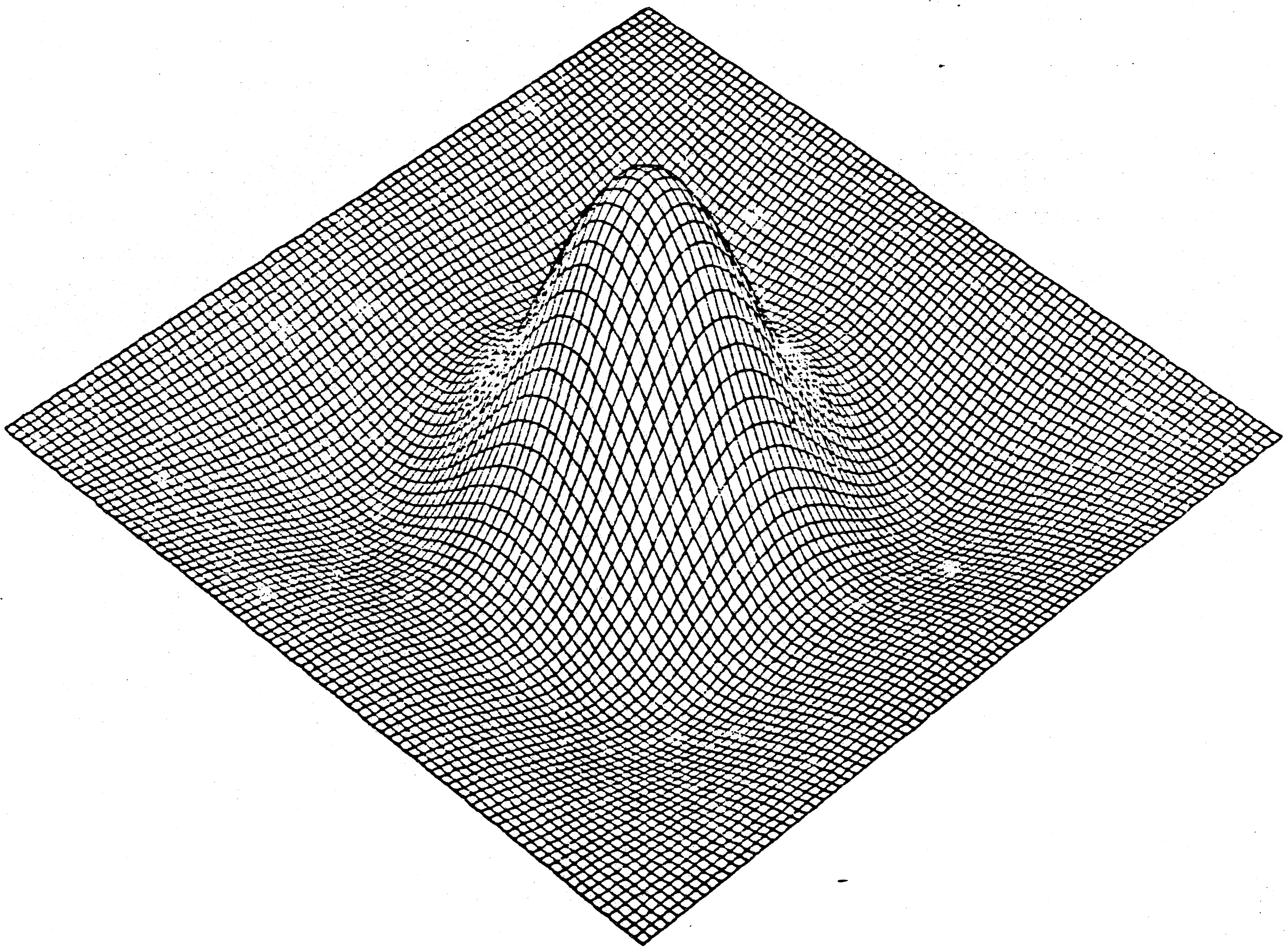

Figure 4: Error in the uniform grid solution to problem 1.

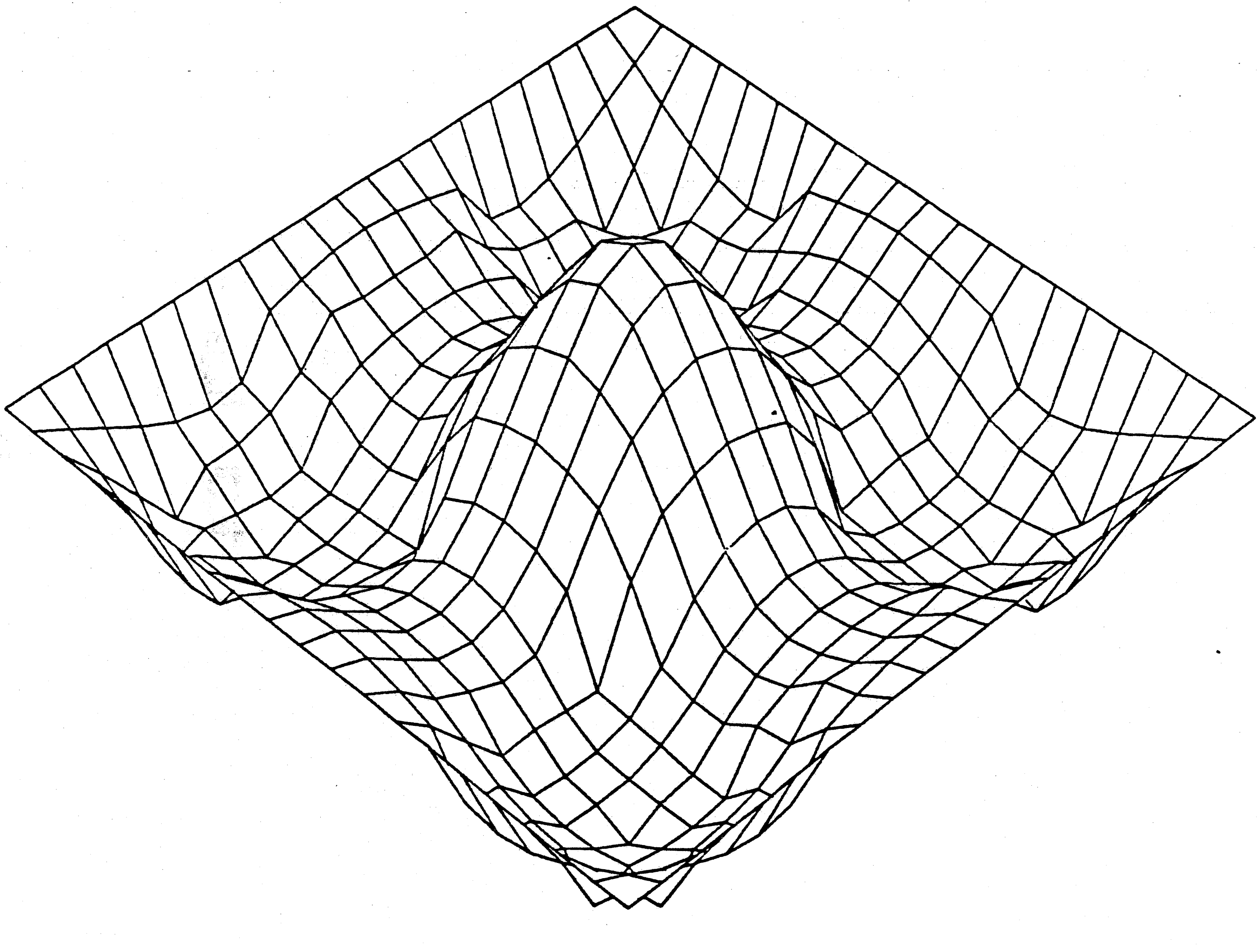

Figure 5: Error in mesh refinement solution to problem 1. $h_c = 0.05$, the ratio between levels was 2, and 3 levels were used. Only one iteration was taken.

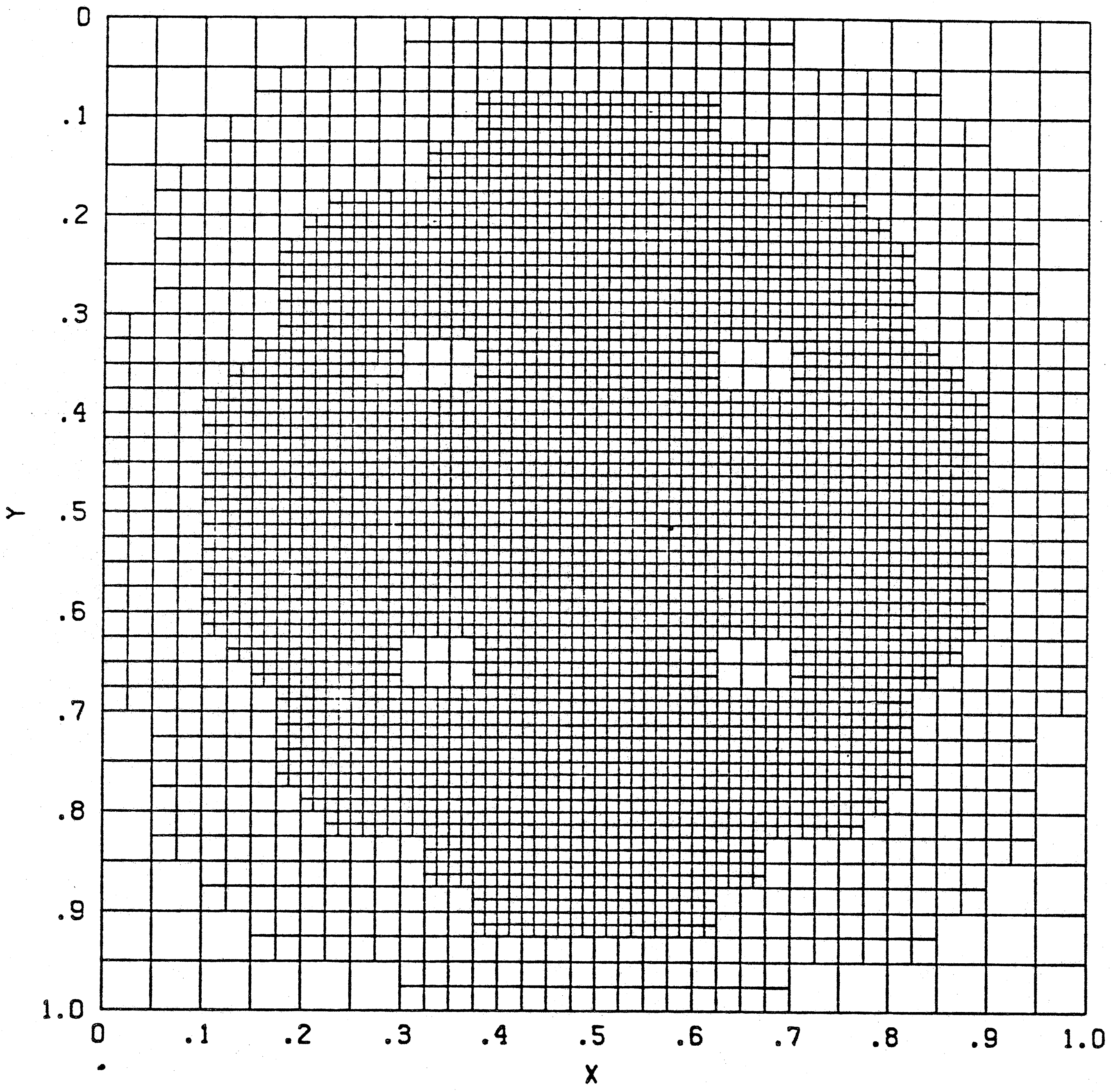

Figure 6: Grid for figure 5

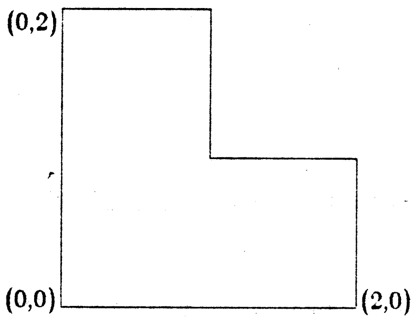

Figure 7: Domain for problem 2.

h_c	levels	time (secs)
.1	1	1.2
.05	1	7.8
.1	2	4.3
.1	3	18.

Table 4: Times for solutions of problem 2. The ratio between levels was 2. The single level results are for a uniform fine grid.

It is well known that the solution has singular first derivative at the interior corner. For this problem, we will see how well a mesh refinement computation can reproduce the results given by a uniform fine grid calculation (we have no exact solution for this problem, so errors are not computed).

We can see from Figures 8 and 9 and Table 4 that the mesh refinement solutions produce a good solution at significantly smaller cost. For example, a uniform coarse grid solution with $h = .05$ took 7.8 seconds, while a 2 level mesh refinement calculation whose finest grid had $h = .05$ and whose solution is quite similar to the uniform grid solutions (cf. Figures 8 and 9) took only 4.3 seconds. The grid for the mesh refinement calculation in Figure 9 is shown in Figure 10.

The final problem is a small disturbance transonic flow calculation over a thin circular arc airfoil. The equation is

$$\left(\beta - (\gamma + 1)M_{\infty}^2 \phi_x\right) \phi_{xx} + \phi_{yy} = 0$$

where $\beta = 1 - M_{\infty}^2$, γ is the ratio of specific heats (1.4 in our examples), and M_{∞} is the Mach number of the flow at infinity. The exact solution of this problem is not known.

Murman-Cole differencing is used [10], and the nonlinear problem is solved by lagging the ϕ_x term and iterating. This iteration to solve the nonlinear problem is combined with the iteration over the grids; the iteration is stopped when the change in two successive solutions is less than 10^{-4} The domain was $-2.5 \leq x \leq 2.5$, $0 \leq y \leq 2.0$. $M_{\infty} = 0.9$, and the thickness ratio of the airfoil was 0.12. The coarsest grid is uniform (the step sizes in x and y are constant).

We can see from Figures 11 and 12 that the solutions are quite similar. Comparisions of the pressure over the airfoil also show the similarity of the solutions. As the mesh refinement solution took less than one third the time (cf. Table 5), this approach is clearly suitable for problems of this kind.

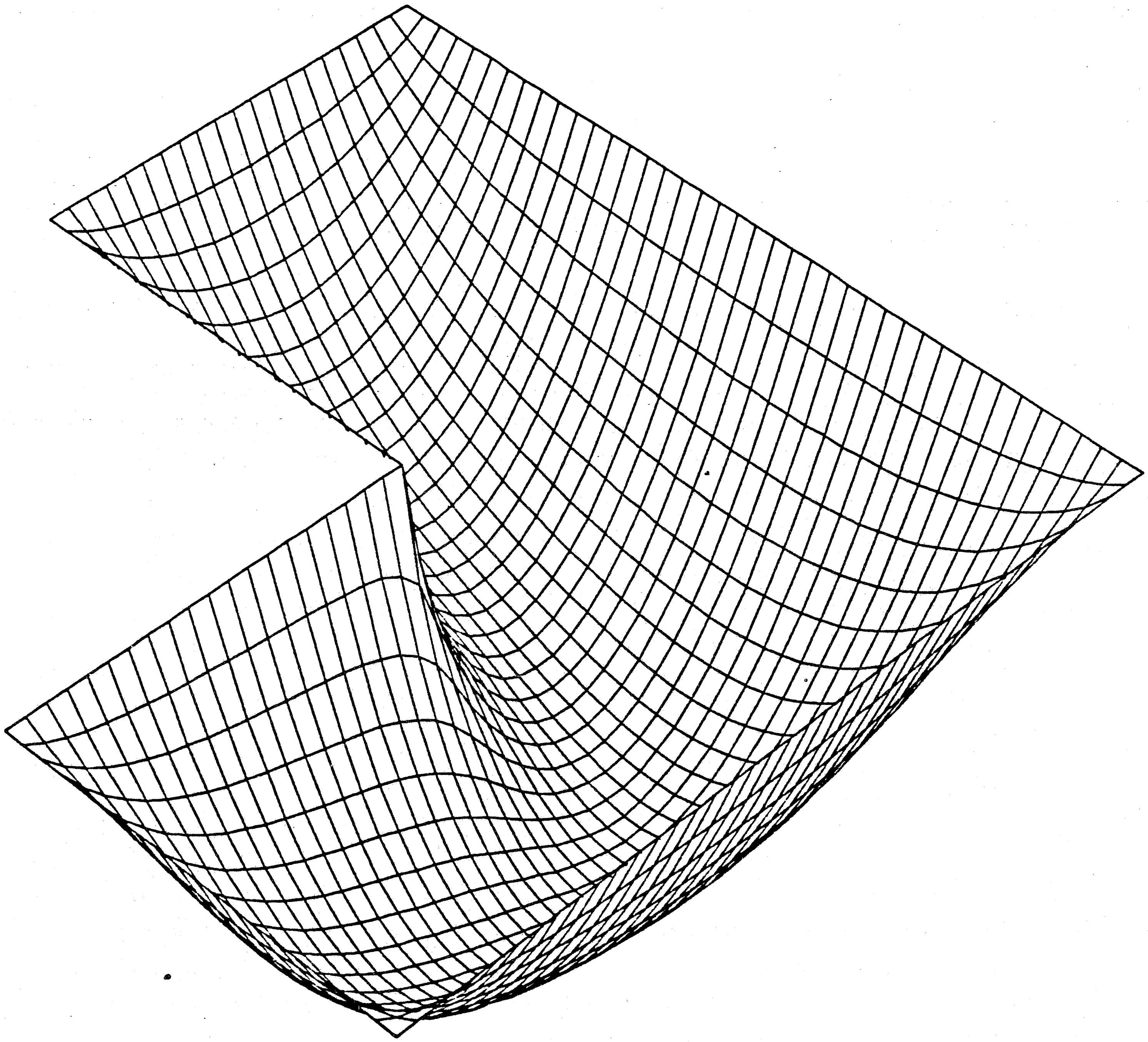

Figure 8: Solution to problem 2 with a uniform grid with step size 0.05.

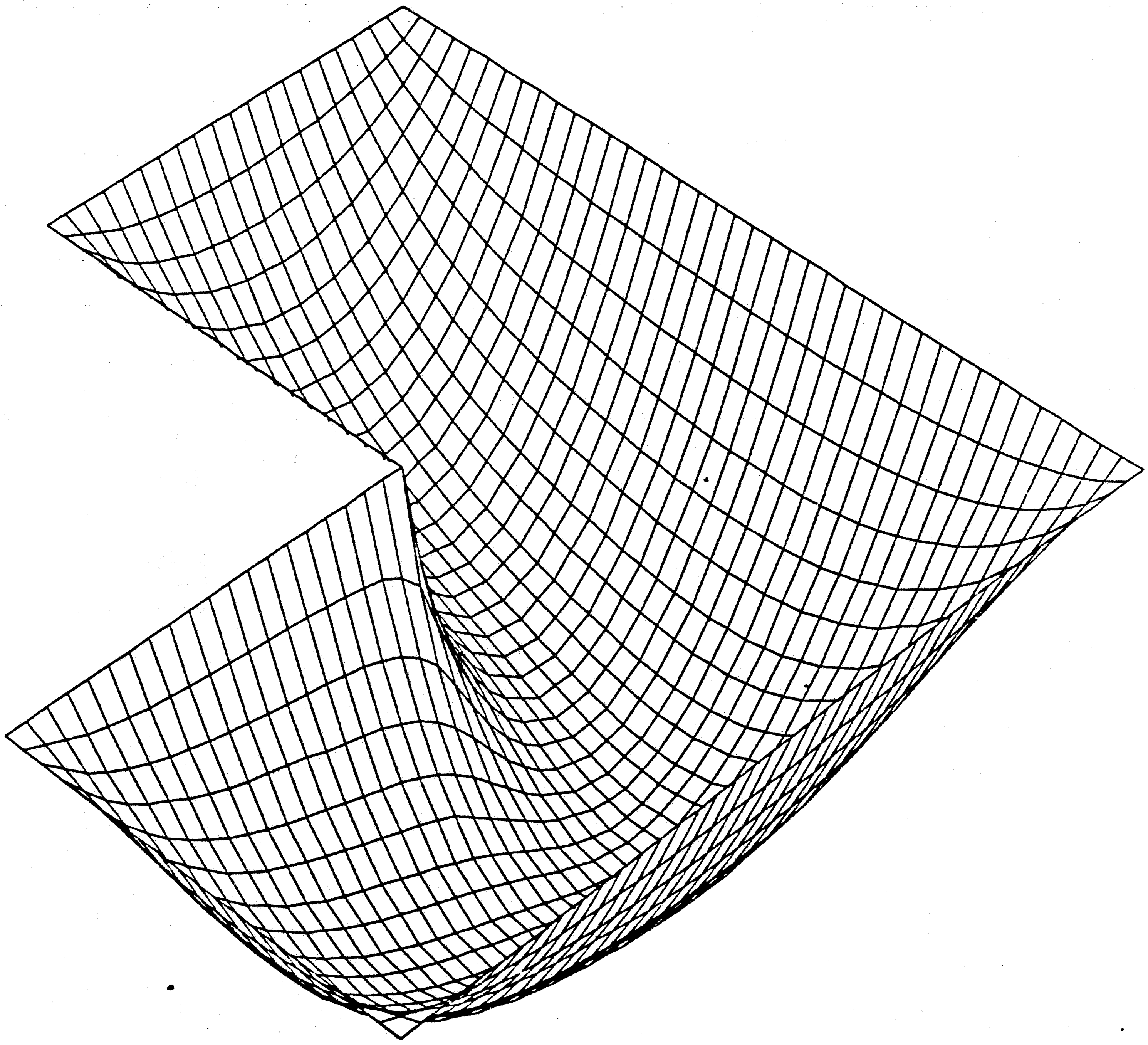

Figure 9: Solution to problem 2 with 2 levels of refinement; the finest grid has step size 0.05.

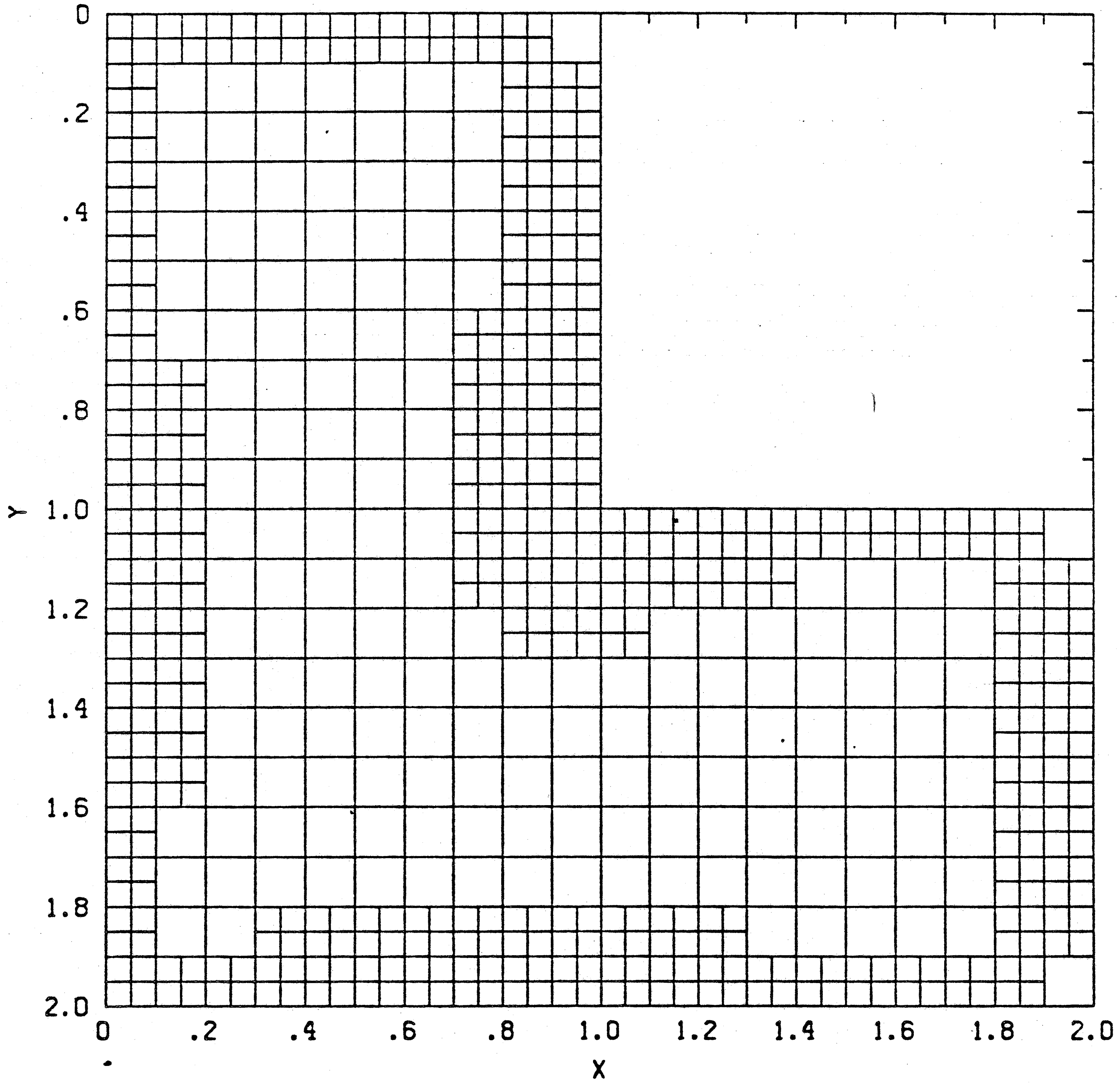

Figure 10: Grid for figure 9.

h_c	levels	time (secs)	iterations
.1	1	113	5
.05	1	3152	10
.1	2	940	13

Table 5: Times for solutions of problem 3. The ratio between levels was 2. The single level results are for a uniform fine grid.

Acknowledgements

I would like to thank M. Schultz and M. Smooke for several helpful discussions.

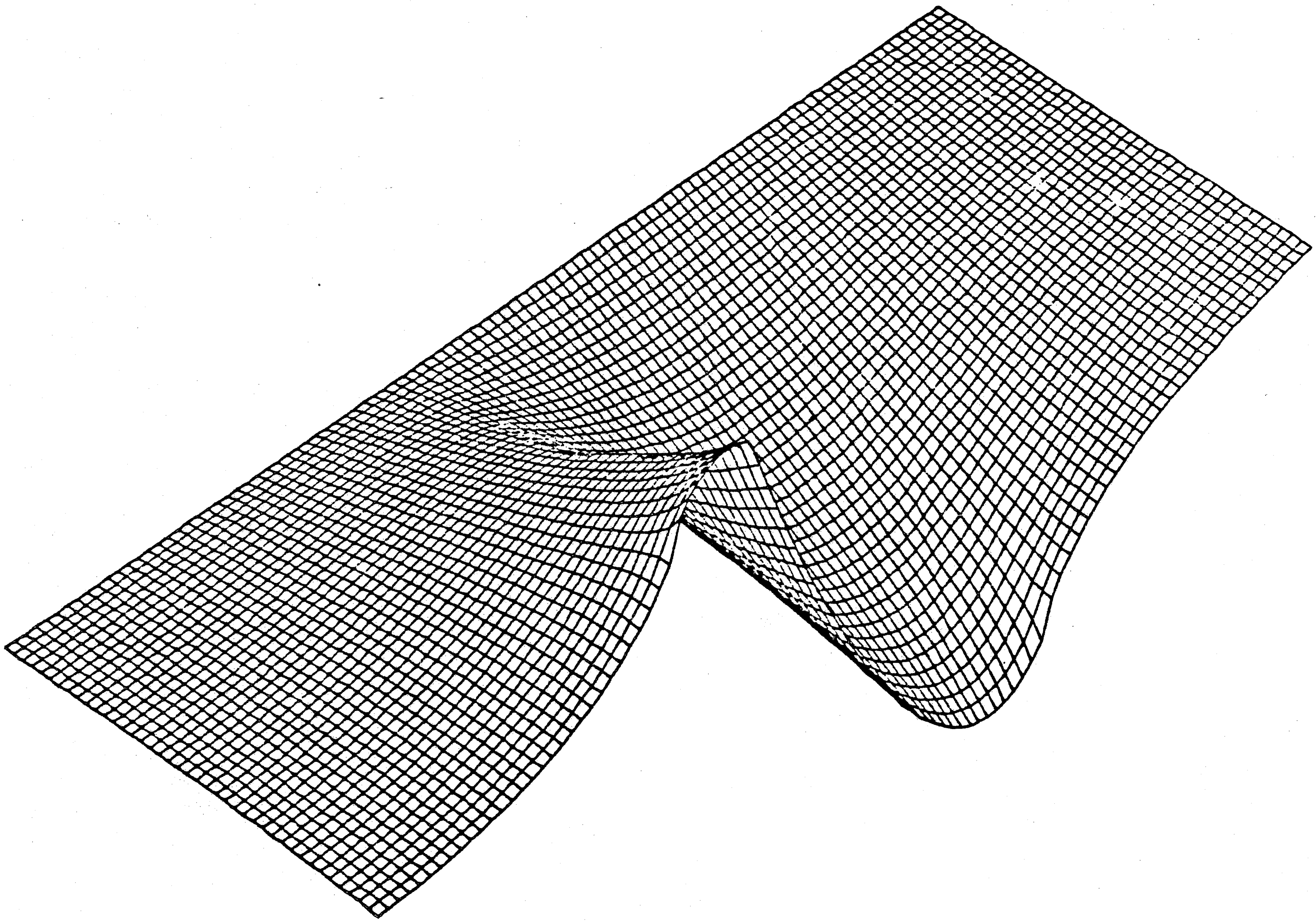

Figure 11: Solution to problem 3 with a uniform grid with step size 0.05.

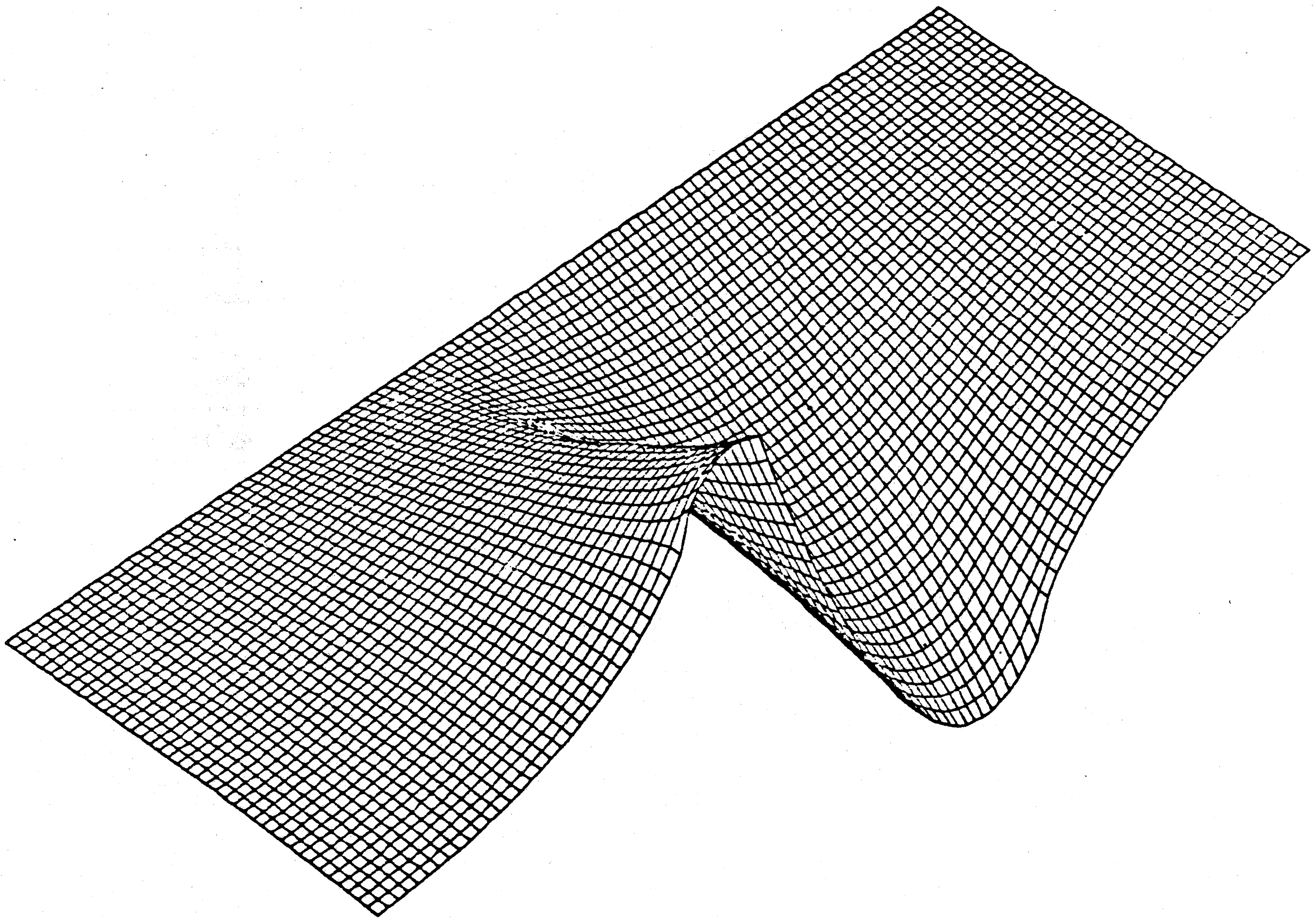

Figure 12: Solution to problem 3 with 2 levels of refinement; the finest grid has step size 0.05.

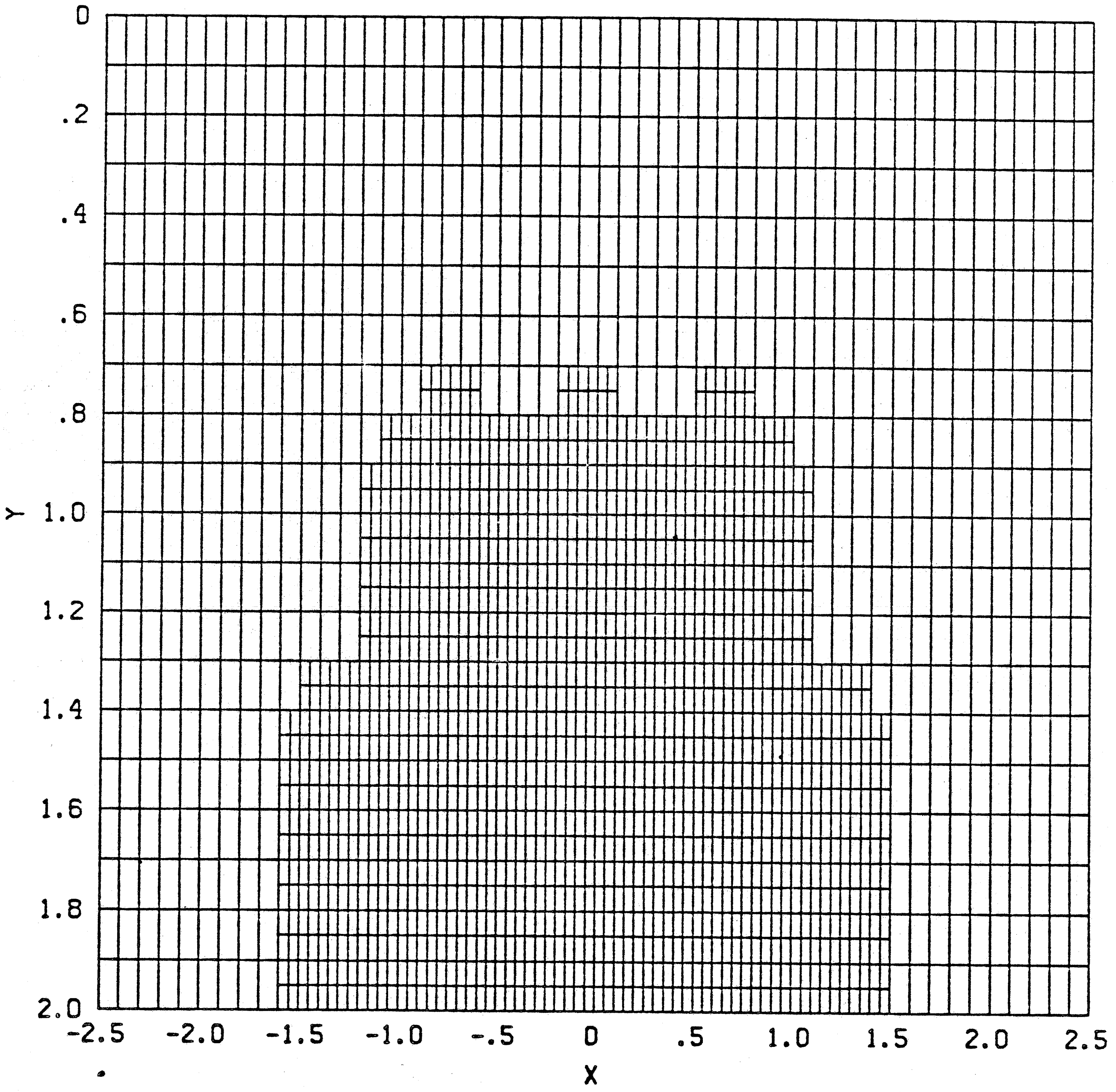

Figure 13: Grid for figure 12

Bibliography

[1] I. Babuska and W. C. Rheinboldt, "Error Estimates for Adaptive Finite Element Computations," *SIAM J. Num. Anal.*, 15 (1978), pp. 736–754.

[2] I. Babuska and W. C. Rheinboldt, "A Posteriori Error Analysis of Finite Element Solutions for One-Dimensional Problems," *SIAM J. Num. Anal.*, 18 (1981), pp. 565–589.

[3] M. J. Berger, *Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations*, Ph.D Thesis, Stanford University, 1982.

[4] M. J. Berger and J. Oliger, *"Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations,"* Stanford University Numerical Analysis Project NA-83-02, 1983.

[5] John H. Bolstad, *An Adaptive Finite Difference Method for Hyperbolic Systems in One Space Dimension*, Ph.D Thesis, Stanford University, 1982.

[6] S. Eisenstat, H. Elman, M. Schultz, and A. Sherman, *"The (New) Yale Sparse Matrix Package,"* Yale University Department of Computer Science, Research report 265, 1983.

[7] W. D. Gropp, "A test of moving mesh refinement for 2-D scalar problems," *SIAM J. Sci. Stat. Comp.*, 1 (1980), pp. 191–197.

[8] W. D. Gropp and J. Oliger, *"Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations: Convergence Theory and Error Estimation,"* in preparation.

[9] T. Meis and U. Marcowitz, *Numerical Solution of Partial Differential Equations*, Springer-Verlag, New York, 1981.

[10] E. Murman and J. Cole, "Calculation of Plane Steady Transonic Flows," *AIAA J.*, 9 (1971), pp. 114–121.

[11] R. S. Varga, *Matrix Iterative Analysis*, Prentice Hall, New Jersey, 1962.

[12] A. Weiser, *Local-Mesh, Local-Order, Adaptive Finite Element Methods with A Posteriori Error Estimators for Elliptic Partial Differential Equations*, Ph.D Thesis, Yale University, 1981.

[13] D. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971.