JOURNAL OF COMPUTATIONAL PHYSICS 100, 246–252 (1992)

# Use of Eigenfunctions in the Optimization of the Grid for the Boundary Element Method

AMBAR K. MITRA

*Department of Engineering Science and Mechanics, Iowa State University, Ames, Iowa, 50011*

Received January 31, 1990; revised May 24, 1991

A scheme for grid optimization for the boundary element method is developed. The scheme utilizes the functional behavior of unknowns along the boundary. The functional behavior of these unknowns is estimated by a local eigenfunction analysis performed at certain critical points on the boundary. Based on this eigenfunction analysis, the user can construct an optimal grid with certain tolerance, in a preprocessor, before the integral equations are solved. The relationships between this tolerance and two error norms are established through numerical experiments. It is shown that the tolerance provides excellent upper bounds for the error norms. As a result, the user gets an idea about the quality of the solution even before the integral equations are solved.


## INTRODUCTION

Scientists and engineers have acquired the ability to solve partial differential equations in domains of complex shape through the high state of development of numerical methods. The construction of a suitable grid is the initial step in any such method. In the boundary element method (BEM), only the boundary of the domain is discretized. Since certain conditions are specified at the boundary, a natural question one might ask is "can the specified conditions be utilized to construct an optimal grid?" However, for the domain methods, e.g., finite element and finite difference, one does not have the opportunity of asking this question. It is shown in this paper that an optimal grid can be constructed from the knowledge of the boundary conditions. The scheme is developed for the Laplace problem in two dimensions posed within a domain with straight boundaries. Although the application of the scheme for curved boundaries is not clear at this stage, future developments for such applications are anticipated.

The philosophy of smart algorithms has been summarized by Oden [1] through few key questions:

(1) How reliable are the computed results?

(2) How does one know that a mesh will produce a reasonable solution?

(3) What can be done to enhance the accuracy?

(4) How can the best possible results be obtained for the least effort?

By answering these questions, we now compare the merits of the proposed method with that of the existing methods.

The first question is usually answered indirectly by solving a problem in a "good" mesh, and in a "better" mesh. If the first $n$ digits in the numbers in these two sets of solutions are same, then one assumes the solution to be accurate up to $n$ digits. Through numerical experiments, using the present scheme, it can be shown that one can provide an upper bound for certain error norms, even before the discretized equations are solved. The user then knows, beforehand, what to expect and can estimate how much computational effort is required to attain a certain level of accuracy.

In the existing methods, the second question can only be answered after the problem is solved at least once and a local error indicator is calculated. In the BEM, the unknown function is expressed as an integral. In this integral, the products of the boundary values and certain kernel functions appear as the integrand [2]. When expressed in this form, the differential equation is exactly satisfied. The boundary conditions need be imposed for the solution of a particular problem. The more rigidly one enforces the boundary conditions, the closer one gets to the desired solution. In the present scheme, one knows, beforehand, how appropriate the grid is, by estimating how closely the boundary conditions have been enforced.

For the existing methods, one can answer the third question by estimating the local error indicators and deciding where the grid requires refinement, and how much refinement is necessary. This makes the process of creating the optimal grid iterative [3–8] and, consequently, expensive. In the present method, the optimal grid is generated in a pre-processor, even before the discretized equations are solved. This makes the procedure less expensive, gives the user the freedom to select an appropriate grid according to



0021-9991/92 $5.00

his needs and resources, and keeps the short optimization process uncoupled from the lengthy solution process.

The answer to the fourth question, as it pertains to the BEM, is shown in this paper to be very simple. The boundary conditions are utilized to develop eigenfunction expansions for all the relevant functions. The lengths of the boundary elements are determined appropriately to best describe the variation of the functions, dictated by the eigenfunctions, along the boundary by utilizing the Lagrangian approximation polynomials.

In the following sections, the BEM is briefly introduced, the algorithm for grid optimization is described, and its utility and advantages are shown through three example problems.

## BEM FORMULATION

Let $u$ be the potential which satisfies the equation

$$\nabla^2 u(\mathbf{x}) = 0 \tag{1}$$

in a polygonal domain $D$, shown in Fig. 1, with boundary $S$. On some portions of the boundary, marked by $U$ in Fig. 1, the function $u$ is specified; and the flux $q(\mathbf{x})$ is specified on the portions marked by $Q$. The flux is defined as

$$q(\mathbf{x}) = \nabla u(\mathbf{x}) \cdot \mathbf{n}, \tag{2}$$

where $\mathbf{n}$ is the unit outward normal on $S$. The function $u(\mathbf{z})$ at any point $\mathbf{z}$ in $D+S$ is related to the boundary values through the integral equation

$$c(\mathbf{z})\, u(\mathbf{z}) = \int_S [H(\mathbf{x}, \mathbf{z})\, u(\mathbf{x}) - G(\mathbf{x}, \mathbf{z})\, q(\mathbf{x})]\, ds(\mathbf{x}). \tag{3}$$

For problems in two dimensions,

$$G(\mathbf{x}, \mathbf{z}) = \ln |\mathbf{x} - \mathbf{z}|, \qquad H(\mathbf{x}, \mathbf{z}) = \nabla_x G \cdot \mathbf{n}(\mathbf{x}), \tag{4}$$

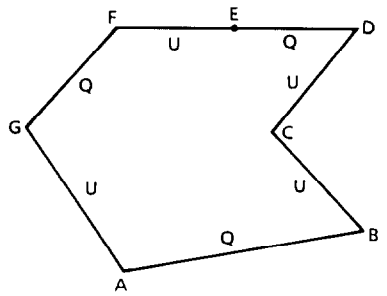


FIG. 1. The polynomial domain.

and $c(\mathbf{z})$ is known to be

$$c(\mathbf{z}) = 2\pi, \quad \mathbf{z} \text{ in } D; \qquad = \omega, \quad \mathbf{z} \text{ on } S \tag{5}$$

where $\omega$ is the angle between two tangents to $S$ drawn on either side of $\mathbf{z}$. The procedure for discretizing the integral equation can be found in a text on BEM [9].

In the present work, the approximating functions for representing $q(\mathbf{x})$ and $u(\mathbf{x})$ along the boundary are assumed to be linear. The main idea can easily be extended to higher order polynomials. However, the linear approximation has been found to yield excellent solutions. Although the order of the approximating polynomials is fixed, the optimal lengths of the elements along the boundary are determined through a minimization process. This method is somewhat like the $r$-method for grid optimization.

The minimization process can only be initiated if one has some knowledge about the behavior of the functions $u$ and $q$ along the boundary. It has long been recognized that sharp corners and points across which the type of the boundary condition changes from Dirichlet to Neumann have a strong influence on the behavior on the potential and flux. It has been found that singularities are often associated with these critical points. This singular behavior often corrupts the quality of the solution. In the present algorithm, these critical points will play the central role. The functional behavior of $u$ and $q$ near each of these critical points can be obtained by performing a local eigenfunction analysis.

## FUNCTIONAL BEHAVIOR OF *u* AND *q*

In Fig. 2a, one critical point of our polygonal domain is shown. By erecting a local polar coordinate system with the origin at the critical point, the solution of the Laplace equation can be written as

$$u = \sum_{\lambda} C_\lambda r^\lambda \cos \lambda\theta + S_\lambda r^\lambda \sin \lambda\theta. \tag{6}$$

The constants $C_\lambda$ and $S_\lambda$ are unimportant in the present analysis. However, the eigenvalues, $\lambda$, are utilized in the algorithm. The eigenfunctions and eigenvalues for $q$ can easily be obtained from Eq. (6). If one enforces the boundary conditions on the segments $AB$ (at $\theta = 0$) and $AG$ (at $\theta = \alpha$), the set of eigenvalues can be determined. One must repeat this process for all the critical points. It is evident that the eigenvalues depend on the boundary conditions on the adjacent segments and the angle $\alpha$ subtended at the critical point. It can now be argued that the part of the segment next to $A$ is dominated by the eigenfunctions

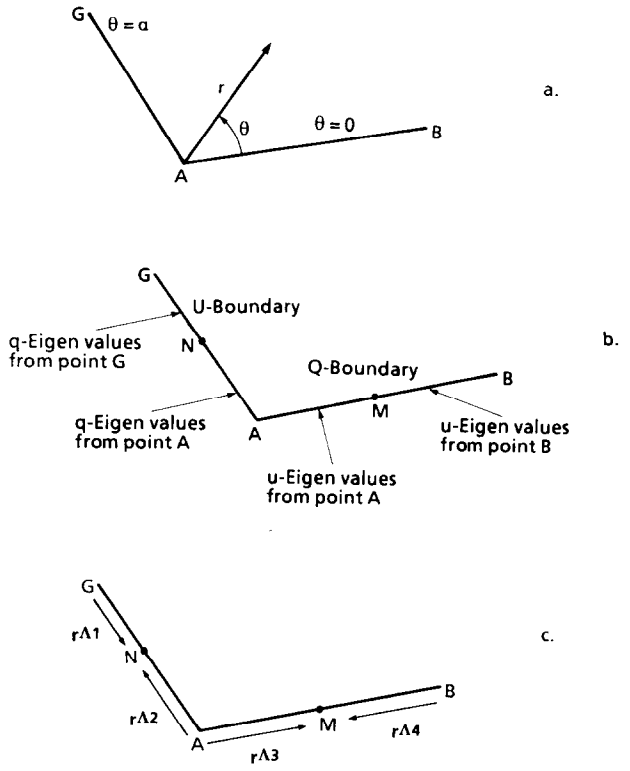


**FIG. 2.** Schematics for the eigenfunction analysis: (a) the critical point and local coordinate system; (b) the relevant eigenvalues; (c) the functional behavior.

originating at $A$, and the rest of the segment is dominated by the eigenfunctions originating at $B$. However, the ranges of influence of these two sets of eigenfunctions are not known. For simplicity, the segment $AB$ is divided into two equal subsegments, where the subsegment next to $A$ is influenced by $A$, and the other by $B$.

Considering each subsegment, one pays attention to the eigenvalues for $u$ if the boundary condition on the subsegment is $Q$-type (Neumann), and vice versa. This rule is schematically shown in Fig. 2b where the mid-points of $AB$ and $AG$ are respectively denoted by $M$ and $N$. This step is a rather intuitive one, and the reason being that on subsegment $AM$ one must determine $u$ from the discretized integral equation, and consequently one must have a grid that is good for such a determination. Similarly, on the subsegment $AN$, $q$ is the unknown, and the grid should be good for this unknown.

In the next step, from the set of eigenvalues identified for a subsegment, the smallest eigenvalue which is different from 0 or 1 is chosen. Since linear approximating functions are being used, the constant and linear variations are exactly represented, and one picks the first eigenfunction which cannot be represented exactly by the linear approximating functions. When quadratic approximating functions are used, one must pick the smallest eigenvalue which is different from 0, 1, and 2. In this manner, one eigenfunction can be associated with each of the subsegments. This step is schematically shown in Fig. 2c. In this figure, the chosen eigenvalues are denoted by $\Lambda_1$, $\Lambda_2$, etc.

In the following section, the procedure for dividing a subsegment is described. A minimization process is employed for this purpose. This process ensures that the lengths of the linear elements are best suited for the variation $r^{\Lambda}$, where $\Lambda$ is the chosen eigenvalue for the subsegment.

## MINIMIZATION PROCEDURE

At the beginning of the minimization process the length of a subsegment is normalized to 1. Such a subsegment is then partitioned into $n$-number of elements of lengths $d_1, d_2, ..., d_n$. The procedure for fixing the value of $n$ will be described later in this section. The function $r^{\Lambda}$ is then approximated by employing piecewise linear interpolation over each element. The differences in the areas under the curve $r^{\Lambda}$, and the assembly of trapezoids under the approximate representation are squared and summed to form a functional. This functional is then minimized to obtain the lengths $d_1, d_2, ..., d_n$. The IMSL subroutine UNLSF is used fo this purpose. After the lengths of the elements are obtained, the difference in area, $\Sigma$, between the smooth and piecewise curves is obtained to test the closeness of the approximate representation. The computation time for this process is less than 1 s on a VAX/VMS machine.

It must be mentioned that the number of elements, $n$, is tied to the closeness of the fit. One usually starts off with a desired level of accuracy $\varepsilon$, and a small value of $n$ (say 2). The value of $n$ is increased until $\Sigma \leqslant \varepsilon$.

As mentioned before, the length of a subsegment is normalized to one, and the minimization is performed over an interval $[0, 1]$. However, if $\Lambda$ is negative, a singularity will appear at $r = 0$, and in that case, fitting a linear piece to a point at infinity would be meaningless. In such a situation, the range is modified to $[\sigma, 1]$, where $\sigma \ll 1$. In general $0 \leqslant \sigma \ll 1$, where $\sigma = 0$ corresponds to the non-singular case.

For a singular case, the description of the singular behavior improves as the value of $\sigma$ is reduced. For a fixed $\varepsilon$, the reduction of $\sigma$ below 0.01 causes the value of $n$ to increase rapidly. A large number of very small elements packed near the singularity makes the computation expensive, and also the conditioning of the coefficient matrix deteriorates. The author's experiments with different values of $\sigma$ show that $\sigma = 0.01$ is a good choice.

After performing the minimization in a singular case, the non-zero $\sigma$ is added to the length of the element farthest away from the singularity while maintaining the fineness of the grid near the singularity.

## NUMERICAL EXAMPLES

Three example problems are considered in this section. The chosen problems have either analytical or published numerical solutions available for comparison.

In order to gage the accuracy of the BEM solutions, two different kinds of error norms are calculated. The first error norm is based on the "integrated absolute boundary error" which will be denoted by $\delta$. This error norm is defined as

$$\delta = \int |f_{\text{exact}} - f_{\text{BEM}}| \, dS. \tag{7}$$

In Eq. (7), $f$ is the unknown on the boundary, i.e., $f = u$ on a $Q$-type boundary and vice versa, and the subscript BEM denotes the BEM solution. The integral over $dS$ denotes the integration along the boundary and is obtained by calculating the absolute error at all the nodes and employing the trapezoidal rule. It must be mentioned at this point that $f_{\text{exact}}$ is in general unknown, and the norm $\delta$ is not obtainable. In this work, $\delta$ is calculated in model test problems to illustrate the performance of the scheme and to establish a relationship between $\varepsilon$ and $\delta$.

The second error norm is based on the "integrated relative domain error" which will be denoted by $\Delta$. This error norm is defined as

$$\Delta = \int \frac{|u_{\text{exact}} - u_{\text{BEM}}|}{|u_{\text{exact}}|} \, dD \tag{8}$$

The integration in Eq. (8) over the domain $D$ is performed by calculating the relative error at uniformly distributed points and employing one point integration over square shaped area patches. In general, $u_{\text{exact}}$ is unknown and the norm $\Delta$ is not obtainable. The norm $\Delta$ is calculated for model example problems to establish a relationship between $\varepsilon$ and $\Delta$.

EXAMPLE 1. The problem is posed in a $1 \times 1$ square as shown in Fig. 3a. The boundary conditions are consistent with the exact solution

$$u = \cosh x \sin y. \tag{9}$$

In Fig. 3b, the chosen eigenfunctions approaching from the two ends of each side of the square domain are shown. There is no singular point on the boundary, and consequently $\sigma$ is zero for all subsegments. In Table I, the number of elements required on each side of the square are given for three values of the tolerance $\varepsilon$.

In order to provide the reader with an idea about the size of the elements, the lengths of the elements along the segment $BC$ (starting from the edge $B$) are given below for the tolerance $\varepsilon = 0.05$:

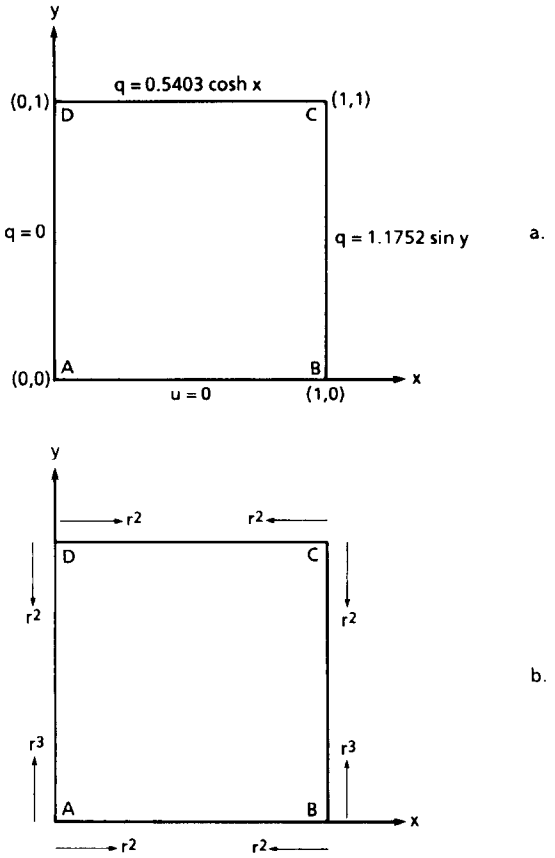


FIG. 3. Example 1: (a) the definition of the problem; (b) functional behavior.

0.1690 0.1236 0.1082 0.0992 0.1667 0.1667 0.1667.

The first four elements are in accordance with the $r^3$ variation, and the size of these elements gets smaller as we move away from the edge $B$ while the $r^3$ function increases rather sharply. The last three elements correspond to the $r^2$ variation, and they are all equal in length. It has been observed that $r^2$ variation forces a uniform distribution.

The problem is solved three times with the grids shown in Table I, and the error norm $\delta$ and $\Delta$ are calculated. The

**TABLE I**

Distribution of Elements for Various Values of Tolerance $\varepsilon$ for Example 1

| $\varepsilon$ | Side $AB$ | Side $BC$ | Side $CD$ | Side $DA$ |
|---|---|---|---|---|
| 0.05 | 3 | 7 | 3 | 7 |
| 0.02 | 5 | 12 | 5 | 12 |
| 0.005 | 10 | 23 | 10 | 23 |

**TABLE II**

Error Norms for Various Values of Tolerance $\varepsilon$ for Example 1

| $\varepsilon$ | $\delta$ | $\Delta$ |
|---|---|---|
| 0.05 | $1.958 \times 10^{-2}$ | $8.553 \times 10^{-3}$ |
| 0.02 | $7.704 \times 10^{-3}$ | $3.541 \times 10^{-3}$ |
| 0.005 | $2.046 \times 10^{-3}$ | $8.497 \times 10^{-4}$ |

values of the norms for various values of the tolerance $\varepsilon$ are shown in Table II.

A close examination of the data in Table II makes it tempting to fit a linear relationship between the tolerance and error norms. Such relationships are given as

$$\delta = k\varepsilon, \qquad \Delta = K\varepsilon. \tag{10}$$

It is natural to anticipate that the proportionality constants $k$ and $K$ would be dependent on the shape of the domain. But if the values of $k$ and $K$ are less than one, then it is clear that the tolerance $\varepsilon$ that is enforced in the pre-processor for grid optimization provides an excellent upper bound for the errors. A crude calculation reveals that $k = 0.39$ and $K = 0.17$. Another fact which will be given as a mere observation is that the absolute boundary error (the integrand in Eq. (7)) was found to be uniform over all the nodes. This probably means that a good grid distributes the error uniformly along the boundary.

As mentioned before, the norms $\delta$ and $\Delta$ are usually not obtainable. However, the value of the tolerance $\varepsilon$ can be enforced at the optimization stage, and then one can claim that $\Delta$ and $\delta$ are at least one order smaller than the tolerance.

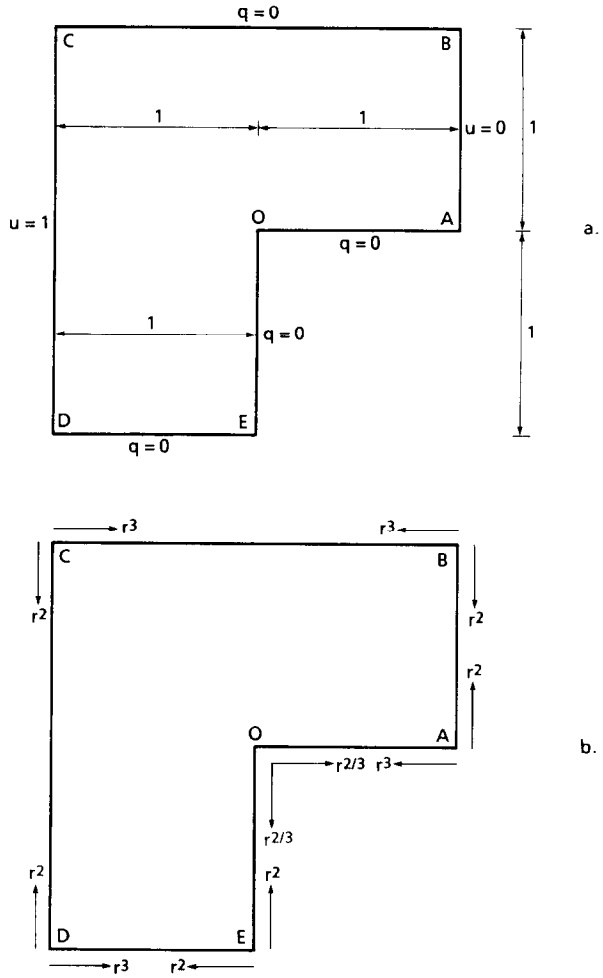


FIG. 4. Example 2: (a) the definition of the problem; (b) functional behavior.

EXAMPLE 2. This problem posed in Fig. 4a shows that the boundary has a re-entrant corner at $O$. For this example, the results obtained by Papamichael and Whiteman [10] by using a numerical conformal transformation method are available for comparison. In Fig. 4b, the chosen eigenfunctions approaching from the two edges of each side are shown.

The lengths of 13 elements that are placed on the side $OA$ are given below for the tolerance $\varepsilon = 0.01$:

0.0433 0.1031 0.1533 0.2002 0.0426 0.0440 0.0457

0.0477 0.0502 0.0534 0.0582 0.0672 0.0910

The first four elements are consistent with $r^{2/3}$ and are crowded toward point $O$; the last nine elements are consistent with $r^3$ and are crowded toward the center of the side $OA$.

This example problem was solved three times for three different tolerance values, and the error norm $\Delta$ was calculated. For this calculation, Papamichael and Whiteman's [10] solution was used as $u_{\text{exact}}$, and 75 square-shaped area patches were used for the integration over the domain $D$. For this problem it was impossible to calculated $\delta$, because the $f_{\text{exact}}$ on the boundary is not available. In Table III, the variation of $\Delta$ with $\varepsilon$ is shown. Also shown in this table are the number of boundary nodes that have been required in each of the cases. Assuming a linear relationship between $\varepsilon$ and $\Delta$, we find the constant of proportionality $K$ to be 0.06. This further established that the tolerance is, in

**TABLE III**

Domain Error for Various Tolerances for Example 2

| $\varepsilon$ | $\Delta$ | No. of Nodes |
|---|---|---|
| 0.02 | $1.247 \times 10^{-3}$ | 66 |
| 0.01 | $5.281 \times 10^{-4}$ | 82 |
| 0.005 | $2.795 \times 10^{-4}$ | 130 |

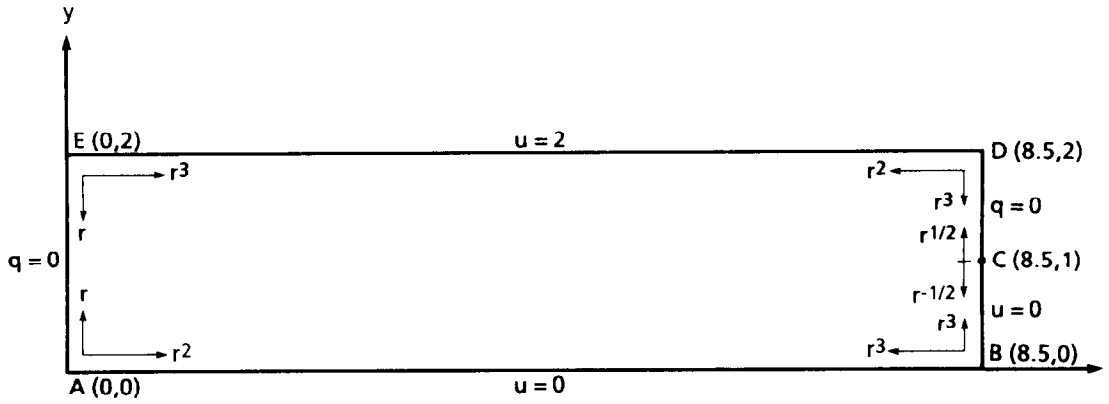


**FIG. 5.** The definition of problem in Example 3, and the functional behavior.

fact, an excellent bound for the "integrated relative domain error."

EXAMPLE 3. In this example, the computational effort involved in the present method is compared with the effort involved in a customary adaptive scheme based on the $p$-method [6]. For this purpose we have picked a problem which involves a singular point. The problem is posed in Fig. 5. The grid was developed for $\varepsilon = 0.02$, and the $\sigma$ on the singular subsegment adjacent to $C$ was chosen to be 0.01. In order to show the packing of nodes near the singularity at $C$, the lengths of elements on the side $BC$ (starting at edge $B$) are

| | | | | |
|---|---|---|---|---|
| 0.1101 | 0.0808 | 0.0706 | 0.0648 | 0.0608 |
| 0.0577 | 0.0553 | 0.1756 | 0.1192 | 0.0807 |
| 0.0530 | 0.0336 | 0.0203 | 0.0116 | 0.0062 |

The first seven lengths correspond to the behavior $r^3$ over the half of side $BC$ next to point $B$. The last eight lengths correspond to the behavior $r^{-1/2}$ over the rest of the side $BC$. The distribution of elements on different segments of the boundary are

Segment $AB$: 12
Segment $BC$: 15
Segment $CD$: 11
Segment $DE$: 5
Segment $EA$: 1

In Table IV, the present solution at a few selected points near the singularity, at (8.5, 1), is compared with two other solutions obtained by Chakravarty [6]. The solution in the second column was obtained on a uniform grid with quadratic elements by solving 293 equations. The solution in the third column was obtained by first solving the problem in a uniform grid with 77 nodes, then updating the grid using the $p$-method, and then solving the problem a second time on the improved grid. In the improved grid, 149 nodes were used.

The data in Table IV shows that, although the adaptive grid is four times as efficient compared to the uniform grid (number of nodes reduced to half), the present method is 36 times as efficient compared to the uniform grid (number of nodes reduced to $\frac{1}{6}$).

**TABLE IV**

Comparison of Present Method with $p$-Method of Grid Refinement [6]

| Coordinates | Uniform grid [6] 293 nodes | Adaptive grid [6] 149 nodes | Present method 49 nodes |
|---|---|---|---|
| (6, 0.5) | 0.4910 | 0.4910 | 0.4913 |
| (6, 1.0) | 0.9874 | 0.9874 | 0.9880 |
| (6, 1.5) | 1.4911 | 1.4911 | 1.4921 |
| (7, 0.5) | 0.4557 | 0.4557 | 0.4564 |
| (7, 1.0) | 0.9392 | 0.9392 | 0.9402 |
| (7, 1.5) | 1.4586 | 1.4585 | 1.4599 |
| (8, 0.5) | 0.2690 | 0.2689 | 0.2698 |
| (8, 1.0) | 0.6968 | 0.6965 | 0.6974 |
| (8, 1.5) | 1.3339 | 1.3337 | 1.3346 |

## CONCLUSION

The method for grid optimization using local eigenfunction expansions near critical points has been shown to be an excellent tool for grid generation. The advantages of the method are as follows:

The grid is optimized in a preprocessor even before the discretized integral equations are solved. In this non-iterative method, the only information that is required for grid generation is the shape of the domain and the boundary conditions. Because of the non-iterative nature of the algorithm, the computational effort involved in the optimization process is much less compared to the existing adaptive

iterative techniques. In the existing techniques, an additional task of the calculation of suitable error norms is to be performed.

The tolerance value that is chosen by the user at the time of grid generation provides an excellent upper bound for error in the variables on the boundary and in the interior of the domain. The user can estimate the error in the final solution while generating the grid, and before the integral equations are solved.

By generating the grids for several values of $\varepsilon$, the user can estimate how much computer time will be expended to attain different levels of accuracy. The user can then pick a grid depending on the available resources.

The method has been shown to perform better than the $p$-method of grid optimization. This observation has been validated by at least one example problem (Example 3).

Last and probably the most convenient aspect of the algorithm is that any existing BEM program with no grid optimization capability need not be modified to acquire such capability. The present optimization module can simply be executed before entering the main solution module.

## REFERENCES

1. J. T. Oden, in *Proceedings of the 21st Midwestern Mechanics Conference, Houghton, U.S.A., 1989*, edited by J. B. Ligon *et al.* (Michigan Technological University, Houghton, MI, 1989), p. 11.
2. G. Fairweather, F. J. Rizzo, D. J. Shippy, and Y. S. Wu, *J. Comput. Phys.* **31**, 96 (1979).
3. M. S. Ingber and A. K. Mitra, *Int. J. Num. Methods Eng.* **23**, 2121 (1986).
4. E. Rank, *Boundary Elements IX*, Vol. 1, edited by C. A. Brebbia *et al.* (Springer-Verlag, Berlin, 1987), p. 259.
5. A. K. Mitra and M. S. Ingber, *Int. J. Num. Methods Fluids* **8**, 327 (1988).
6. R. R. Chakravarty, Thesis, Iowa State University, Ames, Iowa, 1988 (unpublished).
7. E. Alarcon and A. Reverter, *Int. J. Num. Methods Eng.* **23**, 801 (1986).
8. J. J. Rencis and K-Y. Jong, *Comput. Methods Appl. Mech. Eng.* **73**, 295 (1989).
9. P. K. Banerjee, *Boundary Element Methods in Engineering Science* (McGraw–Hill, Maidenhead, England, 1981), pp. 54, 177.
10. N. Papamichael and J. R. Whiteman, *Z. Angew. Math. Phys.* **23**, 655 (1972).