1983

Collocation Software for Second Order Elliptic Partial Differential Equations

Elias N. Houstis
Purdue University, enh@cs.purdue.edu

W. F. Mitchell

John R. Rice
Purdue University, jrr@cs.purdue.edu

Report Number:

83-446

Houstis, Elias N.; Mitchell, W. F.; and Rice, John R., "Collocation Software for Second Order Elliptic Partial Differential Equations" (1983). *Department of Computer Science Technical Reports.* Paper 365.
https://docs.lib.purdue.edu/cstech/365

COLLOCATION SOFTWARE FOR SECOND ORDER ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

E.N. Houstis
Department of Mathematics
University of Thessaloniki
and
Department of Computer Science
University of South Carolina

W.F. Mitchell and J.R. Rice
Computer Science
Purdue University

CSD-TR 446, June 30, 1983

ABSTRACT

We consider the collocation method for linear, second order elliptic problems on rectangular and general two dimensional domains. An overview of the method is given for general domains followed by a discussion of the improved efficiencies and simplifications possible for rectangular domains. A very high level description is given of three specific collocation algorithms which use Hermite bicubic basic functions, (1) GENCOL (collocation on general two dimensional domains). (2) HERMCOL (collocation on rectangular domains) with general linear boundary conditions and (3) INTCOL (collocation on rectangular domains with uncoupled boundary conditions). The linear system from INTCOL has half the band width of that from HERMCOL which provides substantial benefit in solving the system. We provide some examples showing the range of applicability of the algorithms and some performance profiles illustrating their efficiency. Fortran implementation of these algorithms are given in the companion reports CSD-TR 444 and CSD-TR 445.

CONTENTS

COLLOCATION SOFTWARE FOR SECOND ORDER ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

E.N. Houstis[1], W.F. Mitchell[2] and J.R. Rice[2]

1. INTRODUCTION

We consider a two-dimensional partial differential equation (PDE)

$$Lu \equiv au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu = g \qquad (1.1)$$

over a two-dimensional domain R in the x,y plane. It is assumed that the coefficients a,b,c satisfy the ellipticity condition $b^2-4ac < 0$ and that R is defined by the boundary ∂R with clockwise orientation defined parametrically as follows:

$$\left.\begin{array}{lll} x = x_i(p) & \text{for} & b_{1i} \le p \le b_{2i} \\ y = y_i(p) & \text{for} & b_{1i} \le p \le b_{2i} \end{array}\right. \quad i=1,2,\ldots,\text{nbound} \qquad (1.2)$$

Thus R is the interior of the domain defined by these nbound pieces; R need not be simply connected, but we ignore that case in this paper in order to simplify the discussion. In the case of a rectangular domain $R = [ax,bx] \times [ay,by]$, the boundary ∂R is implicitly defined by the end points ax,bx,ay,by. Further, we assume that the solution u of (1.1) is subject to boundary conditions.

$$\Lambda u \equiv \alpha u + \beta u_x + \gamma u_y = \delta \quad \text{for} \quad (x,y) \in \partial R\ . \qquad (1.3)$$

All the coefficients and right sides in (1.1) and (1.3) may depend on x and y.

A large class of methods for approximating the solution u of (1.1), (1.3) involves first the partition of R into a finite element mesh Ω and second the determination of a piecewise polynomial approximation U defined over the partition Ω. This paper describes such a method, called collocation, and discusses specific implementations.

2. OVERVIEW OF THE COLLOCATION METHOD

Collocation is a finite element method to approximate the solution $u(x,y)$ of (1.1), (1.3) consisting of the following four general phases:

Phase 1: Overlay the domains of definition R by a rectangular grid G, identify the rectangular elements of G that are interior, boundary or exterior , and associate boundary pieces with boundary elements. Define Ω, the finite element mesh, as the union of the boundary elements $\partial\Omega$ and interior elements Ω'.

Phase 2: Approximate $u(x,y)$ by a bicubic Hermite piecewise polynomial $U(x,y)$ over the finite element mesh Ω.

---

[1] Work supported in part by NSF MCS78-04878 and US ARO DAAG29-83-K-0026.

[2] Work supported in part by Department of Energy contract DE-AC01-ER81-01997.

Phase 3: Determine $U(x,y)$ so that:

(i) for each element E of Ω

$$[LU - g]\,|_{P_i} = 0, \quad i = 1 \text{ to } 4$$

where the P_i are four interior points of E.

(ii) for each element E of $\partial\Omega$

$$[\Lambda U - \delta]\,|_{Q_j} = 0, \quad j = 1 \text{ to } n(E)$$

where the Q_j are $n(E)$ points of the piece of ∂R associated with E.

Phase 4: Solve the resulting linear system from Phase 3(i) and (ii) to obtain the coefficients of the approximation $U(x,y)$.

See [Houstis et al, 1978], [Houstis et al, 1979] and [Rice, 1983a] for further discussion.

We now present a more detailed, but still very high level descriptions of an implementation of the collocation method. The procedure is broken into seven steps given below and these are then described individually.

/* skeleton of the collocation procedure */

1. Define the problem and I/O
2. Discretize the domain
3. Generate the finite element mesh
4. Define the approximate solution
5. Form the collocation equations
6. Reorder the collocation equations
7. Solve the collocation equations

2.1 Problem Definition and I/O Specification

Following the ELLPACK framework [Rice and Boisvert, 1983], an elliptic PDE problem is specified by a set of functions as follows:

/* 1. Problem definition and I/O specification /*

a. Operator
function PDE(X,Y,CVALUS) where

CVALUS[1:7] = (a,b,c,d,e,f,g) at (x,y) in Ω and PDE = $g(x,y)$

b. Domain of Problem (in parametric form)

$x(p,i)$, $y(p,i)$, i=1 to nbound, $b(1,i) \le p \le b(2,i)$

where

nbound = number of boundary pieces

$b(j,i)$ = ends of parameter range

c. Boundary Conditions

function BCOND(I,X,Y,BVALUS) where
BVALUS[1:4] = (α, β, γ, δ)
at (x,y) on the boundary piece i and BCOND=$\delta(x,y)$.

d. Uniqueness Conditions
In case of Neumann boundary conditions, a unique solution is determined by knowing the solution at boundary node point $(unqx,unqy)$. The information is supplied by the function $unqu(unqx,unqy)$.

e. Ouput Specifications
The output from the method is specified by two arrays OUTFNC and OUTTYP as follows. The user specifies one of three functions for the Ith output through OUTFNC(I) as follows:

OUTFNC(I) = 1 : approximate solution U
= 2 : error = U-TRUE
= 3 : residual = $Lu-g$

where TRUE is the exact solution of (1.1), (1.3) which must be supplied by the user. Other information for the Ith output is specified by OUTTYP(I) as follows:

OUTTYP(I) =1 : max, $L1$,$L2$ norms based on the discretization grid $\Omega \cap G$
=2 : max, $L1$,$L2$ norms based on a user specified grid
=3 : table of function values on the discretization grid
=4 : table of function on a user specified grid

A grid for output of type 2 and 4 is specified by the following parameters:

tabx, taby - tables of (x,y) coordinates of the grid
ntabx, ntaby - number of grid lines in $tabx$ and $taby$.

Finally, the number of output specifications desired is specified by the parameter NOUT.

For example, if OUTFNC(3)=2 and OUTTYP(3)=1, then one requests the max, $L1$ and $L2$ norms of the error $LU-g$ on the discretization grid.

2.2 Domain Discretization

Information must be generated which relates the problem domain R to the rectangular grid G. This geometric information must be fairly detailed, otherwise large amounts of code will appear in other parts of the implementation just to do a basic analysis of the geometry. We use the two dimensional domain processor of [Rice, 1982a], [Rice, 1982b] and, for completeness, briefly describe its input and output here.

The rectangular grid G is defined by the following variables:

AX,BX : left and right endpoints of x-interval
AY,BY : bottom and top endpoints of y-interval

NGRIDX : number of x-grid lines
NGRIDY : number of y-grid lines
GRIDX : vector of length NGRIDX containing values of x-grid lines
GRIDY : vector of length NGRIDY containing values of y-grid lines
EPSGRD : accuracy with which geometric data is to be determined

With this information and the problem definition the domain processor generates the following information: A two dimensional array which associates the grid points with the domain R and its boundary.

/* 2.1 Grid specification */

GTYPE(i,j), i=1 to NGRIDX, j=1 to NGRIDY

The values in GTYPE indicate whether a grid point is interior or exterior and locate it relative to ∂R if it is a neighbor of ∂R.

The domain processor also generates a record of seven 1-dimensional arrays of length NBNDPT+1 where NBNDPT = the number of boundary points. A boundary point is where ∂R intersects the grid G and these are ordered along ∂R. If ∂R changes pieces off the grid G, then these points are also included as boundary points. If B_i denotes the ith boundary point, the fields of the record are defined as follows:

/* 2.2 Boundary coefficients */

XBOUND(i), YBOUND(i) : x and y coordinates of B_i
BPARAM(i) : parameter value of B_i
PIECE(i) : index of boundary piece to which B_i belongs (smallest index if there are two)
BPTYPE(i) : type of boundary point (horizontal, vertical, both and interior)
BNEIGH(i) : pointer to interior grid point neighbors of B_i
BGRID(i) : $ix+1000*jy$ if B_i is in the grid element with lower left corner (ix,jy) in G

The domain processor sets the NBNDPT+1 value of the arrays to the $i=1$ values and it also requires NBDIM = actual dimension of above arrays.

Our implementation of the collocation method is based on this information. In the absence of the domain processor, this information must be provided directly as input. In the special case of rectangular domains, the domain discretization is implicitly defined by the vectors GRIDX and GRIDY and no domain processing is required.

2.3 Finite Element Mesh Generation

In the case of rectangular regions the finite element mesh Ω coincides with the rectangular overlay G and no further processing is needed. For non-rectangular regions each element is identified by the indices (ix,jy) of the lower left corner grid point where

$$1 \le ix < \text{NGRIDX} \quad \text{and} \quad 1 \le jy < \text{NGRIDY} .$$

There may be boundary elements whose intersection with R is very small. In extreme cases, the use of these elements can make later computations numerically unstable. In any case, it is intuitively plausible that very small elements should be discarded just for the sake of efficiency. We thus define the **finite element mesh** Ω to consist of the interior elements, plus those boundary elements E for which the quotient **area of** $E \cap R$ **over area** E is greater than the value of the parameter DSCARE. The portions of ∂R from discarded elements are allocated to neighboring elements or ignored depending on the value of the logical variable GIVOPT.

The finite element mesh Ω forms a rectangular approximation to R. Its exterior sides are called **boundary sides** and they play a key role in the method. See Figure 1 for an example. The construction of the finite element mesh thus consists of the following three steps:

```
            /* 3. Generate finite element mesh */
    3.1  determine the element types
    3.2  determine the finite element mesh
    3.3  associate boundary segments with elements
```

For steps 3.2 and 3.3 the elements of G are classified as **interior**, **boundary** or **exterior** depending on whether they are completely inside R, intersect ∂R, or are completely outside R. Some elements may be changed from boundary to exterior or from interior to boundary by the discard procedure. Our implementation of the above procedure depends very much on the following assumptions:

Assumptions for the finite element mesh generation

a1: A boundary element does not contain an entire boundary piece and there are at most two(2) boundary pieces associated with it.

a2: If a boundary element has exactly two boundary sides, then they must be adjacent; a boundary element cannot have all its sides be boundary sides.

a3: If a boundary element is discarded, then no more than two of the four(4) neighboring elements can be without any boundary segment associated with them.

a4: The domain is parameterized clockwise.

a5: The boundary does not enter an element more than once, except when it leaves the element and reenters it without crossing a grid line and where the neighboring element it enters is discarded.

These assumptions are usually satisfied for a reasonably fine mesh.

We present a skeleton code for the finite element mesh generation.

```
                /* 3.1 DETERMINE THE ELEMENT TYPES */
LOOP:  FOR EACH BOUNDARY POINT $B_I$ DO:
    IF      THE BOUNDARY LEAVES AN ELEMENT AND ENTERS A NEW
            ELEMENT $(IX, JY)$ AT THIS POINT
    THEN    SAVE THE BOUNDARY POINT INDICES FOR THAT NEW ELEMENT
            AS ELTYPE$(IX, JY)$:=IENTER+1000IEXIT WHERE IENTER AND
            IEXIT ARE THE INDICES OF THE BOUNDARY POINTS WHERE THE
            BOUNDARY ENTERS AND EXITS ELEMENT $(IX, JY)$
ENDLOOP;
```

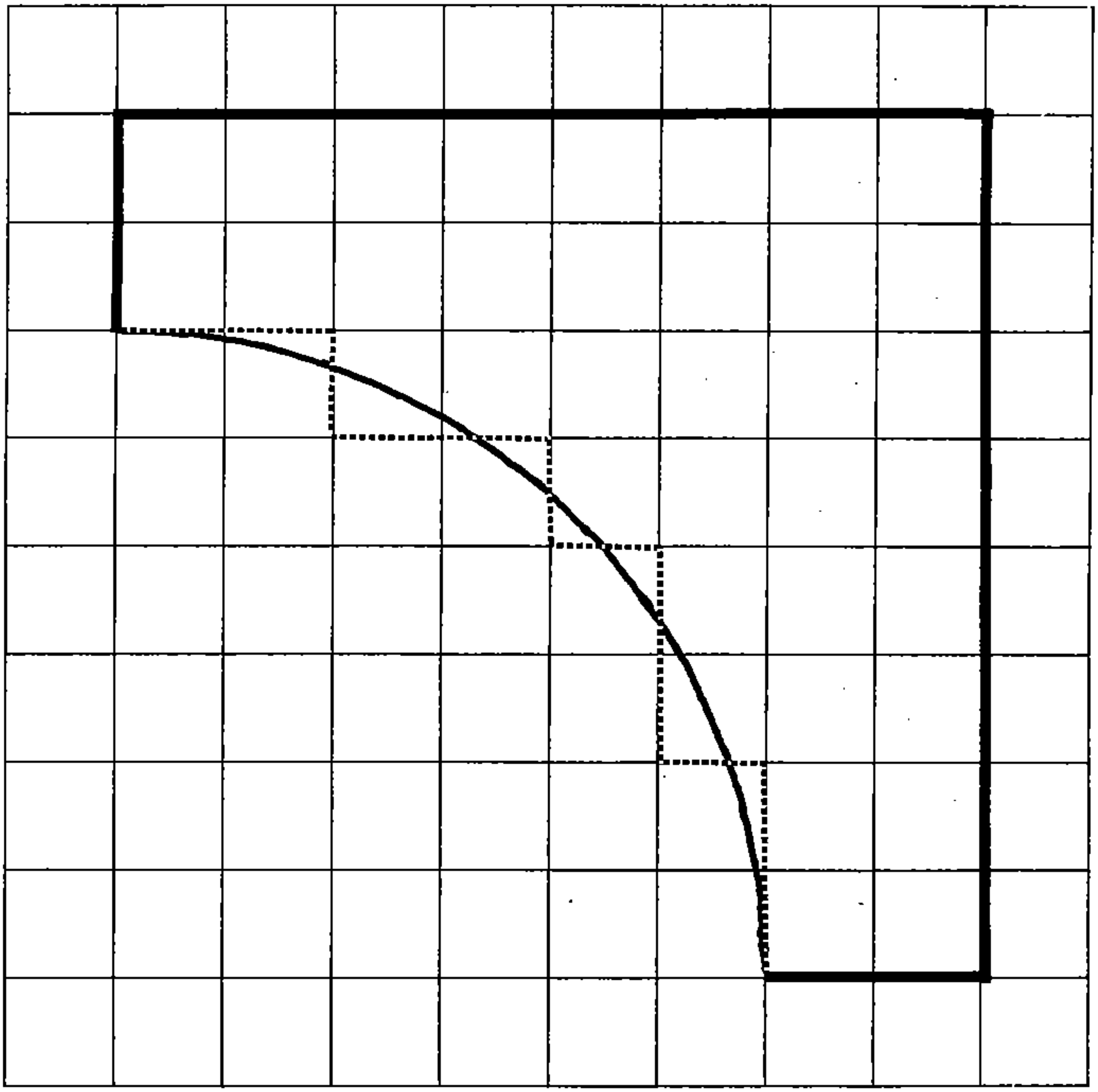


Figure 1. A domain R (heavy lines) with finite element mest Ω embedded in the rectangular grid G (light lines). The boundary sides of Ω are shown dotted where they do not coincide with $\partial\Omega$.

```
                /* 3.2 DETERMINE THE FINITE ELEMENT MESH */

LOOP:  FOR EACH ELEMENT $(IX, JY)$ OF $G$ DO:

CASE ELEMENT TYPE OF

      EXTERIOR: ELTYPE(IX,IY):=-1 /* DO NOT USE ELEMENT */
      INTERIOR: ELTYPE(IX,IY):= 0 /* USE ELEMENT */

      BOUNDARY:   IF $\frac{\text{AREA OF ELEMENT INTERSECTION}}{\text{AREA OF ELEMENT}}$ < DSCARE

                  THEN ELTYPE$(IX, JY)$:=-ELTYPE(IX,JY)   */ DO NOT USE ELEMENT */
                  /* ELSE ELTYPG = (IENTER+1000IEXIT) AND THE ELEMENT IS USED */
ENDCASE
ENDLOOP

              /* 3.3 ASSOCIATE BOUNDARY SEGMENTS WITH ELEMENTS */

A. LOOP  FOR EACH BOUNDARY SEGMENTS DO:

     /* IF SEGMENT IS IN ELEMENT $(IX, JY)$ AND ELTYPE$(IX, JY)$<-1
        THEN THE BOUNDARY SEGMENT IN THE DISCARDED ELEMENT IS ASSIGNED
           TO A NEIGHBORING ELEMENT */
          IF ANY NEIGHBORING ELEMENTS HAVE NO ASSOCIATED BOUNDARY SEGMENT
             THEN THE BOUNDARY SEGMENT IS SPLIT AMONG THEM (UP TO 2 PIECES)
             ELSE IF  GIVOPT = .TRUE.
                THEN THE BOUNDARY SEGMENT IS SPLIT BETWEEN THE TWO ELEMENTS
                     WHOSE ASSOCIATED BOUNDARY SEGMENTS ARE CONNECTED TO IT
   ENDLOOP

/*NOTE: IF GIVOPT IS .FALSE. THEN THE PIECE OF THE BOUNDARY IN THE
       DISCARDED ELEMENT IS NOT USED */

/* 3.3 NUMBER THE NODES AND ELEMENTS OF THE FINITE ELEMENT MESH */

B. NODES AND ELEMENTS ARE NUMBERED VERTICALLY FROM BOTTOM TO TOP,
   THEN HORIZONTALLY FROM LEFT TO RIGHT
   /* SEE FIGURE 2 */
```

Figure 3 shows a complex domain with the discarded elements shaded and the association of the boundary segments made. An x indicates the end of each boundary segment and GIVOPT is .TRUE., that is, boundary segments from discarded segments are shared with neighboring elements. The default values of DSCARE and GIVOPT are .05 and .TRUE. respectively.

2.4 Definition of the Approximate Solution

Consider an interior element of the finite element mesh Ω, it has the associated nodes numbered i, $i+1$, j, $j+1$ as shown in Figure 4. The approximate solution $U(x,y)$ is a Hermite bicubic piecewise polynomial; it is defined on each element in terms of the 1-dimensional local basic functions shown in Figure 4. If we use a local numbering of the 16 **degrees of freedom** (dof) of $U(x,y)$ on this element, then we have

$$\begin{aligned} U(x,y) = {} & q_1 v_1(x) w_1(y) + q_2 v_1(x) w_3(y) + q_3 v_3(x) w_1(y) + q_4 v_3(x) w_3(y) \\ & q_5 v_2(x) w_1(y) + q_6 v_2(x) w_3(y) + q_7 v_4(x) w_1(y) + q_8 v_4(x) w_3(y) \\ & q_9 v_2(x) w_2(y) + q_{10} v_2(x) w_4(y) + q_{11} v_4(x) w_2(y) + q_{12} v_4(x) w_4(y) \\ & q_{13} v_1(x) w_2(y) + q_{14} v_1(x) w_4(y) + q_{15} v_3(x) w_2(y) + q_{16} v_3(x) w_4(y) \end{aligned}$$

It is worth noticing that there is a natural relation of the dofs (the q

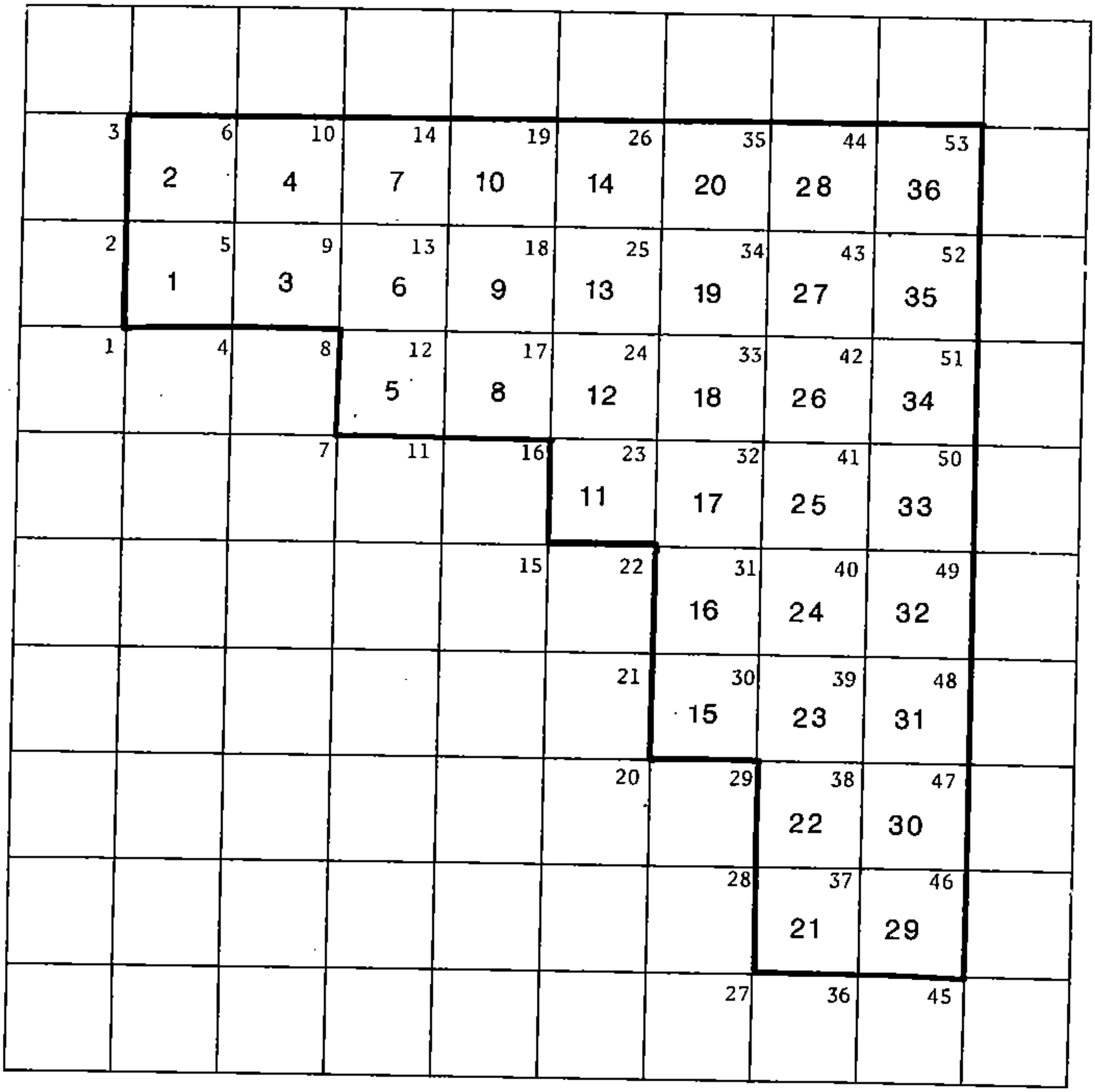


Figure 2. Numbering of finite element mesh and nodes for the domain of Figure 1.

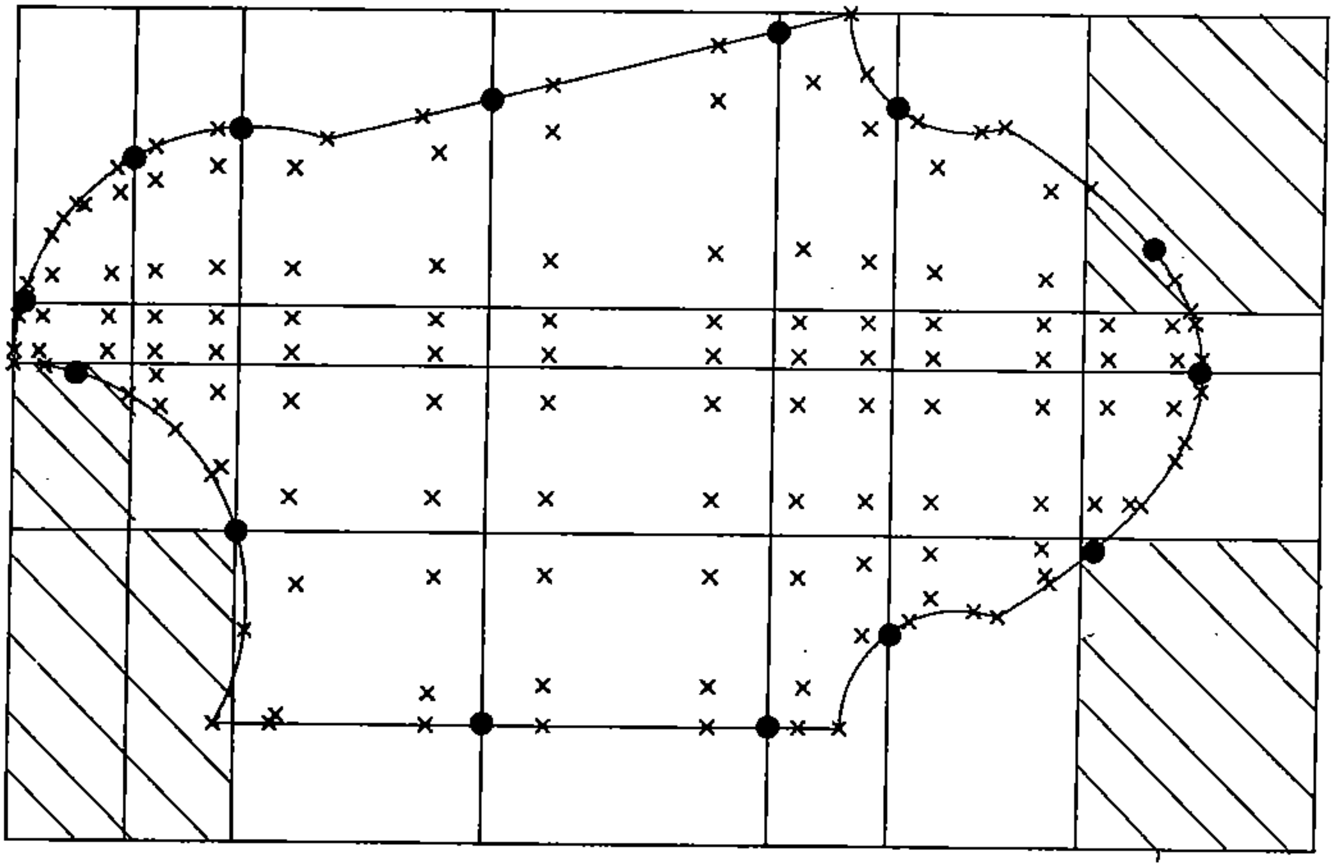


Figure 3. A domain showing the discarded elements shaded and the ends (dot's) of the boundary segments as associated with the retained elements for GIVOPT=.TRUE. The x's are the collocation points for the operator and boundary conditions.

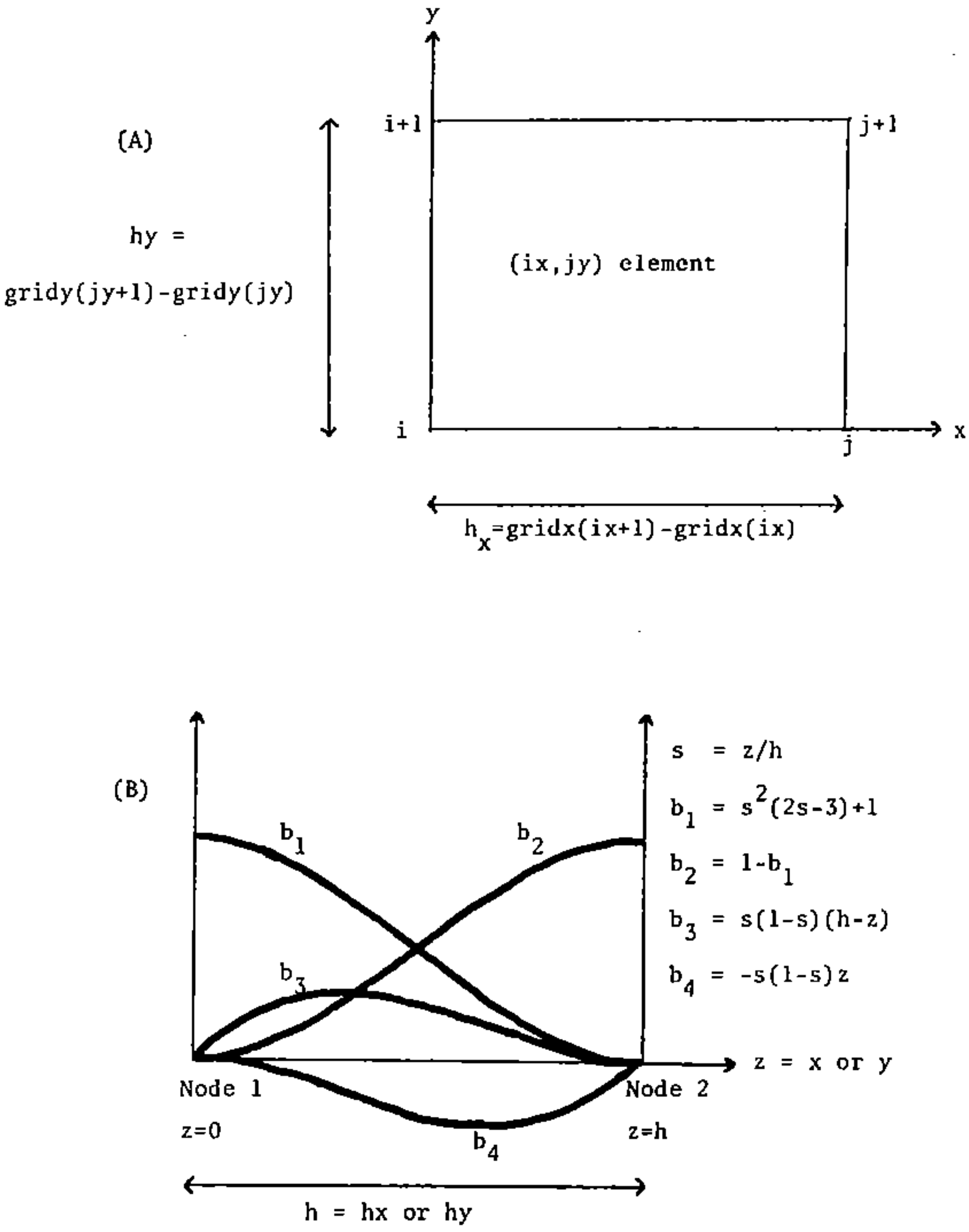


Figure 4. (A) An interior element of the mesh Ω with the nodes numbered. (B) The four non-zero 1-dimensional local basis functions for each variable in the element. These functions are half of the two standard Hermite cubic basis functions for each of the two nodes.

coefficients) of U with the values at each node of

$$U, \ \frac{\partial U}{\partial y}, \ \frac{\partial U}{\partial x}, \ \frac{\partial^2 U}{\partial x \partial y}$$

We have, for example, $q_1 = U(\text{node}(i))$, $q_2 = U_x(\text{node}(i))$. The global numbering used for these 16 values at the m th node is

$$\{4\,m\text{-}3,\ 4\,m\text{-}2,\ 4\,m\text{-}1,\ 4\,m\}$$

2.5 Formation of the Collocation Equations

The generation of the collocation equations is outlined by the following code skeleton.

```
        /* 5. CODE SKELETON FOR COLLOCATION EQUATIONS */

LOOP  OVER ELEMENTS OF $\Omega$;
      INVOKE BOUNDARY COLLOCATION POINTS PROCEDURE
      IF ELEMENT = BOUNDARY
          THEN INVOKE BOUNDARY ELEMENT PROCEDURE
               INVOKE BOUNDARY CONDITION PROCEDURE
          ELSE
               INVOKE INTERIOR ELEMENT PROCEDURE
ENDLOOP
```

To generate the operator collocation equations for the interior elements in Ω', one first determines the interior collocation points P_l, 1=1 to 4 as the Gauss points of the rectangle and then forces U to satisfy the differential equation at these points. The collocation equations are represented by a data structure with two-dimensional arrays:

coef(n,l) = lth coefficient value of equation n
idcoef(n,l) = index of the unknown associated with coef(n,l)

These arrays have 17 columns, 16 for the coefficients and the 17th for the value of the right side g at the collocation point.

procedure INTERIOR ELEMENT

```
    DETERMINE THE $P_I$ AS THE FOUR GAUSS POINTS OF THE ELEMENT;
    LOOP WITH $P_I = (X_I, Y_I)$ FOR $I$=1 TO 4 DO:
          /* NROW IS EQUATION INDEX */
          NROW:= NROW+1
          LOOP FOR N, M = 1 TO 4 DO:
                K:=N + 4(M-1)
                IDCOEF(NROW, K):=GLOBAL NUMBERING OF DOF K
                COEF(NROW, K):=$L(V_N(X_I)W_M(Y_I))$
          ENDLOOP
          COEF(NROW,17):=PDE RIGHT SIDE AT $P_I = G(X_I, Y_I)$
    ENDLOOP
```

To generate the operator collocation equations for a boundary element E requires that E be mapped into $E \cap R$. The image of the four Gauss points under this mapping are the collocation points used.

procedure BOUNDARY ELEMENT

```
/* INTERIOR COLLOCATION POINTS FOR 'BOUNDARY' TYPE ELEMENT */

    DEFINE A MAP FROM THE BOUNDARY $\partial E$ OF THE ELEMENT $E$ TO THE
```

```
BOUNDARY OF THE INTERSECTION OF THE ELEMENT AND DOMAIN, $\partial(E \cap R)$,
BY PARTITIONING $\partial(E \cap R)$ INTO FOUR PARTS AND MAPPING EACH
SIDE OF THE ELEMENT TO ONE OF THOSE PARTS (SEE FIGURE 5).

DEFINE A MAP FROM $E$ TO $E \cap R$ BY LINEARLY BLENDING THE FOUR
MAPS OF THE BOUNDARY.

DETERMINE THE $P_I$'S AS THE IMAGES UNDER THIS MAP OF THE FOUR
GAUSS POINTS OF THE ELEMENT.

FILL THE ARRAYS COEF AND IDCOEF AS IN THE PROCEDURE INTERIOR ELEMENT.
```

The map in the procedure boundary element from ∂E to $\partial(E \cap R)$ depends on several aspects of the geometry and is too complicated to give in complete detail here. However, most of the maps are variants of the four cases shown in Figure 5. See [Gordon and Hall, 1973] for a discussion of linear blending in two dimensions.

It appears that if $E \cap R$ is convex, then the map from E to $E \cap R$ is one-to-one and onto. If $E \cap R$ is not convex, then the map might not be one-to-one and, if there is a strong concavity, might even map points from E to points outside $E \cap R$. However, a proper choice of grid will keep the images of the Gauss points inside $E \cap R$. An example for a portion of an actual domain is shown in Figure 6.

To generate the **boundary condition collocation equations**, one has to determine the location of the boundary collocation points Q_j, $j=1,\ldots,n(E)$ associated with each boundary element E of the finite element mesh Ω. It can be shown that the method described gives $2s+4$ boundary collocation points, where s is the number of boundary element sides of Ω. The process of the distribution of boundary points on the actual boundary is implemented in two passes.

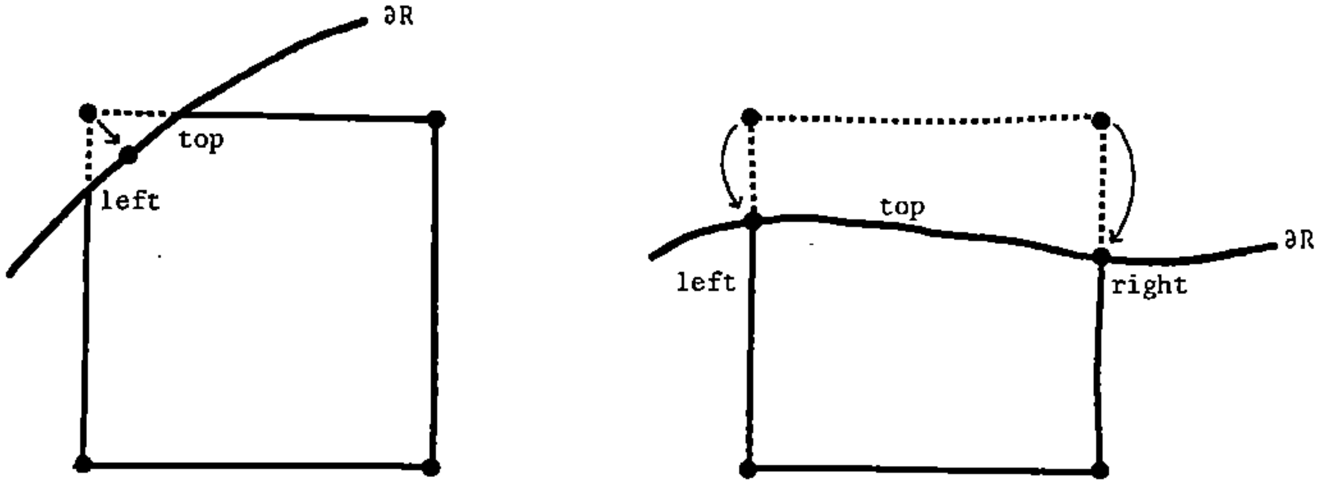
The first pass is to place collocation points on the boundary sides of Ω (not on the boundary itself). Four collocation points are associated with each grid node, one in each element adjacent to the node. Those points that are interior to Ω (possibly exterior to the domain R) are associated with interior collocation points and not considered further. Those points which are exterior to Ω are projected onto the boundary sides of the mesh and become the (intermediate) boundary side collocation points. See Figure 7 for an example.

The second pass is to map the boundary sides of a rectangular element onto the pieces of the boundary associated with the element; the images of the boundary side collocation points are the boundary collocation points used in the discretization procedure.

There are two parameters, BCP1 and BCP2, to adjust the placement of the boundary collocation points in a boundary side. These allow one to vary the placement from the two Gauss points to nodes and midpoints, etc. The default case (BCP1 = BCP2 = 0) selects the Gauss points on an element boundary side. A skeleton code for the placement of the boundary collocation points (BCPs) in the element E follows.

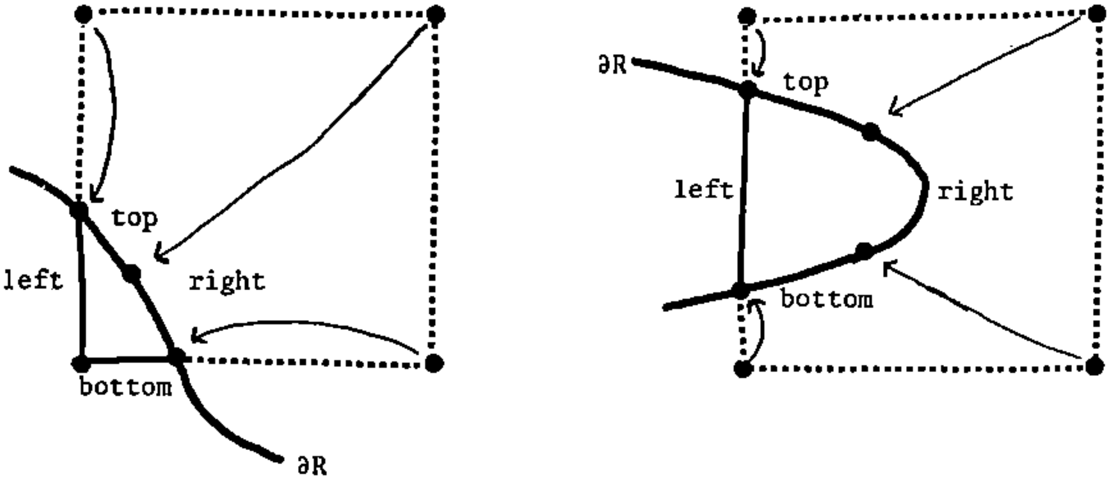
procedure BOUNDARY COLLOCATION POINTS

```
PASS 1:  /* ASSOCIATE BOUNDARY COLLOCATION POINTS WITH BOUNDARY
            OF FINITE ELEMENT MESH */

   PLACE TWO BCPS ON EACH BOUNDARY SIDE OF $E$ IN THE SAME CONFIGURATION AS
     PARAMETERS BCP1 AND BCP2 ARE PLACED IN THE INTERVAL (0,1).
   PLACE ONE BCP AT EACH CORNER OF $\partial\Omega \cap E$.
   IF THE END OF THE LAST BOUNDARY SIDE IS A CONCAVE CORNER
     OF THE FINITE ELEMENT MESH
       THEN REPLACE THE TWO BCP OF THE LAST BOUNDARY SIDE WITH
```



(A) One exterior node

(B) Two exterior nodes



(C) Three exterior nodes

(D) Four exterior nodes

Figure 5. The four basic mappings used for boundary elements. The dashed lines are the boundary sides external to R. The four element sides are mapped to the actual boundary of $E \cap R$ as indicated by the images of the element nodes.

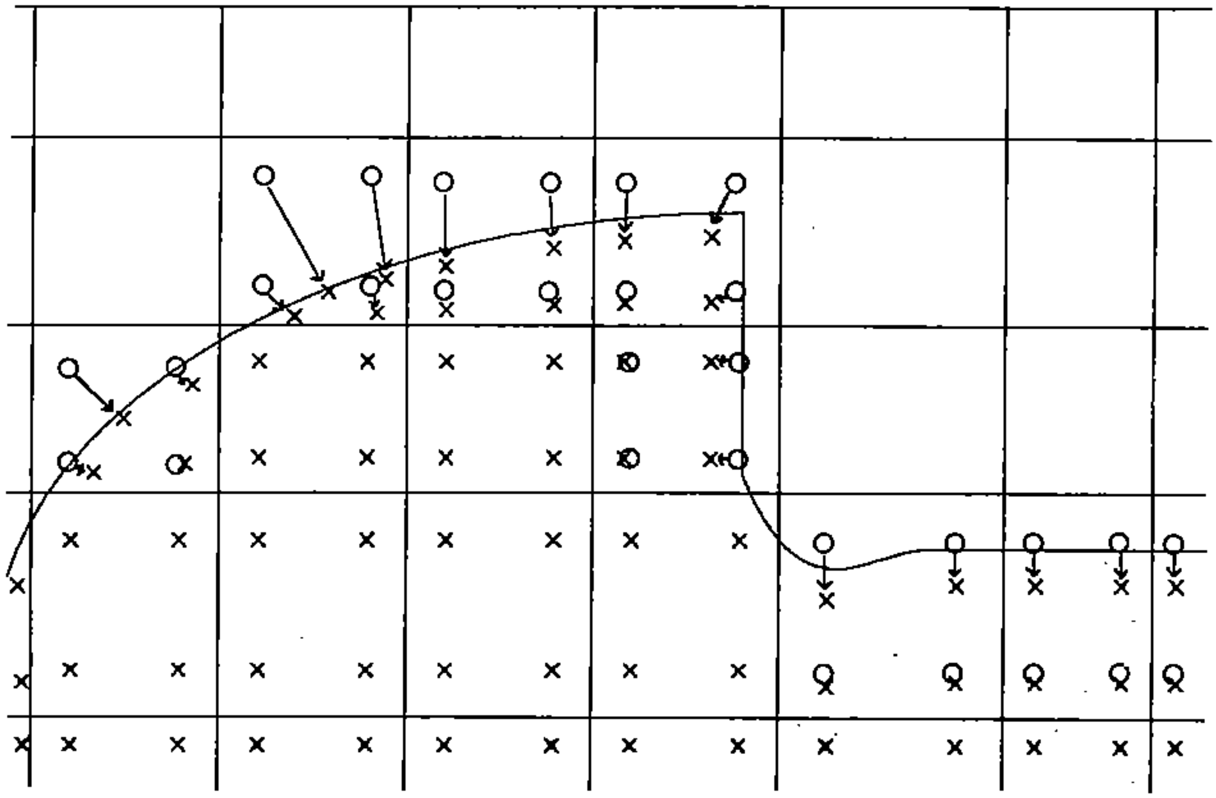
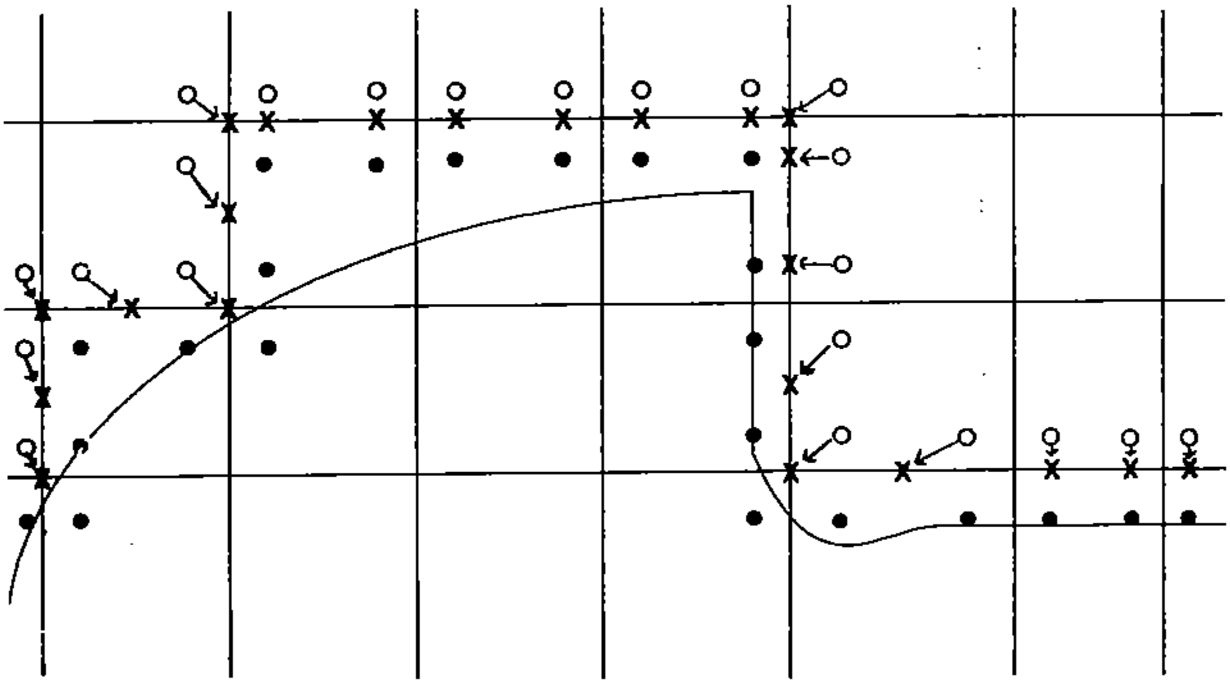
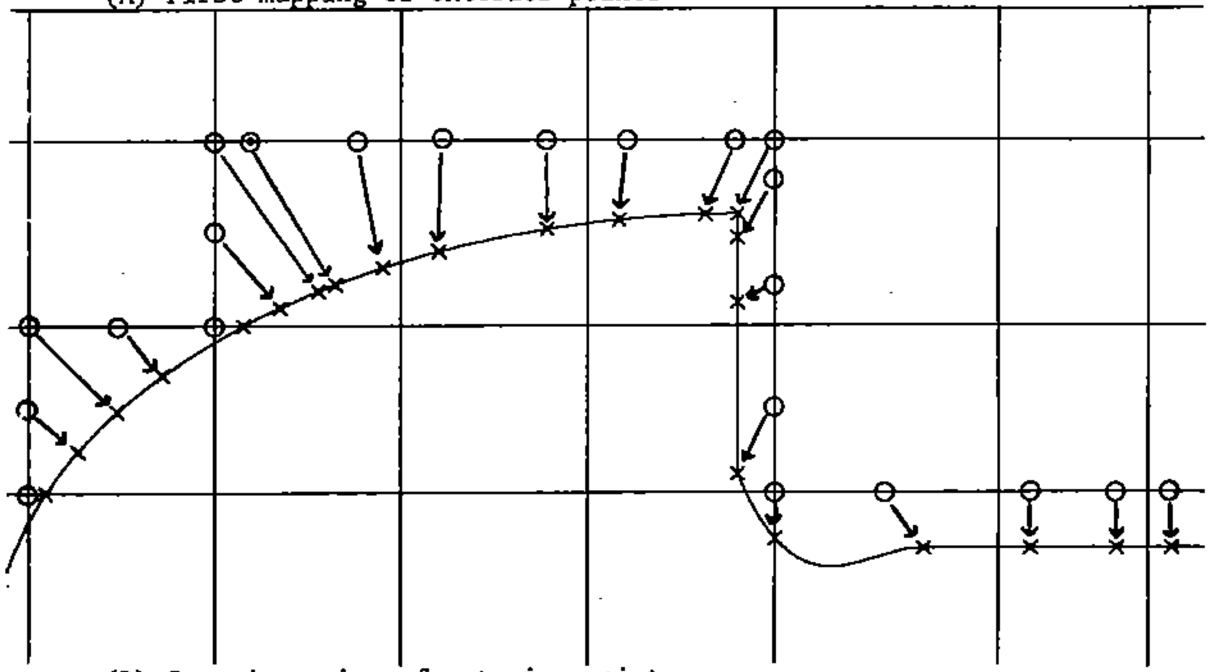


Figure 6. The mappings of interior collocation points along the boundary of a domain. The o's are the Gauss points of boundary elements and the x's are their images under the mapping.



(A) First mapping of exterior points



(B) Second mapping of exterior points

Figure 7. Mapping of points representing degrees of freedom into boundary collocation points. (A) Four points are associated with each node on the boundary of Ω. Those exterior (o's) to Ω are mapped into the boundary (x's) and those interior (solid dots) are mapped to interior collocation points as in Figure 6. (B) The points (o's) on the boundary are then mapped onto ∂R (x's) to be the boundary collocation points.

```
          ONE BCP AT THE MIDPOINT OF THE SIDE
    IF THE BEGINNING OF THE FIRST BOUNDARY SIDE IS A
      CONCAVE CORNER OF THE FINITE ELEMENT MESH
        THEN MOVE THE TWO BCP OF THE FIRST SIDE SO THAT
           THE FIRST BCP IS AT THE BEGINNING OF THE FIRST SIDE AND
           THE SECOND BCP IS AT THE MIDPOINT OF THE FIRST SIDE

/* THIS PLACEMENT IS REPRESENTED BY VALUES IN (0,1) WITH 1/2 CORRESPONDING
  TO THE CORNER IF THERE ARE TWO BOUNDARY SIDES AND 1/3 AND 2/3
  CORRESPONDING TO THE CORNERS IF THERE ARE THREE BOUNDARY SIDES */


PASS 2: /* MAPPING THE BCP FROM $\partial\Omega$ TO $\partial R$ */

/* THIS IS A MAPPING FROM (0,1) TO THE SEGMENT OF $\partial R$
   ASSOCIATED WITH THE ELEMENT $E$ */

IF THE SEGMENT OF $\partial R$ IS CONTAINED IN ONE PIECE OF THE BOUNDARY
   THEN LINEARLY MAP (0,1) TO (PENTER=BPARAM(IENTER), PEXIT=BPARAM(IEXIT))
   DETERMINE THE BCPS FROM THE PASS 1 VALUES AND THE DEFINITION OF $\partial R$.

ELSE IF THE SEGMENT OF $\partial R$ IS CONTAINED IN TWO PIECES OF THE BOUNDARY
THEN LINEARLY MAP (0,1/2) TO (PENTER, $B_{2,I}$) AND (1/2,1) TO ($B_{1,I+1}$, PEXIT), WHERE
     $I$ IS THE NUMBER OF THE FIRST PIECE AND $B_{2,I}$, $B_{1,I+1}$ ARE FROM (1.2).
     DETERMINE THE BCPS FROM THE PASS1 VALUES AND THE DEFINITION OF $\partial R$
ELSE ERROR /* DO NOT ALLOW MORE THAN TWO PIECES
              OF BOUNDARY IN ONE ELEMENT */
```

Once the collocation points for the boundary conditions are determined by the above procedure, the rest of the generations of the boundary condition collocation equations is simple.

procedure BOUNDARY CONDITION

```
/* BOUNDARY ELEMENT $E$ HAS $K$ BOUNDARY SIDES */

LOOP OVER BOUNDARY SIDES OF $E$ DO:
   LOOP OVER BOUNDARY COLLOCATION POINTS $Q_I=(X_I, Y_I)$ FOR BOUNDARY SIDES DO:
      /* NROW IS EQUATION INDEX */
      NROW:=NROW+1
      LOOP FOR N,M = 1 TO 4 DO:
           K:=N+4(M-1)
           COEF(NROW,K) = $\Lambda(V_N(X_I) W_M(Y_I))$
           IDCOEF(NROW,K) = GLOBAL NUMBERING OF K-TH DOF
      ENDLOOP
      COEF(NROW,17) = $\delta(X_I, Y_I)$
   ENDLOOP
ENDLOOP
```

2.6 Reordering of the Collocation Equations

The numbering of the equations and unknowns used in the previous section results in a system of linear equations that is banded in nature. If R is rectangular or close to rectangular, then the system has bandwidth about 4*NGRIDY. As the domain R deviates from being rectangular more and more, the structure of the linear system becomes less and less regular and very little can be said for a completely general region.

The ordering generated in the actual algorithms discussed here is the natural extension of the **finite element ordering** to general domains. That is, if R were rectangular, this ordering would be obtained. There is a second ordering natural to collocation called the **collorder ordering**. This ordering is defined for rectangular domains in [Dyksen and Rice, 1983]; it can be extended to general domains in a straightforward way. The finite element ordering is attractive because it gives minimum band width in the rectangular case. The **collorder ordering** is attractive because it gives a non-zero diagonal and provides maximum numerical stability in the rectangular case. An example of these two ordering for a triangular domain is given in [Rice, 1983a] and reproduced in Figure 8.

2.7 Solution of the Collocation Equations

It is customary for the linear equations arising from finite element methods (such as this collocation method) to be solved by some form of Gauss elimination. If R is not far from rectangular, then the system can be made banded by a variety of orderings and considerable efficiency achieved compared to Gauss elimination for a general system of equations. The widely used frontal method often provides the efficiency of bandedness even when R is far from rectangular, even though it is not guaranteed to do so.

A recent study by [Rice, 1983b] indicates that iterative methods are much more efficient than elimination methods for the Galerkin method equations (on a rectangle) and it is plausible that this is also true for the collocation equations. The usual finite element ordering prevents iterative methods from being applied because there are mostly zeros on the diagonal. The collorder ordering remedies this, but the usual iterative methods diverge rapidly when directly applied to the collocation equation. A convergent iteration method (for the model problem of Laplace's equation on a rectangle) has been presented by [Balart et al, 1982]. It is still open as how to define fast converging methods for the collocation equations in general, but this question should be viewed as one with good prospects for favorable results.

At this time the only reliable way to solve the collocation equations in general is by Gauss elimination with scaled partial pivoting.

3. THE SPECIAL CASES OF RECTANGULAR DOMAINS

The method described in Section 2 can be considerably simplified in case:

(i) the domain R is rectangular

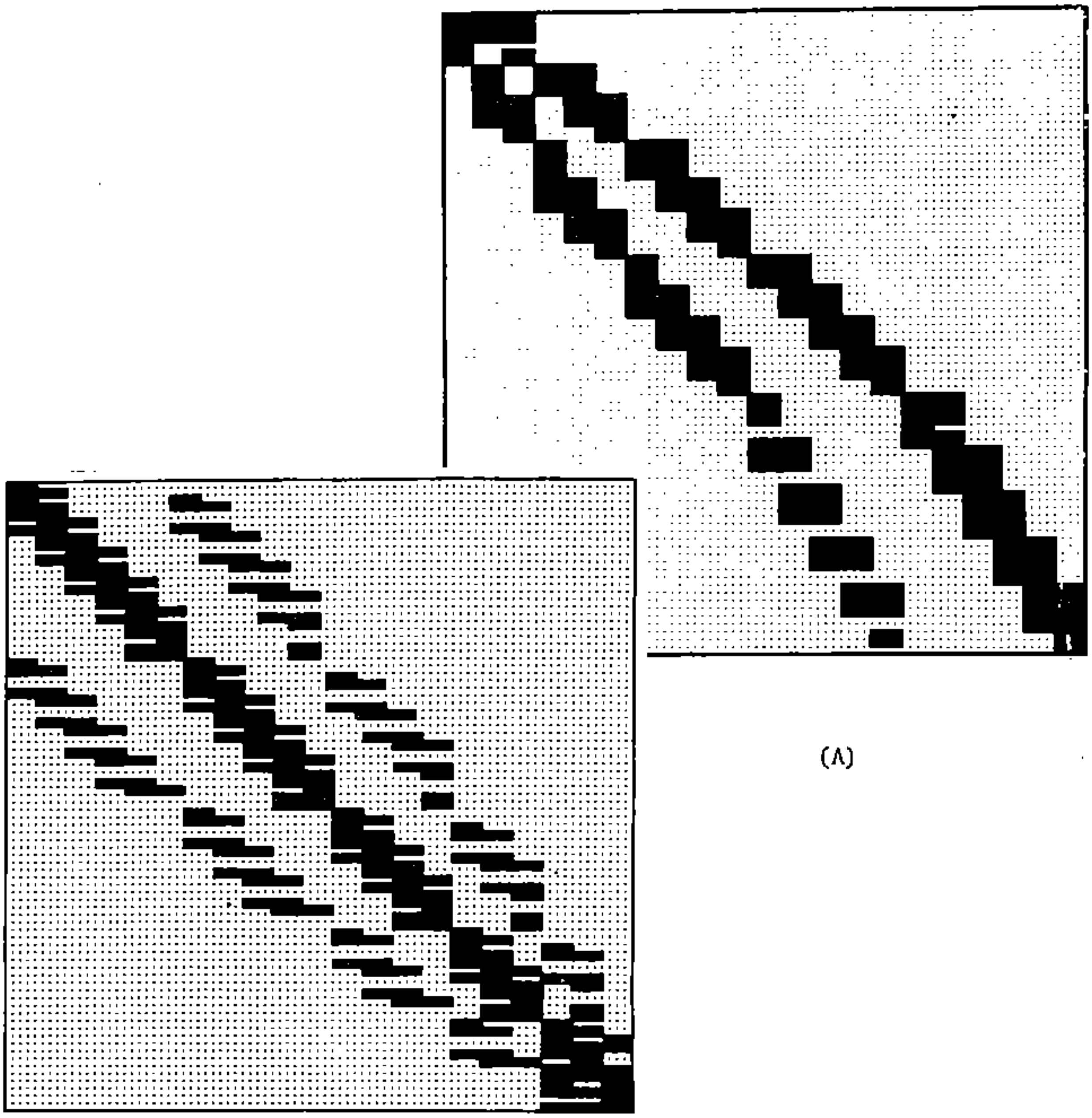
and further simplified if

(ii) the problem has **uncoupled boundary conditions** that is,

$u = \delta$ in part of the boundary ∂R_1,

$\frac{\partial u}{\partial n} = \delta$ in the rest of the boundary $\partial R_2 \equiv \partial R - \partial R_1$

The collocation method for rectangular domains with general mixed boundary conditions is called throughout **hermite collocation** while for rectangular regions with uncoupled boundary conditions it is called **interior collocation**. For rectangular domains, some of the collocation steps are implicitly defined by the input data. First, the domain discretization process is implicitly defined by the vectors GRIDX, GRIDY. Second, the finite element mesh generator process is not needed, since the nodes of Ω coincide with the grid points of rectangular overlay G. The same local definition of the approximate solution is used, but the steps of generating the collocation equations are considerably simplified. We describe



(A)

(B)

Figure 8. The patterns of non-zeros in the collocation equations for a triangular domain with (A) the finite element ordering of the equations and unknowns and (B) the collorder ordering.

these steps for both interior and Hermite collocation.

3.1 Interior Collocation

This is the case of uncoupled boundary conditions where the boundary collocation equations can be solved explicitly during the discretization of the boundary conditions. Thus, one needs only generate and solve the interior collocation equations. The implementation of this step can be done by two parallel asynchronous processes and it is based on the **assumption** that

"the boundary conditions only change type at the boundary nodes"

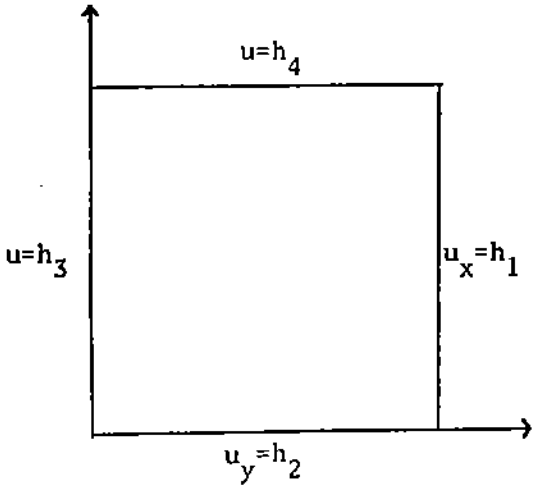
A code skeleton for the two processes follows:

```
        /* OPERATOR DISCRETIZATION */

LOOP OVER ALL ELEMENTS $E$ DO:
     INVOKE INTERIOR ELEMENT PROCEDURE
     /* THIS PROCEDURE IS GIVEN IN SECTION 2.5 */
ENDLOOP

        /* BOUNDARY DISCRETIZATION */

LOOP OVER EACH BOUNDARY PIECE:
   LOOP OVER EACH NODE $T_I$ OF BOUNDARY PIECE;

      DETERMINE THE LEFT OR RIGHT HALF INTERVAL ($[T_{I+\frac{1}{2}}, T_I]$ OR $[T_I, T_{I-\frac{1}{2}}]$)
      WHERE THE BOUNDARY CONDITION $BC$ IS OF THE SAME TYPE AS AT $T_I$.

      /* DENOTE THE INTERVAL BY $\Delta$ AND LET $\tau_1, \tau_2$ BE ITS TWO GAUSS POINTS */
      S = {$\tau_1, \tau_2$ AND END POINTS OF $\Delta$};
      CASE $BC$ TYPE IS OF:

         DIRICHLET(U= $\delta$): DETERMINE $U_X$ (OR $U_Y$) AT $T_I$ BY INTERPOLATING $\delta$ BY A
                         CUBIC AT THE POINTS S = {$\tau_1, \tau_2$ AND END POOINTS OF $\Delta$};
                         IDENTIFY ACTIVE DOFS;

         NEUMANN $\left(\frac{\partial}{\partial N} = \delta\right)$: DETERMINE $U_{XY}(=U_{YX})$ AT $T_I$ BY INTERPOLATING $\delta$ BY
                         A CUBIC AT THE POINTS S;
                         IDENTIFY ACTIVE DOFS;

      ENDCASE;
   ENDLOOP;
ENDLOOP;
```

Figure 9 shows the numbering of active dofs at the end of boundary discretization process. Finally, the nonactive dofs predetermined in the boundary discretization process are eliminated from equations generated in the operator discretization process.

3.2 Hermite Collocation

This is the case of mixed boundary conditions where the boundary condition collocation equations are explicitly generated along with the operator collocation equations. The user can set the location of the two boundary collocation points at each boundary element side by using the parameters $0 \leq BCP1, BCP2 \leq 1$. The default case ($BCP1 = BCP2 = 0$) selects the Gauss points in each boundary element side.



(A) Boundary Conditions

(B) Degree of freedom numbering associated with nodal value of (u, u_y, u_x, u_{xy}) for the boundary conditions of (A)

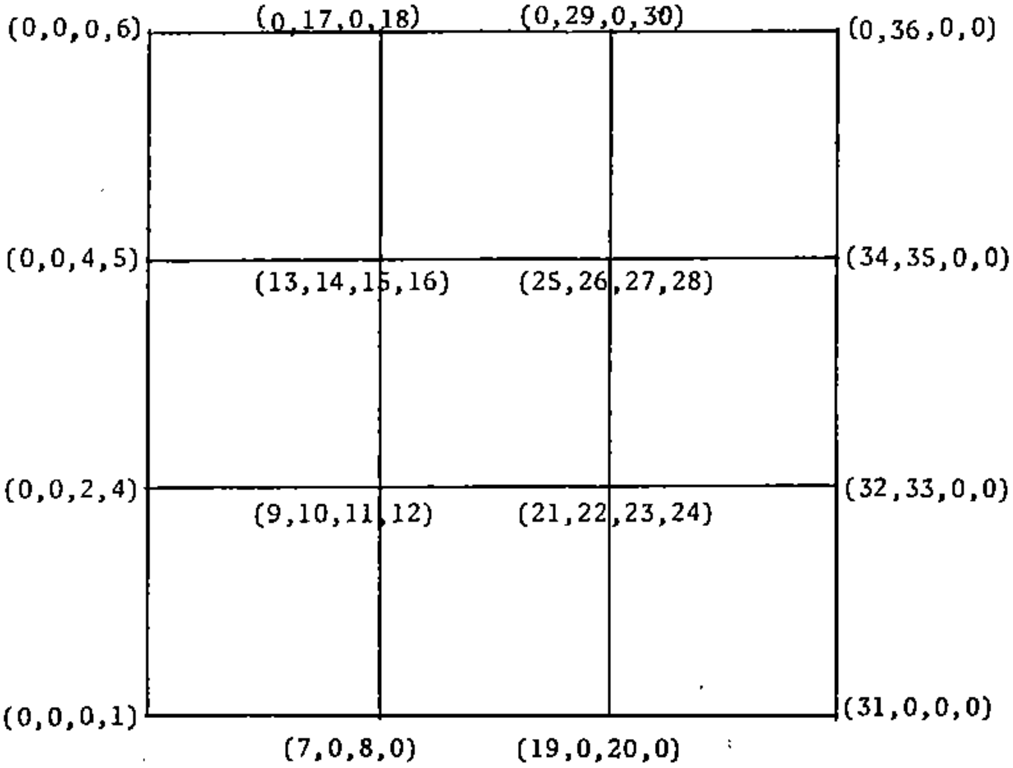


Figure 9. Example of the interior collocation process of identifying degrees of freedom. (A) Indicates boundary conditions of problem (B) Numbering of active degrees of freedom. The inactive or predetermined degrees of freedom are denoted by 0.

The Hermite collocation method is a direct simplification of the general method described previously. A code skeleton of the method follows:

```
LOOP OVER ELEMENTS $E$ OF $\Omega$ DO:
     INVOKE INTERIOR ELEMENT PROCEDURE
     IF ELEMENT = BOUNDARY
     THEN LOOP OVER BOUNDARY COLLOCATION POINTS $(X_I, Y_I)$ DO:
              NROW = NROW+1
              LOOP FOR N,M=1 TO 4 DO:
               K:=N+4(M-1)
                 COEF(NROW,K) = $\Lambda(V_N(X_I) W_M(Y_I))$
                 IDCOEF(NROW,K) = GLOBAL NUMBERING OF K-TH DOF
              ENDLOOP
              COEF(NROW,17) = $\delta(X_I, Y_I)$
          ENDLOOP
ENDLOOP
```

4. THE ALGORITHMS GENCOL, INTCOL AND HERMCOL

The collocation methods described above have been implemented as three Fortran programs [Houstis, Mitchell and Rice, 1983a], [Houstis, Mitchell and Rice, 1983b]. The initial comments of those programs provide a concise summary of each algorithm; we do not repeat that here. We identify the three algorithms and their input. Each applies to the operator (1.1) and boundary conditions (1.3).

GENCOL: General Collocation

```
INPUT: GENERAL DOMAIN SPECIFIED IN PARAMETRIC FORM CLOCKWISE
       RECTANGULAR OVERLAY: [AX,BX] × [AY,BY]
       7 PDE COEFFICIENT FUNCTIONS: CUXX(X,Y),...,CU(X,Y),G(X,Y)
       4 BOUNDARY CONDITION FUNCTIONS: $\alpha$(X,Y),$\beta$(X,Y),$\gamma$(X,Y),$\delta$(X,Y)
       2 OUTPUT SPECIFICATION ARRAYS: OUTFNC(I),OUTTYP(I), I=1 TO NOUT
```

INTCOL: Interior Collocation

```
INPUT:  RECTANGULAR DOMAIN
        RECTANGULAR GRID: POINTS X(I), I=1 TO NX+1; Y(J), J=1 TO NY+1
        7 PDE COEFFICIENT FUNCTIONS: CUXX(X,Y),...,CU(X,Y), G(X,Y)
        4 UNCOUPLED BOUNDARY COND. FUNCTIONS: $\alpha$(X,Y), $\beta$(X,Y), $\gamma$(X,Y), $\delta$(X,Y)
        2 OUTPUT SPECIFICATION ARRAYS: OUTFNC(I), OUTTYP(I), I=1 TO NOUT
```

HERMCOL: Hermite Collocation

```
INPUT:  RECTANGULAR DOMAIN
        RECTANGULAR GRID: POINTS X(I), I=1 TO NX+1; Y(J), J=1 TO NY+1
        7 PDE COEFFICIENT FUNCTIONS: CUXX(X,Y),..., CU(X,Y), G(X,Y)
        4 BOUNDARY CONDITION FUNCTIONS: $\alpha$(X,Y), $\beta$(X,Y), $\gamma$(X,Y), $\delta$(X,Y)
        2 BOUNDARY SPECIFICATION ARRAYS: OUTFNC(I), OUTTYP(I), I=1 TO NOUT
```

5. EXAMPLES

A wide class of elliptic PDEs have been solved by GENCOL, INTCOL and HERMCOL. Many results can be found in the references [Houstis et al, 1978], [Houstis et al, 1979], [Houstis et al, 1983] and [Houstis and Rice, 1980]. We include a set of problems to illustrate the usage of this software and to give a general indication of the applicability of the software.

5.1 Example 1: Incompressible Flow in a Circular Tube.

This example involves an elliptic PDE that models an incompressible Newtonian fluid flow in an internally finned circular tube [Masliyah and Kumar, 1980]. The problem is defined by:

PDE: $$u_{xx} + \frac{1}{x^2}u_{yy} + \frac{1}{x}u_x = -1$$

DOMAIN: $[0,1] \times [0, \alpha]$

BC:
$u=0$ at $x=1$ and $0 \le y \le \alpha$
$u=0$ at $y=\alpha$ and $l \le x \le 1$
$u_y=0$ at $y=\alpha$ and $0 < x < l$
$u_x=0$ at $x=0.$ and $0 \le y \le \alpha$
$u_y=0$ at $y=0.$ and $0 < x < 1$

We choose $\alpha=\pi/4$, $l=.5$ and a uniform spaced 32x17 mesh. Note that for this example all three algorithms can be applied. The one used is INTCOL (interior collocation) as it is the most efficient whenever it is applicable. Figure 10 presents a contour plot of the approximation to the unknown true solution of this problem. This problem has been solved with a variety of meshes and the agreement between the solutions for finer meshes is quite good, which suggests the solutions are accurate.

5.2 Example 2: Distribution of Diffused Particles

This example involves a non-self adjoint problem used to model the distribution of diffused particles [Rice et al, 1981]. The problem is defined by

PDE: $$u_{xx} + \frac{1}{x^2}u_{yy} + \frac{2}{x}u_x + (1/(x\tan y))u_y = g$$

DOMAIN: unit square

BC: $u=h$

In order to test the convergence of Hermite collocation, the functions g and h are chosen so that $u=e^{x+y}$. In this case, all three algorithms can be applied. The problem is solved for various meshes and for each mesh various performance indicators are computed. These data are summarized in Table 1. These data indicate that the rate of convergence of the collocation method is of order 3.8. This is similar to the fourth order convergence in the approximation with bicubic Hermite polynomials; order 4 is the highest possible order of convergence. The order is estimated at the i-th grid by

$$\text{order} = \log\left(\frac{\text{error}(i)}{\text{error}(i-1)}\right) / \log\left(\frac{\Delta x_i}{\Delta x_{i-1}}\right)$$

Note that INTCOL has fewer equations and produces an insignificantly different

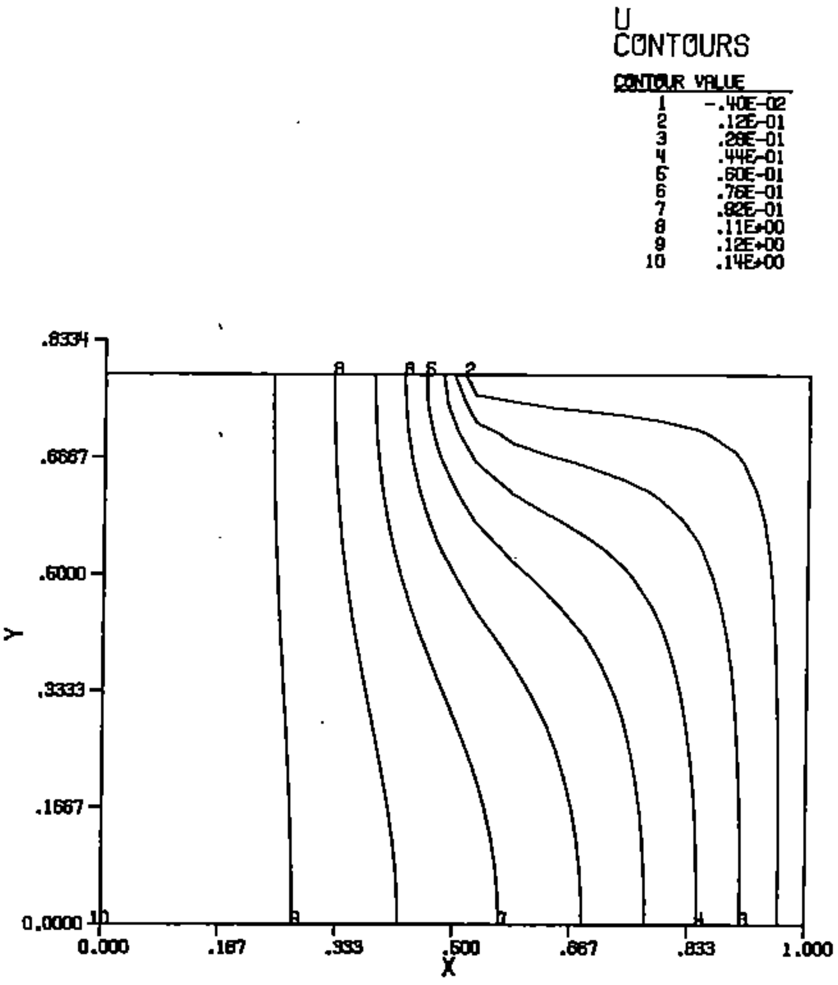


Figure 10. Contour plot of the solution of Example 1 by INTCOL.

solution.

Table 1.. Performance data for Example 2. Time-D is the time in seconds on a VAX 11/780 for discretization, Time-T is the total time for problem solution (excluding I/O). Error and order are estimates of the maximum error and the order of convergence as a function of Δx.

		INTCOL		HERMCOL		GENCOL			
Grid	Equations	Time-D	Time-T	Time-D	Time-T	Time-D	Time-T	Error	Order
3x3	36	.08	.20	.17	.47	.15	.53	2.84e-4	
5x5	100	.13	.75	.28	1.83	.27	1.81	2.39e-5	3.57
9x9	324	.60	5.72	.58	11.94	.72	11.92	1.70e-6	3.81
13x13	676	1.17	22.12	1.10	41.67	1.48	45.38	3.50e-7	3.90
17x17	1156	2.40	57.83	1.62	109.88	2.55	109.47	1.15e-7	3.87

This example is modified to make the domain nonrectangular. The Dirichlet boundary condition is kept the same so the problem is defined by $u = h(x,y)$ on

lines: 1.0,0.0 to 0.0,0.0 to 0.1,0.5 to 0.5,0.5
arc: x=0+.5

The performance results of GENCOL for this modified problem are given in Table 2. These data indicate that the rate of convergence of the collocation method is about 3.7. There is no theoretical basis upon which to base a conjecture about the rate one should expect here, but this example suggests that the convergence may be about the same as collocation on rectangular domains.

Table 2. Performance data for Example 2 with a non-rectangular domain. The notation is the same as for Table 1.

		GENCOL			
Grid	Equations	Time-D	Time-T	Error	Order
3x3	36	.15	.46	1.04e-4	
5x3	60	.27	1.16	1.02e-5	3.78
9x5	176	.67	4.64	7.40e-7	3.78
17x9	560	1.73	21.90	7.96e-8	3.22
15x13	1172	3.38	68.92	1.21e-8	4.65

5.3 Example 3: Flux Distribution in Magnetic Materials (Nonlinear Problem

The caluclation of flux distribution in magnetic material with saturation, leads to the nonlinear elliptic PDE

$$\frac{d}{dx}\left(\frac{1}{\mu}\frac{d\psi}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{1}{\mu}\frac{\partial\psi}{dy}\right) = 0$$

where ψ is the flux function and μ is the permeability which can be developed as the ratio of the magnitudes of the flux vector B and field vector H. It is shown in [Poritsky, 1951] that $B = \left(\psi_s^2 + \psi_y^2\right)^{1/2}$ and $H = \left(\varphi_x^2 + \psi_y^2\right)^{1/2}$ where ψ is the potential function. In [Poritsky, 1951] a number of methods are applied to determine the distribution of magnetic flux in a transformer core with periodic

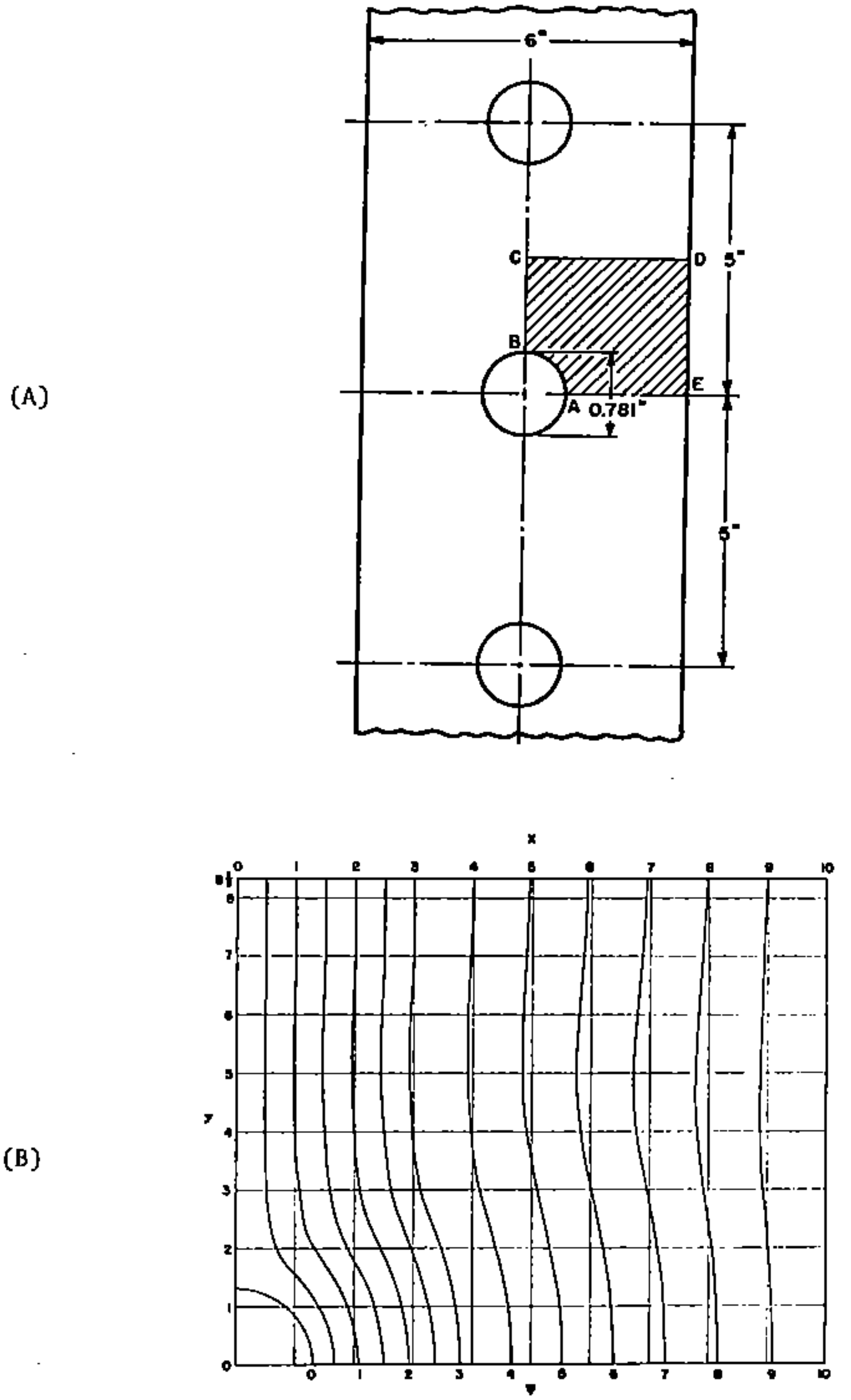


Figure 11. Domains for Example 3. (A) The physical arrangement. (B) The domain after symmetry is exploited.

circular bolt holes (Figure 11A), for an (H,B)-relation shown in Table 3 and average flux density $B_0 = 15{,}000$ lines per centimeter across a 6-inch lamination width. We denote by H_0, μ_0 the values of H, μ corresponding to B_0 ($H_0=3.1, \mu_0=5{,}000$). Because of symmetry, it is sufficient to solve the PDE in the domain shown in Figure 11B with the indicated boundary conditions. A dimensionless form of the problem is obtained by replacing μ by μ/μ_0 where μ_0 is an average value. Similarly, B and H are replaced by B/B_0 and H/H_0 in the dimensionless form.

The PDE can be written in the form

$$\psi_{xx} + \psi_{yy} - \frac{\mu_x}{\mu}\psi_x - \frac{\mu_y}{\mu}\psi_y = 0$$

In this form, μ_0 and H_0 do not enter into the calculations. We select $B_0 = 3333$ so that $\|B\|_\infty$ is about 18000 and Table 3 can be use to calculate μ as a function of B.

The nonlinear problem is solved by the following simple iteration:

```
GUESS $\psi^{(0)} = 10 \cdot XY$
LOOP FOR K = 1 TO L DO
  SOLVE $\psi_{XX}^{(K)} + \psi_{YY}^{(K)} + \frac{\mu_X^{(K-1)}}{\mu^{(K-1)}}\psi_X^{(K)} + \frac{\mu_Y^{(K-1)}}{\mu^{(K-1)}}\psi_Y^{(K)} = 0$ FOR $\psi^{(K)}$
ENDLOOP
```

The term $\mu^{(k-1)}$ is computed by $\mu^{(k-1)}(x,y) = \frac{B^{(k-1)}(x,y)}{H^{(k-1)}(x,y)}$ where $B^{(k-1)}(x,y) = ((\psi_x^{(k-1)}(x,y))^2 + (\psi_y^{(k-1)}(x,y))^2)^{1/2}$ and $H^{(k-1)}(x,y)$ is computed by linear interpolation of Table 3. The terms $\mu_x^{(k-1)}$ and $\mu_y^{(k-1)}$ are computed similarly.

Table 3. Flux distributions with saturation for Example 3.

B, Lines per Square Centimeter	H, Gilberts per Centimeter
500	0.0500
1,000	0.0645
2,000	0.0850
4,000	0.117
6,000	0.149
8,000	0.190
10,000	0.259
12,000	0.399
13,000	0.544
14,000	0.815
15,000	1.67
16,000	6.50
17,000	17.7
18,000	41.0

The four methods used in [Poritsky, 1951] were (1) ordinary finite differences, (2) a graphical method, (3) an analog device, and (4) a hodograph method. None were very satisfactory, but that is no surprise since the paper is over 30 years old. The simple iteration converges very well. Table 4 indicates the convergence as measured by

$$\mathrm{DIFF}(\mathrm{K}) = \max_{(x,y)\in \text{grid points}} |\psi^{(k)}(x,y) - \psi^{(k-1)}(x,y)|$$

The contour plot obtained after 6 iterations with a 13x11 grid is shown in Figure 12. It agrees qualitatively with the crude results obtained by Poritsky.

Table 4. Maximum differences between iterates for Example 3.

K	DIFF(K)
1	6.500e+01
2	3.645e-03
3	3.740e-04
4	1.764e-05
5	2.861e-06
6	1.907e-06

5.4 **Example 4. Gas Lubrication (Nonlinear Problem)**

The Navier-Stokes equations for compressible, non-viscous fluid flow in thin films reduce to Reynold's equation. It models the pressure distribution in the gas films that lubricate high speed devices such as gyroscope bearings or magnetic read heads. The PDE is

$$(uh^3u_x) + (uh^3u_y)_y + c\,(uh)_x = 0$$

with boundary conditions $u=1.0$ everywhere. The function $h(x,y)$ is the height of the thin lubrication film. The parameter c is a physical constant, when c is small (low speed), the problem is easy and when c is large (high speed), the problem becomes quite difficult.

This problem is solved by Newton's method, see [Rice, 1983] and [Rice and Boisvert, 1983] for a derivation of this iteration. The Newton iteration is

```
GUESS $U^0$ (X,Y)
FOR K=0 TO L DO:
   (1) SOLVE THE LINEARIZED PROBLEM
         $U^{(K)}U_{XX} + U^{(K)}U_{YY} + D(X,Y)U_Y + E(X,Y)U_X + F(X,Y)U = G(X,Y)$
   (2) SET $U^{(K+1)} = U(X,Y)$
ENDLOOP
```

where

$$d(x,y) = 2u_x^{(k)} + 3h_x u^{(k)}/h + c/h^2$$
$$e(x,y) = 2u_y^{(k)} + 3h_y u^{(k)}/h$$
$$f(x,y) = u_{xx}^{(k)} + u_{yy}^{(k)} + 3(u_x^{(k)} + u_y^{(k)})/h + ch_x/h^3$$
$$g(x,y) = u^{(k)}(u_{xx}^{(k)} + u_{yy}^{(k)}) + (u_x^{(k)})^2 + (u_y^{(k)})^2 + 3h(u_x^{(k)} + u_x^{(k)})/u^{(k)}$$

We choose as example a simple slider bearing, that is $h(x,y)$ is linear in x and constant in y. The bearing geometry is a square pad with a half disk on the leading edge. Thus we have

$$h(x,y) = 1+2x$$

and the domain is defined by

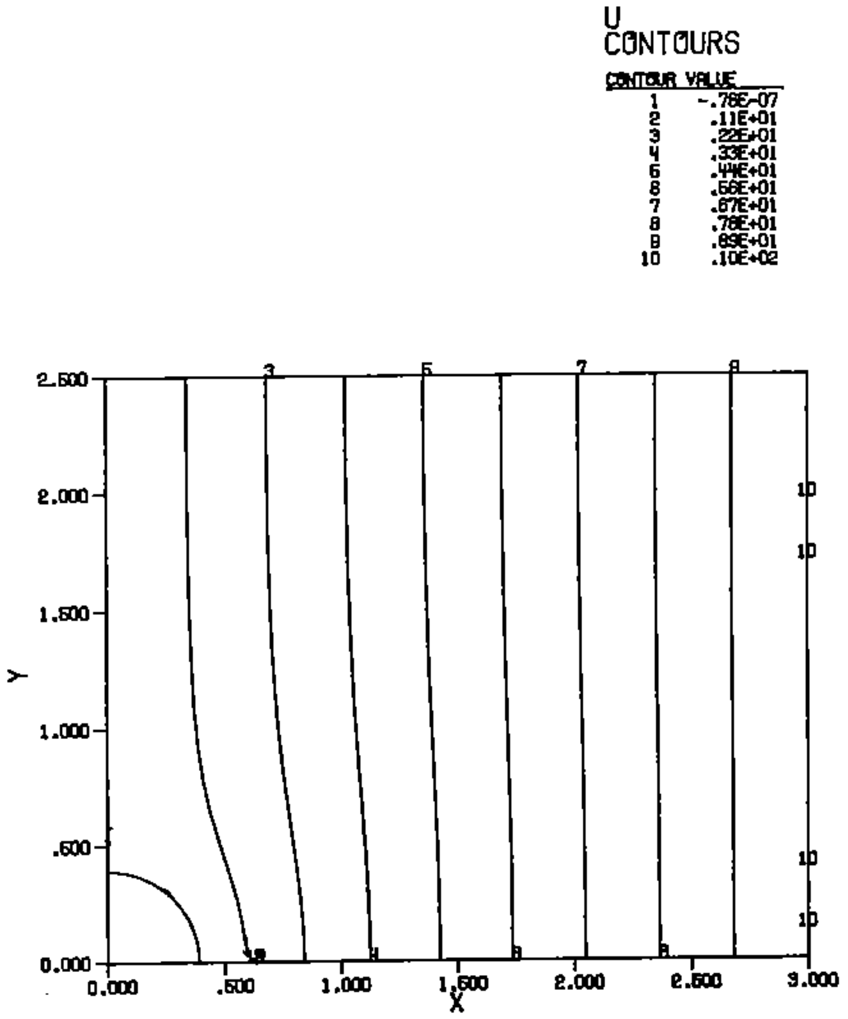


Figure 12. Contour plot of the solution of the nonlinear problem of Example 3.

lines: 1.0,0.0 to 0.0,0.0 to 0.0,1.0 to 1.0,1.0
arc: $x=1.0 + 0.5 \sin p$, $y=0.5(1 + \cos p)$ for $p=0.0$ to π

We also choose c=12.0, a moderate value for this physical parameter.

With the initial guess of $u^{(0)}=1.0$ (no motion at all) the values of DIFF as defined in Example 3 are given in Table 5 for a 12 by 9 grid. We see that the Newton iteration converges rapidly. A contour plot of the final $u(x,y)$ is shown in Figure 13. The problem has been solved again with a 16 x 12 grid; the contour plot is essentially the same and the error using the 12 by 9 grid is about 9.6×10^{-4}. This is estimated by comparing with a solution on a 10 by 10 grid.

Table 5. Maximum differences between iterates for Example 4.

K	DIFF(K)
1	1.418e+00
2	6.064e-02
3	3.116e-03
4	1.102e-03
5	3.788e-06
6	1.295e-07

5.5 Example 5. Minimal Surface (Nonlinear Problem)

The minimal surface equation (or **Plateau problem**) is:

$$(1+u_y)^2 u_{xx} - 2u_x u_y u_{xy} + (1+u_x)^2 u_{yy} = 0$$

Its solution is the shape a soap film takes on a wire loop defined by the boundary conditions, see [Courant, 1950].

We select the example of an elliptical domain and boundary condition that have an upward peak at the top of the ellipse and a downward peak at the bottom of the ellipse. Specifically, the domain and boundary conditions are given by

$$u = e^{-4x^2} \text{ on } x = 2\sin(t),\ y = \cos(t) \text{ for } t = -\frac{\pi}{2} \text{ to } \frac{\pi}{2}$$

$$u = e^{-4x^2} \text{ on } x = 2\sin(t),\ y = \cos(t) \text{ for } t = \frac{\pi}{2} \text{ to } \frac{3\pi}{2}$$

The problem is solved by Newton's method, see, for example, Chapter 5 of [Rice and Boisvert, 1983] for a derivation of the following iteration:

```
GUESS $U^{(0)} = 0$
FOR K = 1 TO L DO
    SOLVE THE LINEARIZED PROBLEM
    $A(X,Y)U_{XX}^{(K)} + B(X,Y)U_{XY}^{(K)} + C(X,Y)U_{YY}^{(K)} + D(X,Y)U_X^{(K)} + E(X,Y)U_Y^{(K)} = G(X,Y)T$
ENDLOOP
```

where

$A = (1+u_y^{(k-1)})^2$

$B = -2u_x^{(k-1)}u_y^{(k-1)}$

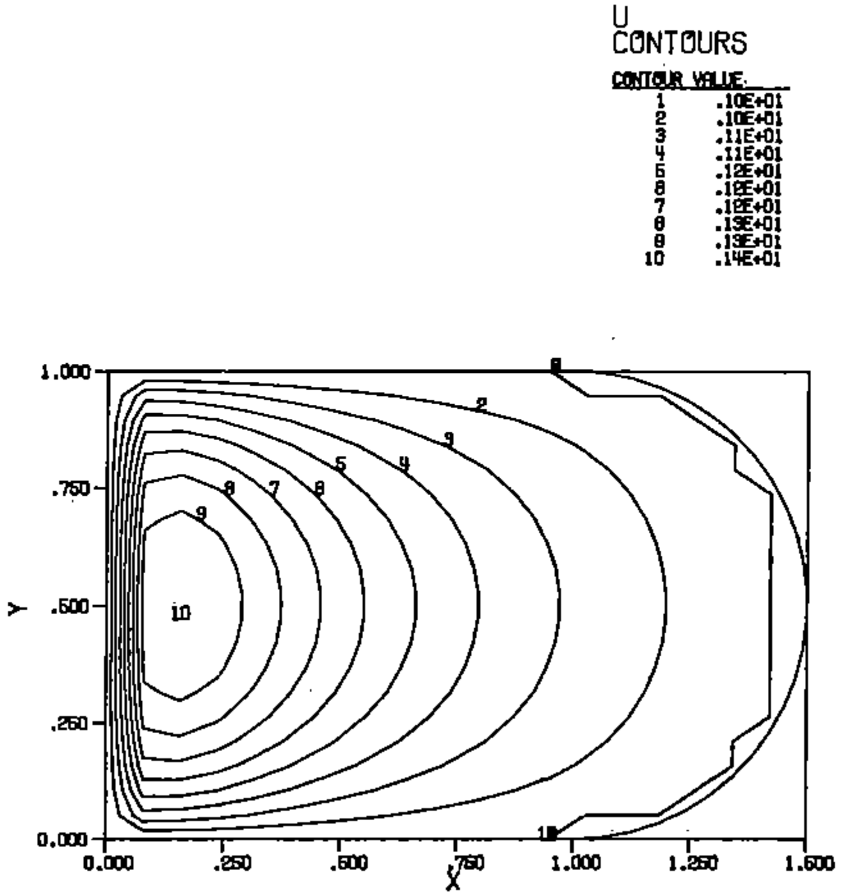


Figure 13. Contour plot of the solution to Example 4, Reynold's equation, computed by Newton's method and GENCOL.

$C = (1+u_x^{(k-1)})^2$

$D = 2(u_x^{(k-1)}u_{yy}^{(k-1)} - u_y^{(k-1)}u_{xy}^{(k-1)})$

$E = 2(u_y^{(k-1)}u_{xx}^{(k-1)} - u_x^{(k-1)}u_{xy}^{(k-1)}$

$G = 2((u_x^{(k-1)}u_{yy}^{(k-1)} - u_y^{(k-1)}u_{xy}^{(k-1)})\, u_x^{(k-1)} + (u_y^{(k-1)}u_{xx}^{(k-1)} - u_x^{(k-1)}u_{xy}^{(k-1)})u_y^{(k-1)})$

Again, the iteration converges well. Table 6 shows the differences between iterates for a 17 by 9 grid. Figure 14 shows the contour plots of the solution for the first 3 iterations. The plot of the final solution is nearly identical to the third plot.

Table 6. Maximum differences between iterates for Example 5.

K	DIFF(K)
1	1.000e+00
2	1.020e-01
3	9.719e-02
4	2.156e-02
5	1.408e-03
6	8.613e-06
7	8.138e-06

6. PERFORMANCE EVALUATION

There are four principal performance questions for software such as considered here: (1) How much time does it take to run? (2) How much memory does it use? (3) How much accuracy does it achieve? (4) How reliable is it? We do not attempt a scientific analysis of reliability at all. We merely note that the three algorithms have been used on a large number of varied problems. The only difficulty that has been observed is in GENCOL's handling of nonrectangular geometry. Sometimes a grid used does not satisfy the assumptions stated in Section 2.3 and must be modified. Less frequently, but still possible, there are domains with sharp corners where considerable care must be taken in selecting the grid overlay so that reasonably accurate results are obtained. We have also noted for very large problems that linear equation solvers which do not do scaling (such as the LINPACK software) may produce unacceptable magnification of round-off errors.

Time and memory use are the standard and most easily handled measures of performance. The algorithms INTCOL and HERMCOL are parameterized by the grid sizes NGRIDX and NGRIDY and time and memory can be expressed in terms of these two parameters. We distinguish two phases of the solution: the discretization (which is fixed for each algorithm) and the solution of the linear system (which may be changed by the user). Table 5 presents basic estimates of performance for these two algorithms. We use the following notation in this table:

NX,NY = NGRIDX-1, NGRIDY-1

Time-D = Time to discretize problem (in units of one arithmetic operation)

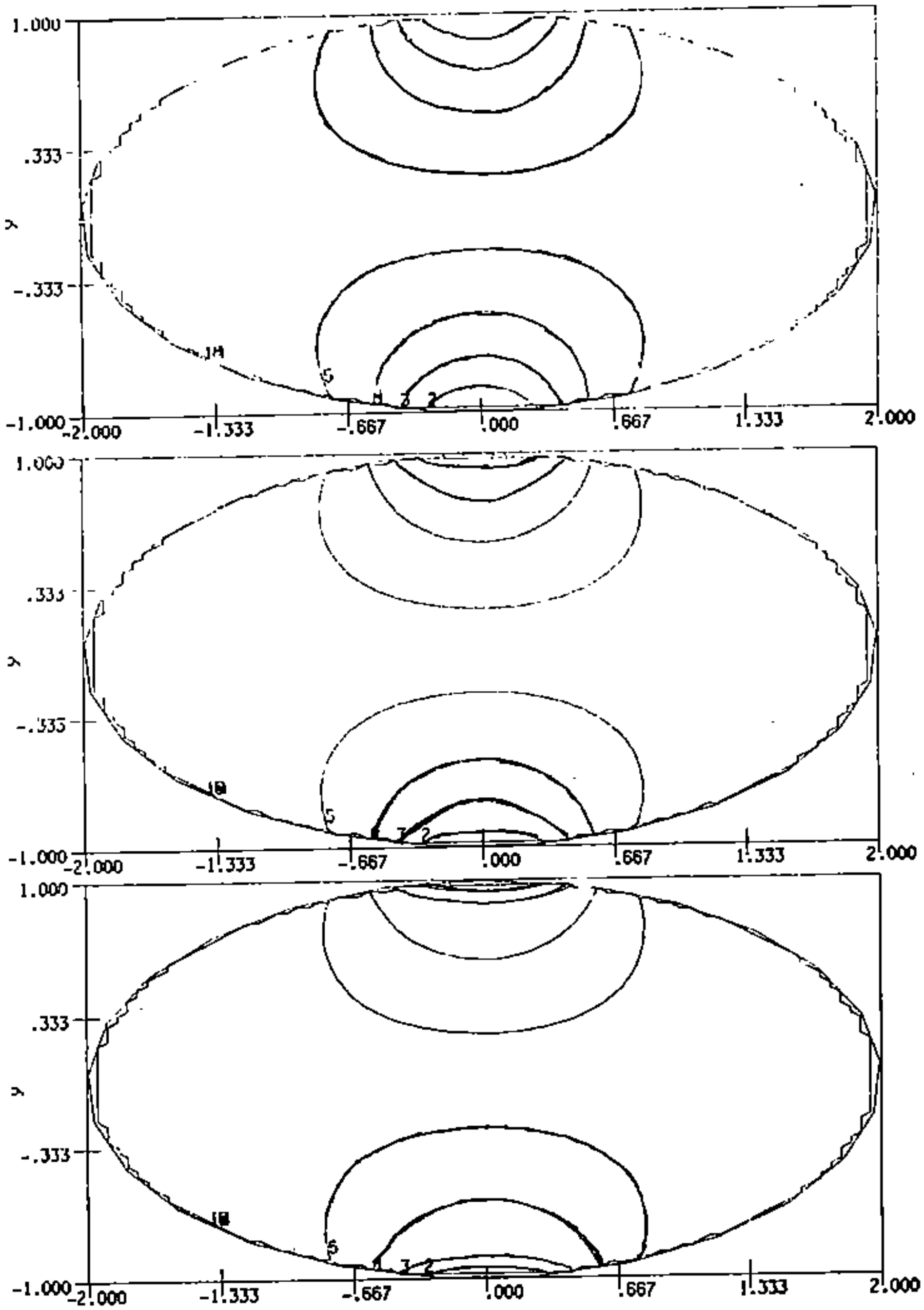


Figure 14. Contour plots for Example 5. The contours for the first three iterations are shown in top to bottom order. The width of the two peaks decreases by a factor of about 3 from the initial Laplace solution estimate (the resolution of the plotter and does not allow this to be seen clearly.

Time-T = Time to solve problem using standard Gauss elimination software for band matrices (in units of one arithmetic operation)

Memory-D = Memory used to discretize problem

Memory-T = Memory used to solve problem

Table 7. Time and Memory Performance Orders for INTCOL and HERMCOL.

	INTCOL	HERMCOL
Time-D	$O(NX*NY)$	$O(NX*NY)$
Memory-D	$34*NX*NY$	$34*(NX+1)*(NY+1)$
Time-T	$O(NX*NY^2)$	$O(NX*NY^2)$
Memory-T	$O(NX*NY^2)$	$O(NX*NY^2)$

The orders given in Table 7 do not distinguish much between INTCOL and HERMCOL; INTCOL is more efficient in both time and memory. The paper [Houstis et al, 1982] gives specific data for a large number of problems.

Time and memory use for GENCOL are less easily parameterized because the shape of the domain R enters. The orders given in Table 7 for HERMCOL are applicable provided (a) the domain is reasonably "compact" and (b) the grid G just covers R. In Table 8, we give specific performance data for three of the problems from Section 5.

Table 8. Time and Memory Performance Data for GENCOL

Ratio	2	3	4	5
Time-$D/(NX*NY)$				
$NX=4$	.017		.067	.021
$NX=8$	.011	.057	.056	.016
$NX=12$	.010	.059	.051	.012
$NX=16$	.010	.056	.059	.010
Memory-$D/(NX*NY)$				
$NX=4$	212.813		214.375	212.813
$NX=8$	172.203	173.469	169.344	163.953
$NX=12$	159.646	159.903	155.319	141.313
$NX=16$	153.551	153.133	148.492	132.926
Time-$T/(NX*NY^2)$				
$NX=4$	.033		.041	.032
$NX=8$	.026	.032	.028	.024
$NX=12$	.028	.032	.024	.016
$NX=16$	.028	.029	.024	.015
Memory-$T/(NX*NY^2)$				
$NX=4$	200.609		196.953	195.797
$NX=8$	109.990	109.287	108.727	107.912
$NX=12$	86.435	86.083	85.666	84.438
$NX=16$	75.782	75.547	75.241	74.208

Accuracy achieved is perhaps the most important measure of performance and yet it is sometimes not considered at all. High efficiency is only meaningful when related to accuracy achieved. Accuracy performance is highly problem dependent, whereas time and memory performance are almost problem independent. Therein lies the difficulty of measuring accuracy performance in a broadly meaningful way, see [Houstis and Rice, 1980] for more details on this topic. There is theoretical reason to expect that collocation on rectangular domains is a fourth order method. That is, the error $|u(x,y) - U(x,y)|$ should be order of $1/N^4$ where N = min(NGRIDX, NGRIDY), provided the problem is well-behaved. This expectation is correct; the data in [Houstis et al, 1982] provides ample evidence of this and there is much more such evidence elsewhere, see, for example, [Rice and Boisvert, 1983].

The accuracy to be expected by GENCOL is unknown at present. It surely should be at least $O(1/N^2)$; it should sometimes be as good as $O(1/N^4)$ and one can hope that it is usually $O(1/N^3)$ - or even $O(1/N^4)$. Figures 15 and 16 show accuracy versus NX and versus total computing time for GENCOL applied to Examples 2,3,4 and 5 of Section 5. The error data is plotted on a log-log scale and the slopes of the data estimate the order of convergence. These figures strongly suggest fourth order convergence (error versus grid size or computer time). The reliability of this suggestion is suspect because this is a very small sample and the errors for three of these are only estimates because the true solutions are unknown.

REFERENCES

Balart, R., Houstis, E.N. and Papatheodorou, T.S. (1982), On the iterative solution of collocation method equations, 10th IMACS World Congress , Vol. 1, pp. 98-100.

Courant, R. (1950), *Dirichlet's Principle, Conformal Mappings, and Minimal Surfaces*, (Pure Appl. Math. Vol. 3) Interscience, New York.

Dyksen, W.R. and Rice, J.R. (1983), A new ordering scheme for the Hermite bicubic collocation equations, in *Elliptic Problem Solvers II* (G. Birkhoff and A. Schoenstat) Academic Press, New York, pp. aa-bb.

Dyksen, W.R., Houstis, E.N., Lynch, R.E. and Rice, J.R. (1982), The performance of the collocation and Galerkin methods with Hermite bicubics, CSD-TR 413, Computer Science, Purdue University.

Gordon, W.J. and Hall, C.A. (1973), Construction of curvilinear coordinate systems and applications to mesh generation, J. Num. Meth. Engr., **7**, pp. 461-477.

Houstis, E.N., Lynch, R.E., Papatheodorou, T.S. and Rice, J.R. (1978), Evaluation of numerical methods for elliptic partial differential equations, Journal of Computational Physics, **27**, (1978), pp. 323-350.

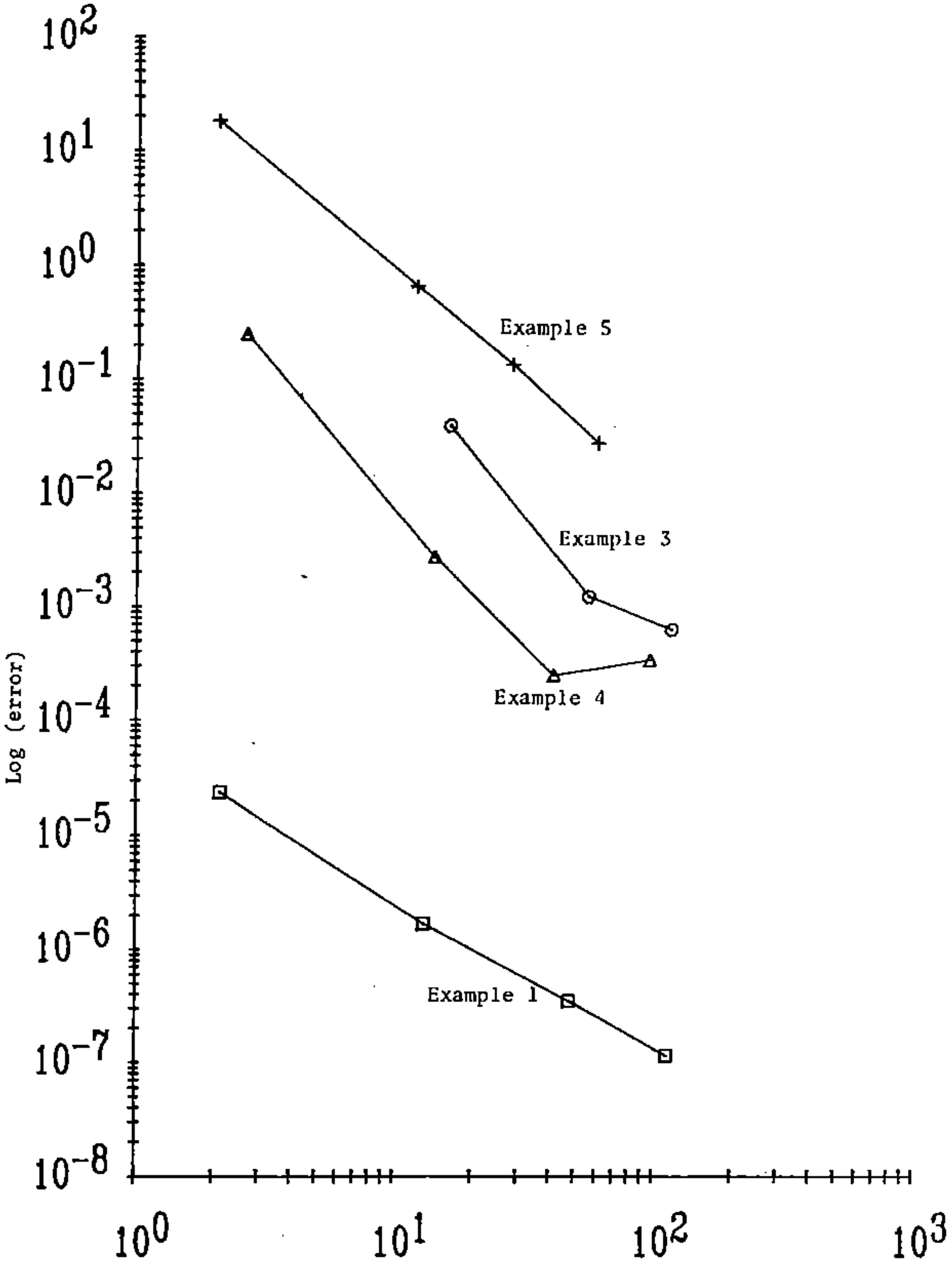


Figure 15. Performance of GENCOL applied to Examples 2, 3, 4 and 5. The plot is log(error) versus log(computing time).

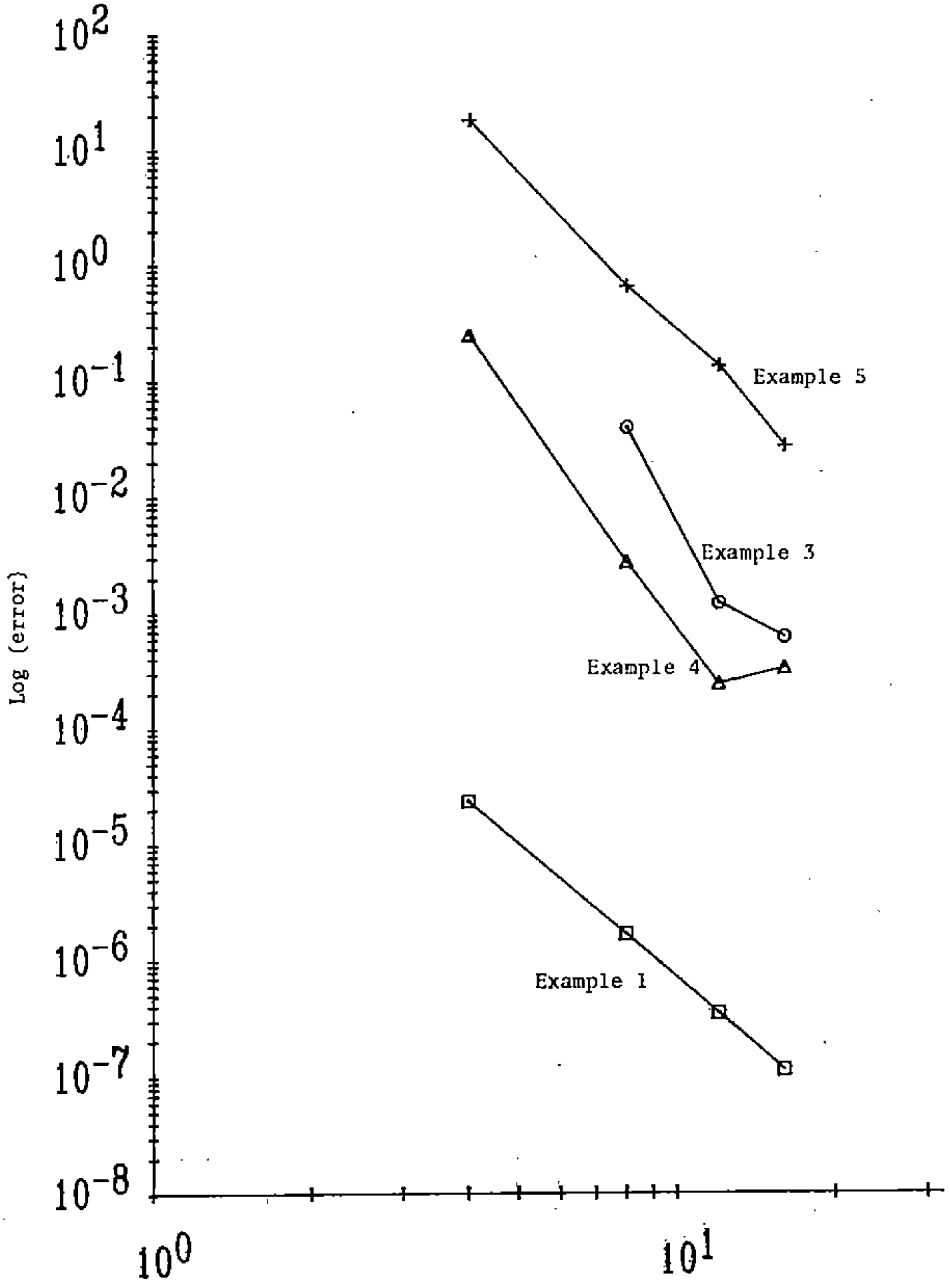


Figure 16. Performance of GENCOL applied to Examples 2, 3, 4 and 5. The plot is log(error) versus log(NX); the slopes of the lines are estimates of the order of convergence of the collocation method.

Houstis, E.N., Mitchell, W.F. and Papatheodorou, T.S. (1979), A C1-collocation method for mildly nonlinear elliptic equations on general domains, in *Advances in Computer Methods for Partial Differential Equations II*, (Vishnevetsky, ed.), IMACS, Rutgers University, pp. xx-yy.

Houstis, E.N. Mitchell, W.F. and Papatheodorou, T.S. (1983), Performance evaluation of algorithms for mildly nonlinear elliptic problems, Journal of Numerical Methods in Engineering, pp. aa-bb.

Houstis, E.N., Mitchell, W.F. and Rice, J.R. (1983a), Algorithm GENCOL: Collocation on general domains with bicubic Hermite polynomials, CSD-TR 444, Computer Science, Purdue University.

Houstis, E.N., Mitchell, W.F. and Rice, J.R. (1983b), Algorithms INTCOL and HERMCOL: Collocation on rectangular domains with bicubic Hermite polynomials, CSD-TR 445, Computer Science, Purdue University

Houstis, E.N. and Rice, J.R. (1980), An experimental design for the computational evaluation of partial differential equation solvers, in *Production and Assessment of Numerical Software*, (Delves and Hennell, eds.), Academic Press, London, pp. 57-66.

Masliyah, J.H. and Kumar, D. (1980), Application of orthogonal collocation on finite elements to a flow problem, Mathematics and Computers in Simulation, **22**, pp. 49-54.

Poritsky, H. (1951), Calculation of flux distributions with saturation, AIEE Transactions, **70**, pp. 309-319.

Rice, J.R. (1982a), Numerical computation with general two dimensional domains, CSD-TR 416, Computer Science, Purdue Univiversity.

Rice, J.R. (1982b), Algorithm: A two dimensional domain processor, CSD-TR 417, Computer Science, Purdue University.

Rice, J.R., (1983a), *Numerical Methods, Software and Analysis*, Chapter 10, McGraw-Hill, New York.

Rice, J.R., (1983b), On the performance of 13 methods to solve the Galerkin method equations, Linear Algebra App., pp. aa-bb.

Rice, J.R. and Boisvert, R.F. (1983), *Solving Elliptic Problems with ELLPACK*, Springer-Verlag, New York.

Rice, J.R., Houstis, E.N. and Dyksen, W.R. (1981), A population of linear, second order elliptic partial differential equatons on rectangular domains, Math. Comp., **36**, pp. 475-484.