1994

General Interior Hermite Collocation Methods for Second Order Elliptic Partial Differential Equations

Yu-Ling Lai

Apostolos Hadjidimos

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

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

Report Number:
94-004

GENERAL INTERIOR HERMITE COLLOCATION METHODS FOR SECOND ORDER ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

Yu-Ling Lai
Apostolos Hadjidimos
Elias N. Houstis
John R. Rice

CSD-TR-94-004
January 1994

GENERAL INTERIOR HERMITE COLLOCATION METHODS FOR SECOND ORDER ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS*

YU-LING LAI†, APOSTOLOS HADJIDIMOS‡, ELIAS N. HOUSTIS‡, AND JOHN R. RICE‡

Abstract. The method of collocation based on C^1 piecewise polynomials has been shown [4, 5, 6] to be an efficient approach to solving general elliptic partial differential equations (PDEs). The iterative solution of the discrete collocation equations is a very challenging problem and only recently it has been fully resolved [7] in the case of rectangular domains. For the case of general PDE domains, the iterative solution of the corresponding discrete equations is still an open problem. In this paper we generalize the method of *interior collocation* for PDEs defined on rectilinear regions, study the structure of these equations under different ordering schemes, and apply AOR and CG type iterative solvers to the discrete equations. One of the ordering schemes introduced here has been successfully applied to iterative solvers for the general discrete collocation equations. A number of numerical experiments are reported that indicate the applicability and effectiveness of the AOR and CG iterative schemes. Moreover, we have identified experimentally the appropriate range of the semi-optimal acceleration parameters and some effective preconditioning matrices. These preliminary results indicate that iterative approaches are efficient in solving the general Hermite collocation equations on general domains.

1. Introduction. In a series of papers [4, 5, 6], Houstis, Mitchell and Rice proposed three algorithms for the numerical solution of the second order linear elliptic partial differential equations (PDEs) on general two-dimensional domains using the cubic Hermite collocation discretization method. Their software is available in the collected algorithms of the ACM. The most general of the above algorithms, called GENCOL, implements the general exterior cubic Hermite collocation approach where the boundary collocation equations are coupled with the interior ones. A simplified version of the GENCOL algorithm, called INTCOL, implements the interior cubic Hermite collocation method when the boundary collocation equations are uncoupled from the rest. The applicability of the INTCOL algorithm is limited to PDEs defined on rectangular domains. One of the main objectives of this work is to extend the INTCOL algorithm for general rectilinear domains (by rectilinear we mean the boundaries are parallel to one of the axes). Throughout, we refer to it by the acronym GINCOL. Moreover, because the ordering of the unknowns and equations in the collocation discretization methods plays a vital role for the numerical solution of the linear system produced, we develop two indexing modules for the GINCOL algorithm. One is based on the finite-element ordering [13] and the other is based on the tensor-product ordering [8]. The collocation coefficient matrix based on a finite-element ordering for the GINCOL algorithm is in general non-symmetric and is not diagonally dominant; many of its diagonal entries are zero. Thus, a straightforward application of the classical point iterative methods for the solution of the corresponding linear system is not possible. Currently, these systems are solved by Gauss elimination with scaling and partial pivoting [3]. Using the tensor-product ordering, the linear system derived by the GINCOL algorithm generates the same block structure that is produced by INTCOL. Motivated by the recent successful application of the classical block iterative methods for the INTCOL algorithm [7], we explore the applicability and the convergence properties of the block iterative methods for GINCOL applied to model

* This work was supported by AFOSR 91-F49620 and NSF grant CCR 86-19817

† Department of Mathematics, Purdue University, West Lafayette, IN 47907

‡ Department of Computer Sciences, Purdue University, West Lafayette, IN 47907.

problems defined on L-shaped domains as well as for a few more general rectilinear domains. Furthermore, the tensor-product ordering was successfully applied to the discrete equations produced by GENCOL together with the AOR and CG iterative solvers. A number of experiments were carried out to study the computational behavior of these iterative schemes and to estimate the various parameters involved.

The organization of this paper is as follows. In Section 2, we briefly describe the basic idea of the cubic Hermite collocation method. In Section 3, we outline the GENCOL algorithm in [4], [5] (see also [11]). Next, in Section 4, we formulate the GINCOL algorithm. In Section 5, two different indexing modules to be used with GINCOL are developed and one tensor-product ordering is introduced for the GENCOL algorithm. Finally, in Section 6, a wide class of PDE problems are solved by using the GINCOL algorithm with some block iterative linear solvers and a number of concluding remarks are made based on observations from these experiments.

2. The Cubic Hermite Collocation Method. Suppose we are given the second order linear elliptic PDE

$$\begin{aligned} Lu &\equiv au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu = g \ \text{ in } \ \Omega \\ Bu &\equiv \alpha u + \beta \tfrac{\partial u}{\partial n} = \delta \qquad\qquad\qquad\qquad \text{on } \partial\Omega \end{aligned}$$

where Ω is a bounded region in the k-dimensional space and $\partial\Omega$ is the boundary of Ω. The method of collocation consists of finding a function u_h in a finite dimensional approximate solution subspace of the space of square integrable functions on Ω. The function u_h is chosen so that $L(u_h) = g$ and $B(u_h) = \delta$ are satisfied exactly at certain interior and boundary points, respectively. These points are called collocation points. There are many ways to select the approximate solution subspace and the collocation points. Throughout this paper we use the subspace of cubic Hermite piecewise polynomials which defines the cubic Hermite collocation method. This method has been shown to be highly accurate for some second order elliptic PDE problems (see [9] and [10]). For brevity in the sequel, when we refer to the collocation method without any further explanation, we mean the cubic Hermite collocation method.

The finite-element mesh Ω_h is a set of intervals, rectangles and rectangular parallelepiped regions for 1-D, 2-D and 3-D problems, respectively. The exact definition of Ω_h is given in the next section. The approximate solution u_h is defined on each mesh element in terms of one-dimensional local basis functions ϕ_1, ϕ_2, ϕ_3 and ϕ_4 defined on the interval (t_0, t_1) as follows:

$$\begin{aligned} &\phi_1(t) := (1 - \tfrac{t-t_0}{t_1-t_0})^2(1 + 2\tfrac{t-t_0}{t_1-t_0}),\ \phi_2(t) := (t-t_0)(1 - \tfrac{t-t_0}{t_1-t_0})^2, \\ &\phi_3(t) := (1 + \tfrac{t-t_1}{t_1-t_0})^2(1 - 2\tfrac{t-t_1}{t_1-t_0}),\ \phi_4(t) := (t-t_1)(1 + \tfrac{t-t_1}{t_1-t_0})^2. \end{aligned}$$

The corresponding expressions for u_h are

$$\begin{aligned} u_h(x) &= \textstyle\sum_{i=1}^{4} \rho_i \phi_i(x), && \textit{for } 1\text{-D elements}, \\ u_h(x,y) &= \textstyle\sum_{i,j=1}^{4} \rho_{ij} \phi_i(x)\phi_j(y), && \textit{for } 2\text{-D elements}, \\ u_h(x,y,z) &= \textstyle\sum_{i,j,z=1}^{4} \rho_{ijk} \phi_i(x)\phi_j(y)\phi_k(z), && \textit{for } 3\text{-D elements}. \end{aligned}$$

From the definition of the basis functions it is clear that there are 2, 4 and 8 unknowns associated with each node for the 1-D, 2-D and 3-D cases, respectively. Furthermore, one can easily show that the values of the unknown ρ's coincide with the values of the

approximate solution and its derivatives at the nodes. For example, let $(\rho_1, \rho_2, \rho_3, \rho_4)$ be the four unknowns associated with a node q on a 2-D domain, then

$$\rho_1 = u_h(q),\ \rho_2 = \frac{\partial u_h}{\partial y}(q),\ \rho_3 = \frac{\partial u_h}{\partial x}(q),\ \rho_4 = \frac{\partial^2 u_h}{\partial x \partial y}(q).$$

From the definition of the basis functions, we can easily see that the second derivative of u_h is not continuous at the element boundaries. On the other hand, using Gaussian quadrature theory [9], higher accuracy is obtained if the interior collocation points are located at the Gaussian points of the mesh element rather than at the grid nodes. As for the placement of the boundary collocation points, we follow the scheme suggested in [4]. One of the restrictions is that the number of these points must be equal to the difference of the dimension of the approximate solution subspace and the number of interior collocation points.

3. GENCOL: The General Collocation Method for a General 2-D Domain. The procedure of solving a PDE problem by the general collocation method can be roughly broken into the five steps indicated below (see [4]):

(1) define the PDE problem,
(2) place a rectangular grid over the domain of definition,
(3) generate the finite-element mesh,
(4) locate the collocation points and form the linear system,
(5) solve the linear system.

Steps (3) and (4) are the ones that constitute the core of the general collocation method. A detailed description of these two steps follows.

First we overlay the domain Ω by a rectangular grid G and identify the rectangular elements of G that are interior or exterior to $\partial\Omega$ or that intersect $\partial\Omega$. The latter ones are called boundary elements. It might happen that the intersection of certain boundary elements with Ω is very small. Their inclusion as elements of the finite-element mesh Ω_h will not only enlarge the linear system to be solved but may, in some extreme cases, also cause numerical instability in its solution. It is thus natural to discard those boundary elements which may cause trouble. We define the finite-element mesh Ω_h as the union of the interior elements and those boundary elements e_b for which the ratio of the area of $e_b \cap \Omega$ over the area of e_b is greater than a certain amount called $DSCARE$. The portions of $\partial\Omega$ in the discarded elements are either allocated to neighboring elements or ignored. This is controlled by a logical variable called $GIVOPT$ ($GIVOPT = .TRUE.$ means allocate to neighboring elements). Note that by using this "discarding" procedure some elements may change from boundary to exterior or from interior to boundary. To assure the implementation of this procedure, some assumptions must be satisfied (see [4]):

- The boundary $\partial\Omega$ of Ω, consisting of at least two pieces, is given in a parameterized form in a clockwise manner.
- A boundary element does not contain a whole boundary piece of Ω, and there are at most two boundary pieces in it.
- The sides of a boundary element which are treated as pieces of the boundary of Ω_h must be adjacent and the number of them is at most three.
- If a boundary element is discarded, then no more than two of its neighboring elements can be interior elements.

- The boundary does not enter an element more than once, except when it leaves the element and reenters it along the same element edge. Further the neighboring element to this edge is discarded.

The above assumptions are usually satisfied for a reasonably fine mesh. Below we present a code outline for the above procedure ([4]). For this a rectangular element of G is identified by the indices (IX, JY) of its lower left corner grid point, where $1 \leq IX \leq$ *number of x-grid lines* and $1 \leq JY \leq$ *number of y-grid lines.*

```
LOOP: FOR EACH BOUNDARY POINT B_l DO :
  IF THE BOUNDARY LEAVES AN ELEMENT AND ENTERS
     A NEW ELEMENT (IX, JY) AT THIS POINT
     THEN SAVE THE BOUNDARY POINT INDICES FOR
       THE NEW ELEMENT AS
       ELTYPE(IX, JY) = IENTER + 1000 × IEXIT
       WHERE IENTER AND IEXIT ARE THE INDICES OF
       THE BOUNDARY POINTS WHERE THE BOUNDARY
       ENTERS AND EXITS THE ELEMENT (IX, JY)
  ENDIF
ENDLOOP ;

LOOP: FOR EACH ELEMENT (IX, JY) OF G DO :
CASE TYPE OF ELEMENT (IX, JY)
  EXTERIOR: ELTYPE(IX, JY) := −1 /* do not use element */
  INTERIOR: ELTYPE(IX, JY) := 0 /* use element */
  BOUNDARY:
    IF (AREA OF ELEMENT INTERSECTION) / (AREA OF ELEMENT) < DSCARE
       THEN ELTYPE(IX, JY) := −ELTYPE(IX, JY)
            /* do not use element */
       ELSE ELTYPE(IX, JY) := (IENTER + 1000 * IEXIT)
            /* the element is used with ELTYPE unchanged */
    ENDIF
ENDCASE;
ENDLOOP;

LOOP: FOR EACH BOUNDARY SEGMENT DO :
  /* if segment is in element (IX, JY) and ELTYPE(IX,Y) < −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 TWO PIECES
     ELSEIF GIVOPT = .TRUE.
       THEN THE BOUNDARY SEGMENT IS SPLIT BETWEEN
       THE TWO ELEMENTS WHOSE ASSOCIATED
       BOUNDARY SEGMENTS ARE CONNECTED TO IT
     ENDIF
ENDLOOP
/* note : if GIVOPT = .FALSE. then the piece of the boundary in
the discarded element is not used */
```

Now, we can determine the interior collocation points on $\Omega_h \cap \Omega$. We split the points into two groups. One group consists of all the sets of the four Gaussian points on the corresponding interior mesh elements. Since the four Gaussian points in a boundary mesh element e_b might not be in Ω, a mapping from e_b onto $e_b \cap \Omega$ is

necessary. Thus, the other group of elements is composed of the images of the four Gaussian points of each boundary element under this mapping. The map depends on several aspects of the geometry and is too complicated to give a detailed description here (see [4]). However, the main idea is the following: First, the boundary $\partial(e_b \cap \Omega)$ is partitioned into four parts and each side of e_b is mapped by a one-to-one mapping onto one of those parts. Then, the map from e_b to $e_b \cap \Omega$ is determined by linearly blending those four maps of the boundary.

To locate the boundary collocation points, one has to compute the number of boundary points such that the total number of collocation points is equal to the number of the unknowns. Let N_v and N_e be the numbers of nodes and mesh elements, respectively, on the finite-element mesh Ω_h. Since there are four unknowns associated with each node and we set four interior collocation points on each mesh element, it follows that there are $4N_v - 4N_e$ boundary collocation points that need to be determined. On the other hand, it can be shown using the Euler-Poincare characteristic of the regular region of a surface [1] that $N_e - N_s + N_v = 1 - N_h$, where N_s is the number of element sides of Ω_h and N_h is the number of holes of Ω_h. Furthermore, it is easy to find that $N_s = B_s + I_s$ and $4N_e = B_s + 2I_s$, where B_s and I_s are the numbers of element sides on ∂E and in the interior of Ω_h, respectively. A little manipulation using these relations shows that

$$4N_v - 4N_e = 2B_s + 4(1 - N_h).$$

The procedure of determining the boundary collocation points consists of two passes. The first pass is to place the collocation points on the boundary of Ω_h. The second pass is to map the boundary sides of a boundary element of Ω_h onto the boundary segment of Ω associated with this element. Then the images of the collocation points placed by the first pass are the boundary collocation points sought to generate the boundary collocation equations. A more detailed description of these two passes in code form is presented below (see [4]).

```
PASS 1: /* associate boundary collocation points (BCPS) with boundary of finite element mesh */
   PLACE TWO BCPS ON EACH BOUNDARY SIDE OF Ω_h IN
   THE SAME CONFIGURATION AS PARAMETERS BCP1 AND
   BCP2 ARE PLACED IN THE INTERVAL (0,1)
   PLACE ONE BCP AT EACH CORNER OF ∂Ω ∩ E
   IF THE END OF THE LAST BOUNDARY SIDE IS A CONCAVE
     CORNER OF THE FINITE ELEMENT MESH
     THEN REPLACE THE TWO BCPS OF THE LAST
     BOUNDARY SIDE WITH ONE BCP AT THE MIDPOINT OF
     THE SIDE
   ENDIF
   IF THE BEGINNING OF THE FIRST BOUNDARY SIDE
     IS A CONCAVE CORNER OF THE FINITE ELEMENT MESH
     THEN MOVE THE TWO BCPS 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
   ENDIF
/* 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 BCPS from ∂Ω_h to ∂Ω */
```

/* this is a mapping from (0,1) to the segment of $\partial\Omega$ associated with an element of Ω_h */
IF THE SEGMENT OF $\partial\Omega$ IS CONTAINED IN ONE PIECE OF THE BOUNDARY
THEN LINEARLY MAP (0,1) TO $(PENTER, PEXIT)$
DETERMINE THE BCPS FROM THE PASS 1 VALUES AND THE DEFINITION OF $\partial\Omega$
ELSEIF THE SEGMENT OF $\partial\Omega$ 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 THE PARAMETRIZED FORM OF BOUNDARY PIECE.
DETERMINE THE BCPS FROM THE PASS 1 VALUES AND THE DEFINITION OF $\partial\Omega$
ELSE ERROR /* allow no more than two boundary pieces in a element */
ENDIF

It is easy to see that the above procedure does give $2B_s + 4(1 - N_h)$ boundary collocation points. The user is allowed to adjust the placement of the boundary collocation points in a boundary edge by changing the two parameters $BCP1$ and $BCP2$. The default case ($BCP1 = BCP2 = 0$) selects two Gaussian points in a boundary edge.

Once the collocation points are determined, to generate the collocation equations is a simple task. The collocation equations are represented by the following arrays :
$COEF(n,l)$ = lth coefficient value of equation n
$IDCO(n,l)$ = index of the unknown associated with $COEF(n,l)$
$BBBB(n)$ = right hand side value of equation n

4. GINCOL:The General Interior Collocation Method for a Rectilinear Domain. The method presented in the previous section can be simplified in case (i) the domain Ω is rectilinear, and (ii) the problem has uncoupled boundary conditions, that is, at no point are the boundary conditions mixed, i.e.,

$u \equiv \delta \ \ on \ \ \partial\Omega_1 \subset \partial\Omega$,
$\frac{\partial u}{\partial n} \equiv \delta \ \ on \ \partial\Omega_2 = \partial\Omega - \partial\Omega_1 \subset \partial\Omega$.

In order to distinguish this case from the general collocation method case, the simplified version is called interior collocation. First, we use the algorithm in the previous section to generate a finite-element mesh Ω_h. Then, since an entire boundary piece is either horizontal or vertical, some unknowns associated with nodes on a boundary piece can be determined beforehand using the following two assumptions:
(i) The boundary condition changes type only at a boundary node.
(ii) The boundary of the mesh Ω_h coincides with the boundary of the domain Ω.
The assumption (ii) is satisfied for the domain Ω when its boundary pieces are contained in the union of the grid lines of the mesh. So the user is simply required to place a grid line on each boundary piece of the domain Ω as part of the discretization. In this case, the boundary collocation equations can be solved explicitly when the discretization of the boundary conditions takes place. Here we adapt the code in [4] for the rectangular domain case, the new code is:

LOOP OVER EACH BOUNDARY PIECE:
LOOP1 OVER EACH NODE T_i OF THE BOUNDARY PIECE:

```
      DETERMINE THE LEFT OR RIGHT HALF-INTERVAL
      ([T_{i-1/2}, T_i] OR [T_i, T_{i+1/2}]) WHERE THE BOUNDARY
      CONDITION IS OF THE SAME TYPE AS AT T_i.
      /* denote the interval by Δ and its two Gauss points by τ1, τ2 */
      S = {τ1, τ2 AND END POINTS of Δ};
      CASE BOUNDARY CONDITION TYPE IS:
        DIRICHLET (U = δ): DETERMINE U_x (OR U_y) AT T_i
          BY INTERPOLATING δ BY A CUBIC POLYNOMIAL AT
          THE POINTS S; IDENTIFY THE ACTIVE UNKNOWNS;
        NEUMANN (∂U/∂N = δ): DETERMINE U_xy (= U_yx) AT T_i
          BY INTERPOLATING δ BY A CUBIC POLYNOMIAL AT
          THE POINTS S; IDENTIFY THE ACTIVE UNKNOWNS;
      ENDCASE;
    ENDLOOP1;
  ENDLOOP;
```

Since the boundary element e_b coincides with $e_b \cap \Omega$, we can simply select the four Gaussian points on each mesh element as the interior collocation points. Note that there are three unknowns associated with a concave corner of Ω and they have been solved for in the boundary discretization procedure. This makes the corresponding linear system over-determined. To derive a completely determined linear system, we pretend that there are only one unknown solved at a non-convex corner during the boundary discretization procedure according to the following rule : *if (U solved) then the three unknowns are U_y, U_x and U_{xy} else the three unknowns are U, U_x and U_{xy}*. Finally, we are left with the task of generating $COEF$ and $IDCO$ and then eliminating the nonactive unknowns, namely those predetermined during the boundary discretization process from $BBBB$. There are three local two-dimensional arrays that are used for this task.

$NODELM(i,l)$ = the global index of the ith local node in element l

$INUNKN(i,n)$ = the global index of the ith local unknown associated with node n

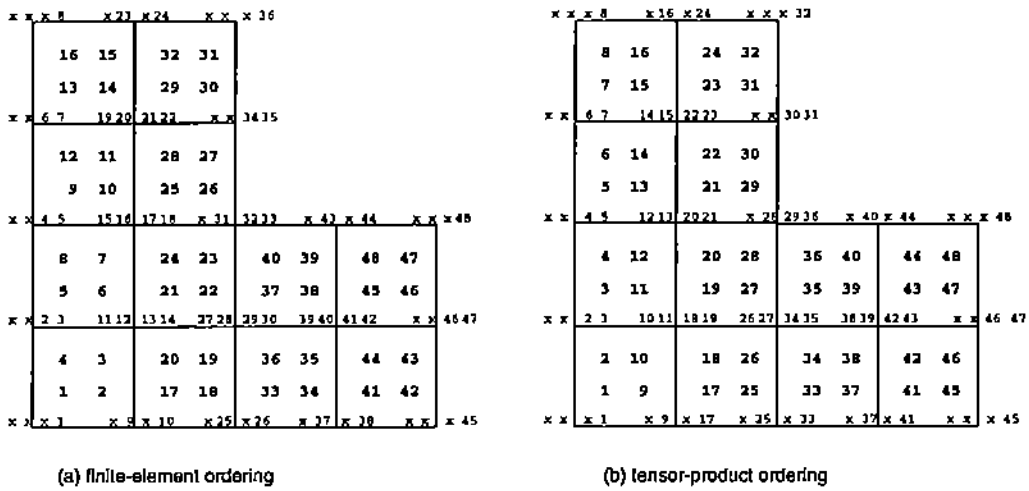
$OLUNKN(i,n)$ = the value of the nonactive ith local unknown associated with node n

A code skeleton for this procedure is:

```
LOOP OVER ELEMENTS OF Ω_h:
  GENERATE NODELM, COEF and BBBB
  IF INTERIOR ELEMENT
    THEN GENERATE INUNKN ASSOCIATED WITH THE
    LOWER LEFT NODE OF THIS ELEMENT
  ELSE GENERATE INUNKN ASSOCIATED WITH THE LOWER
    LEFT NODE OF THIS ELEMENT AND INUNKN
    ASSOCIATED WITH OTHER NODES OF THIS ELEMENT
    ON THE BOUNDARY.
    FOR THE NONACTIVE UNKNOWN SET INUNKN TO
    ZERO AND SUPPLY THE VALUE OF OLUNKN
  ENDIF
ENDLOOP;
GENERATE IDCO AND MODIFY BBBB BASED ON NODELM,
INUNKN AND OLUNKN
```

5. The Ordering of Unknowns and Equations. The properties of the coefficient matrix of the linear system arising from the discretization of a PDE problem by the collocation method strongly depends on the ordering of the unknowns and equations. A specific ordering may produce a linear system suitable for an iterative solver while the same iterative solver might not be applicable to the linear system

obtained by another ordering. It appears that there are three basic approaches to the ordering of the unknowns and the equations for the collocation method. We only discuss two of them. A detailed description of the third one is found in [2]. Before giving a detailed description of those two orderings, we depict the numbering of the unknowns and equations on a L-shaped domain with Dirichlet boundary conditions in Figure 1 for GINCOL and Figure 2 for GENCOL. Collocation points are shown in bold and their numbering indicates the ordering of the equations. The unknowns are associated with nodal points and are numbered in light font style. Those unknowns eliminated symbolically during the discretization of boundary conditions are denoted by x.

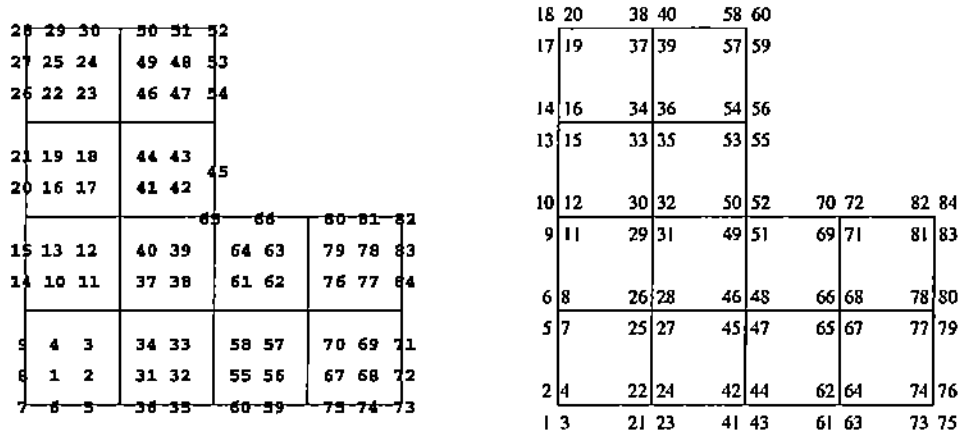


(a) finite-element ordering (b) tensor-product ordering

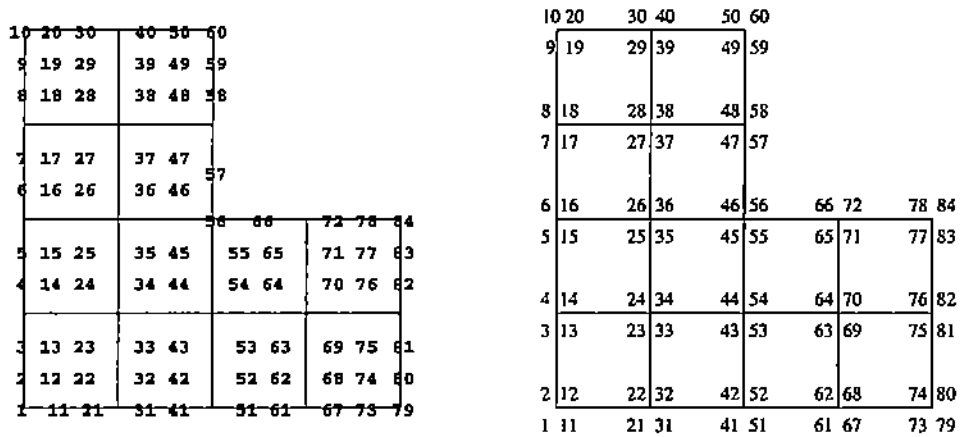
FIG. 1. *Two orderings of the collocation points and unknowns associated with GINCOL.*

One of the orderings is obtained by a natural extension of the finite-element ordering in [13] to the general domains. We call it the *finite-element ordering.* More specifically, once the finite-element mesh is defined, the mesh nodes and the mesh elements are numbered in a natural way from south to north, west to east. Note that there are four unknowns associated with a mesh node in the algorithm GENCOL. Thus, the unknowns are numbered in groups of four in the order of the corresponding mesh node. The four unknowns associated with a mesh node are locally ordered so they respectively represent the values of u, u_y, u_x ans u_{xy} at the mesh node. In this ordering the GENCOL collocation points are numbered element by element following the element numbering in the mesh. In the case of boundary elements the interior collocation points are numbered first counter-clockwise followed by the clockwise numbering of boundary collocation points. Figure 2a displays this ordering for a finite element mesh of an L-shaped region. This ordering can be extended directly to GINCOL. In this case, there are fewer unknowns because some of them corresponding to boundary nodes can be eliminated symbolically. The GINCOL active unknowns are numbered node by node following the node numbering of the finite element mesh. In the case of GINCOL we have only interior collocation points which are numbered element by element. Figure 1a displays the numbering of unknowns and interior collocation points.

The second ordering is called the *tensor-product ordering.* This scheme was originally defined in [8] for the interior collocation method on a rectangular domain and



(a) finite-element ordering



(b) tensor-product ordering

FIG. 2. *Two ordering of the collocation points and unknowns associated with GENCOL.*

is extended to the general collocation method on rectangular domains in [7]. Here, we utilize it for the algorithms GINCOL and GENCOL in a straightforward manner. First, the GENCOL unknowns are split into two sets $\{u, u_y\}$ and $\{u_x, u_{xy}\}$. Then, on each x-grid line we number the unknowns $\{u, u_y\}$ node by node (south to north) followed by the numbering of $\{u_x, u_{xy}\}$ unknowns corresponding to the nodal points of the same grid line. For the tensor-product numbering of the GENCOL collocation points we consider the auxiliary exterior boundary collocation points introduced in [4] to determine the actual boundary points. By definition the auxiliary and interior collocation points are located on x-Gauss grid lines corresponding to x-coordinates of the Gauss points. Then, these points are numbered along the x-Gauss grid lines from south to north and west to east. The indices of the actual boundary collocation points and the auxiliary boundary points coincide. Figure 2b displays this scheme for an L-shaped region.

In the case of GINCOL, we have only interior collocation points, thus they are ordered from south to north along x-Gauss grid lines as in the case of GENCOL. Then the numbering of the active unknowns is determined by the indices of the interior collocation points as follows. At each nodal point, the active unknowns use the same index as the nearest interior collocation points. Figure 1b illustrates this ordering

scheme for an L-shaped region.

The finite-element ordering is attractive because it yields a coefficient matrix which has smaller bandwidth than the one using the tensor-product ordering. The advantage of the tensor-product ordering is that the corresponding coefficient matrix for the GINCOL and GENCOL algorithms has the block structure indicated in Figure 3. For more general domains, the structure of (b) depends very much on the placement of the boundary collocation points. Figure 4 shows the detailed structure of the coefficient matrix for GINCOL in the case of the L-shaped domain of Figure 1. For GINCOL the diagonal blocks of the coefficient is always a band matrix with bandwidth 2 and non-zero diagonal elements. Some block iterative linear solvers may benefit from this property.

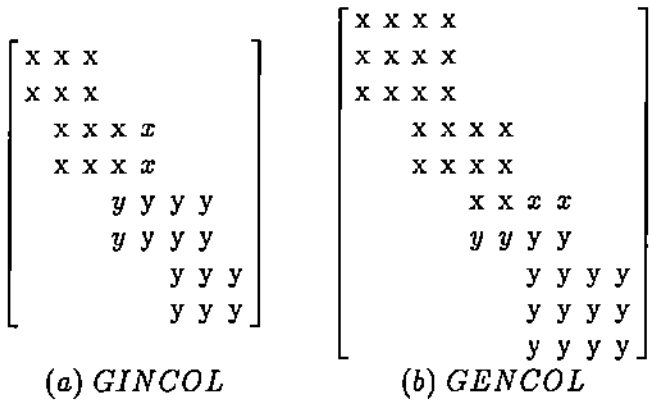


FIG. 3. *The structure of GINCOL and GENCOL equations assuming tensor-product ordering. The entries x,x, y and y denote* 8×8, 8×4, 4×8 *and* 4×4 *submatrices in (a) and* 10×10, 10×6, 6×10 *and* 6×6 *submatrices in (b) respectively.*

Figure 5 shows the detailed structure of the coefficient matrix for GINCOL in the case of the L-shaped domain of Figure 1. The finite-element ordering provides the efficiency of bandedness but the presence of many zeros on the diagonal of the coefficient matrix prevents most iterative methods from being applied. So, the most reliable and preferable way to solve the linear system is to use Gauss elimination with scaling and partial pivoting [3]. However, direct methods tend to require much more memory as well as more time and their parallelization is difficult. It is very desirable to have a suitable iterative solver for the collocation equations in general, this can be accomplished using the tensor-product ordering.

6. Application of Iterative Linear Solvers. In this section, we use the algorithm GINCOL developed in the previous section to discretize a number of elliptic PDEs with uncoupled boundary conditions on a L-shaped domain Ω_1 as well as on a general rectilinear domain Ω_2 shown in Figure 6. We consider only the tensor-product ordering as the finite-element ordering prevents us from applying an iterative linear solver.

For the iterative solution of the GINCOL equations, we consider two approaches: the overrelaxation, AOR(SOR)-type, approach and the conjugate gradient, CG-type, approach. Among the AOR-type methods and in view of the block structure of matrix (a) in Figure 3, it is customary to use a block iterative method instead of point

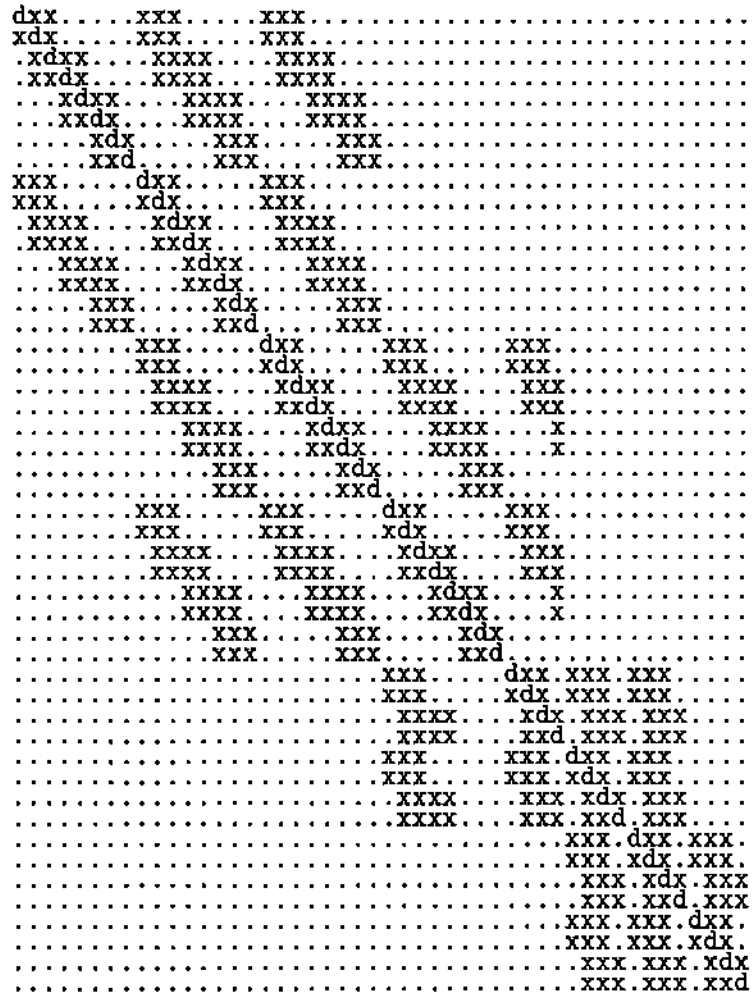


FIG. 4. *The detailed structure of the collocation matrix derived from GINCOL using the tensor-product ordering for the example in Figure 1(b).*

iteration. Three different block partitionings from [7] of the collocation coefficient matrix are considered. They are denoted by P_I, P_{II} and P_{III}, respectively, and defined in Figure 6.

Among the CG-type methods, the preconditioned GMRES (generalized minimal residual) method [12] is an often successful method for solving nonsymmetric linear systems. The preconditioner used should be easily inverted and the diagonal blocks of P_I, P_{II} and P_{III} can be used. After experimentation we conclude that P_{II} preconditioner is the best for GMRES.

In Tables 1 to 3, we study the convergence behavior of SOR under different block partitionings of the GINCOL collocation matrix in different rectilinear domains. Specifically, we display the maximum discretization error $||u-u_h||_\infty$ based on the grid points inside the domain, where u is the exact solution of the PDE problem and u_h is the computed cubic Hermite piecewise polynomial solution. These tables also give the number of iterations required for the various methods to converge. These numbers are good indications of the actual efficiencies of the methods. The mesh size entry is the size of the mesh in the smallest rectangle that contains the domain. The values in parentheses beside them are the orders of the linear systems. For the adaptive SOR

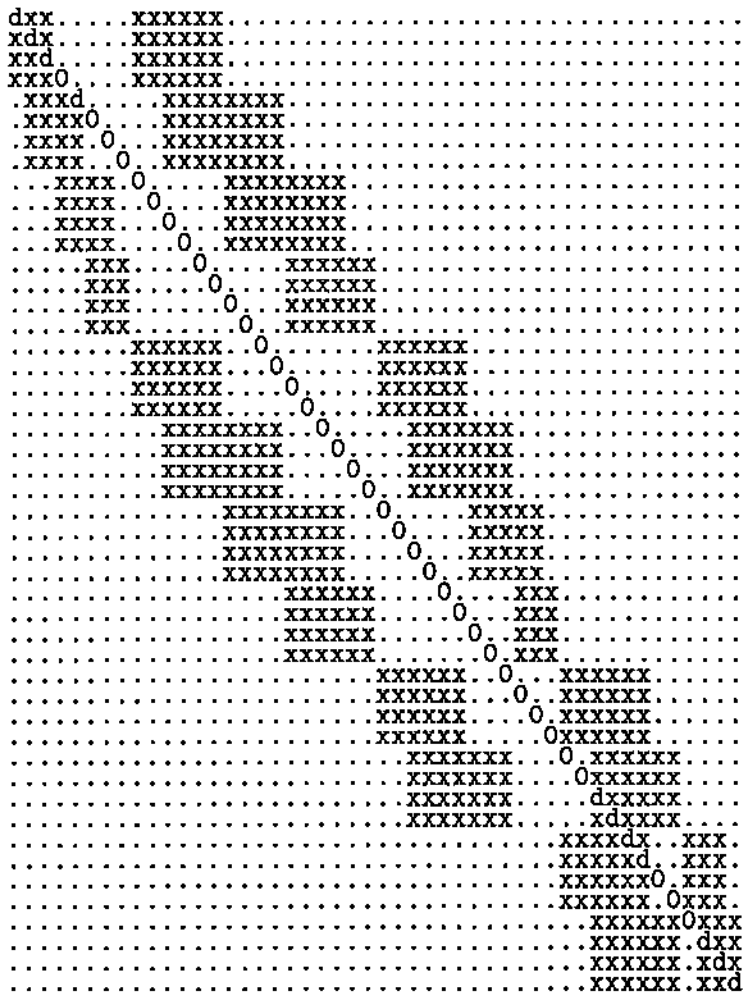


FIG. 5. *The detailed structure of the collocation matrix derived from GINCOL using the finite-element ordering for the example in Figure 1(a).*

method, we also display the final value of ω used; the initial guess of ω is given in the heading. In order to compare the efficiency among the various iterative solvers, we use the stopping criterion, namely, $\frac{\|x_{n+1}-x_n\|_2}{\|x_{n+1}\|_2} < \epsilon$ for SOR and $\frac{\|b-Ax_n\|_2}{\|b-Ax_0\|_2} < \epsilon$ for GMRES with the same initial solution $x_0 = [0.5, 0.5, \ldots, 0.5]^T$.

In the iterative computation, one wants the error in solving the linear system to be less than the discretization error in approximating the PDE. In all tables the convergence tolerance $\epsilon = 10^{-5}$ is used for SOR and $\epsilon = 10^{-6}$ for GMRES. As the data in Tables 1 and 2 indicate, this tolerance is too large as the discretization error on the coarsest mesh is already about 2×10^{-5} for the first example and even less for the second. Nevertheless, these data clearly show that all these iteration methods converge. For the non-adaptive SOR, the relaxation parameter ω is the optimal ω value corresponding to the case of the same problem defined on the smallest rectangle containing Ω. The AOR method used here is the one used in [8].

The fewest iterations by a factor 3 to 5 are required using the P_{II} preconditioning and an SOR iteration with relaxation parameter near the usual 1.8 value. The adaptive SOR can locate a "locally optimum" parameter less than 1 which provides performance

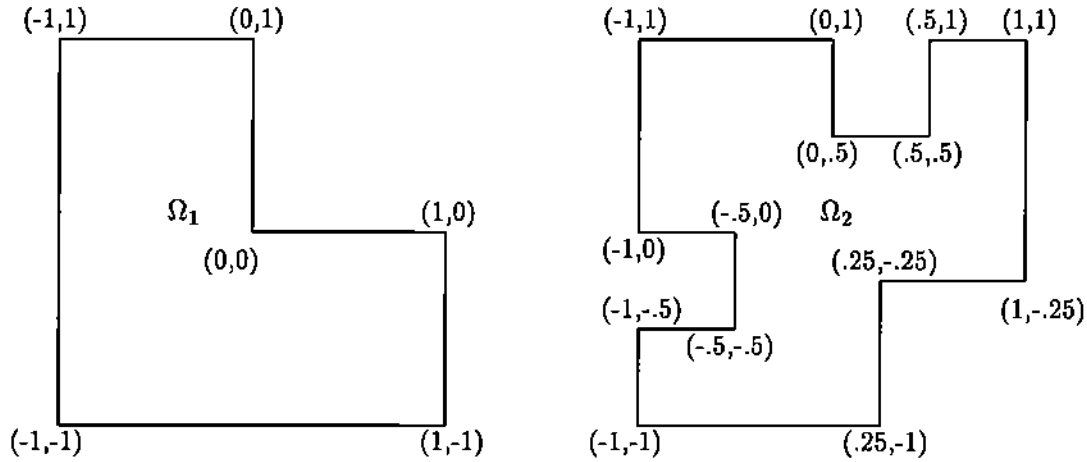


FIG. 6. *The domains used in the computational experiments.*

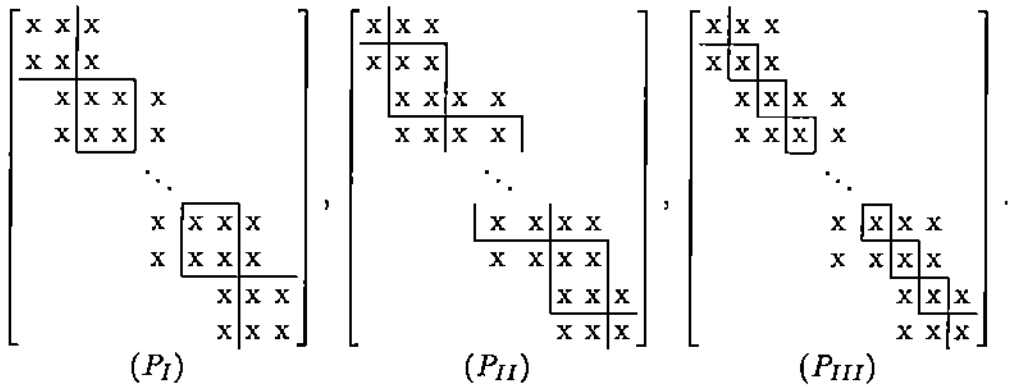


FIG. 7. *Three block partitionings of the GINCOL matrix. The diagonal block matrix outlined are used as preconditioners for the GMRES iteration.*

similar to that using the other preconditioners. These data suggest that this iteration approach has the promise to become an efficient and robust solver for the GINCOL collocation equations.

In Tables 4 to 6, we estimate the computational complexity of the GMRES and SOR iterative schemes for solving the GINCOL equations and compare them with BAND GE direct solver [11]. The data indicate that iterative solvers are more efficient for fine grids and produce solutions with the same level of discretization error. Furthermore, the convergence behavior of GMRES and SOR does not depend on the PDE operators considered in these experiments. For example, in the case of the SOR method the same ω values were used for a model problem and a general one. Finally, in Table 6, we apply the iterative schemes to solve the GENCOL equations using the tensor-product ordering. The PDE problem used here is defined on a rectangle, thus the optimal value of ω can be found in [7] for SOR. In this case we see that the iterative schemes are becoming more efficient than direct solvers even for coarse meshes.

Additional preliminary experiments indicate the GMRES is an efficient alternative to BAND GE for the solution of GENCOL equations with tensor-product ordering obtained from the discretization of PDE problems defined on general domains. All

TABLE 1

The convergence behavior of block iterative methods for solving the GINCOL linear system obtained by discretizing the equation $u_{xx} + u_{yy} = f$ in Ω_1 with Dirichlet boundary condition $(u = g)$. The functions f and g are selected so that $u(x, y) = e^{x+y}$.

mesh size (neqn)	P_I								P_{III}		
	AOR		*adaptive SOR*			*SOR*			*SOR*		
	iter	error	$\omega(1.0)$	iter	error	ω	iter	error	ω	iter	error
4 X 4 (48)	24	1.91e-5	0.8285	41	2.04e-5	0.7537	16	2.00e-5	0.5	24	2.52e-5
8 X 8 (192)	68	1.18e-5	0.8285	58	8.20e-6	0.7374	48	8.73e-6	0.5	60	2.29e-5
16 X 16 (768)	225	4.62e-5	0.8285	219	3.46e-6	0.7334	191	7.63e-6	0.5	242	3.30e-5

mesh size (neqn)	P_{II}								
	adaptive SOR			*adaptive SOR*			*SOR*		
	$\omega(1.0)$	iter	error	$\omega(1.01)$	iter	error	ω	iter	error
4 X 4 (48)	0.8285	41	2.04e-5	1.6880	14	2.13e-5	1.1786	9	2.06e-5
8 X 8 (192)	0.8285	58	8.20e-6	1.7998	28	7.75e-6	1.4271	19	1.43e-6
16 X 16 (768)	0.8285	219	3.46e-6	1.7070	60[1]	8.34e-6	1.6536	45	8.34e-7

[1] At this step the stopping criterion is not satisfied. The corresponding iteration error is 6.35e-5

TABLE 2

The convergence behavior of block iterative methods for solving the linear system obtained by discretizing the equation $u_{xx} + u_{yy} = f$ in Ω_2 with Dirichlet boundary condition $(u = g)$. The functions f and g are selected so that $u(x, y) = e^{x+y}$.

mesh size (neqn)	P_I								P_{III}		
	AOR		*adaptive SOR*			*SOR*			*SOR*		
	iter	error	$\omega(1.0)$	iter	error	ω	iter	error	ω	iter	error
8 X 8 (188)	49	1.22e-5	0.8284	40	3.52e-6	0.7374	36	2.86e-6	0.5	45	5.72e-6
16 X 16 (752)	179	1.35e-5	0.8285	156	1.97e-6	0.7334	138	5.42e-6	0.5	175	2.90e-5

mesh size (neqn)	P_{II}								
	adaptive SOR			*adaptive SOR*			*SOR*		
	$\omega(1.0)$	iter	error	$\omega(1.01)$	iter	error	ω	iter	error
8 X 8 (188)	0.8284	32	1.39e-5	1.7901	20	3.58e-6	1.4271	19	1.907e-6
16 X 16 (752)	0.8284	124	2.53e-5	1.6669	45	1.68e-6	1.6536	41	2.38e-6

results indicate that SOR is applicable to solve the GINCOL equations with tensor-product ordering, at least for rectilinear domains. The extension of GINCOL to general domains is part of our ongoing research efforts.

REFERENCES

[1] M. P. CARMO, *Differential Geometry of Curves and Surfaces*, Prentice-Hall, Englewood Cliffs, N. J., 1976.

[2] W. R. DYKSEN AND J. R. RICE , *A New Ordering Scheme for the Hermite Bicubic Collocation Equations* , in *Elliptic Solver II*, (G. Birkhoff and A. Schoenstat, eds), Academic Press, New York, 1984, pp. 467–480.

[3] W. R. DYKSEN AND J. R. RICE, *The Importance of Scaling for the Hermite Bicubic Collocation Equations*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 707–719.

[4] E. N. HOUSTIS, W. MITCHELL AND J. R. RICE, *Collocation Software for Second Order Elliptic Partial Differential Equations*, ACM Trans. Math. Software, 11 (1985), pp. 379–412.

TABLE 3

The convergence behavior of block iterative methods for solving the linear system obtained by discretizing the equation $u_{xx} + u_{yy} = f$ in Ω_2 with Dirichlet boundary condition ($u = g$). The functions f and g are selected so that $u(x,y) = 10\phi(x)\phi(y)$, where $\phi(x) = e^{-100(x-0.1)^2}(x^2 - x)$.

mesh size (neqn)	P_{II}								
	adaptive SOR			*adaptive SOR*			*SOR*		
	ω(1.0)	iter	error	ω(1.01)	iter	error	ω	iter	error
8 X 8 (188)	0.8284	30	7.89e-2	1.90	52	7.89e-2	1.4271	21	7.89e-2
16 X 16 (752)	0.8284	91	2.03e-2	1.9	69	2.03e-2	1.6536	44	2.03e-2
32 X 32 (3008)	0.8284	243	5.54e-4	1.8701	70	5.68e-4	1.8054	92	5.68e-4

TABLE 4

The performance data of some solvers for solving the discrete equations obtained by applying GINCOL algorithm to the equation $u_{xx} + u_{yy} = f$ with Dirichlet boundary conditions in domain Ω_2. The function f is selected so that $u(x,y) = 10\phi(x)\phi(y)$, where $\phi(x) = e^{-100(x-0.1)^2}(x^2 - x)$. BAND GE is applied with partial pivoting and "natural ordering" of the equations and unknowns. The iterative solver is used to solve the linear system using tensor-product ordering.

		BAND GE		GMRES(50)			SOR			
mesh	neqn	error	time	error	iter	time	error	iter	time	ω
8x8	188	7.89e-2	0.21	7.89e-2	16	0.40	7.89e-2	17	0.8	1.1786
16x16	752	2.03e-2	5.75	2.03e-2	51	6.23	2.03e-2	38	4.37	1.4271
32x32	3008	5.68e-4	22.65	5.61e-4	58	27.97	5.68e-4	71	28.63	1.6536
64x64	12032	3.03e-5	279.60	7.27e-5	122	242.97	3.09e-5	145	233.93	1.8054

[5] E. N. HOUSTIS, W. MITCHELL AND J. R. RICE, *Algorithm GENCOL: Collocation on General Domains with Bicubic Hermite Polynomials*, ACM Trans. Math. Software, 12 (1985), pp. 413–415.

[6] E. N. HOUSTIS, W. MITCHELL AND J. R. RICE, *Algorithms INTCOL and HERMCOL: Collocation on Rectangular Domains with Bicubic Hermite Polynomials*, ACM Trans. Math. Software, 12 (1985), pp. 416–418.

[7] Y.-L. LAI, A. HADJIDIMOS, E. N. HOUSTIS AND J. R. RICE, *On the Iterative Solution of Hermite Collocation Equations* , to appear in SIAM J. Matrix Anal. Appl., also Technical Report, CSD-TR-92-094, Computer Science Department, Purdue Univ., West Lafayette, IN, 1992.

[8] T. S. PAPATHEODOROU, *Block AOR Iteration for Non-symmetric Matrices*, Math. Comp., 41 (1983), pp. 511–525.

[9] P. M. PRENTER, *Splines and Variational Methods*, Wiley, New York, 1975.

[10] P. M. PRENTER AND R. D. RUSSELL, *Orthogonal Collocation for Elliptic Partial Differential Equations*, SIAM J. Numer. Anal., 3 (1976), pp. 923-939.

[11] J.R. RICE AND R.F. BOISVERT , *Solving Elliptic Problems Using ELLPACK*, Springer-Verlag, New York, 1985.

[12] Y. SAAD AND M. H. SCHULTZ, *A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM J. Sci. Stat. Comput., 6 (1985), pp. 865–869.

[13] O. ZIENKIEWICZ, *The Finite Element Method in Engineering Science*, McGraw-Hill, London, 1971.

TABLE 5

The performance data of some solvers for solving the discrete equations obtained by applying GINCOL algorithm to the equation $[2+(y-1)e^{-y^4}]u_{xx}+[1+\frac{1}{(1+4x^2)}]u_{yy}+5[x(x-1)+(y-0.3)(y-0.7)]u = f$ *with Dirichlet boundary conditions in domain* Ω_2. *The function* f *is selected so that* $u(x,y)=10\phi(x)\phi(y)$, *where* $\phi(x)=e^{-100(x-0.1)^2}(x^2-x)$. *BAND GE is applied with partial pivoting and "natural ordering" of the equations and unknowns. The iterative solver is used to solve the linear system using tensor-product ordering.*

	BAND GE		GMRES(50)			SOR			
mesh	error	time	error	iter	time	error	iter	time	ω
8x8	8.65e-2	0.23	8.65e-2	16	0.40	8.65e-2	17	0.8	1.1786
16x16	2.05e-2	2.12	2.05e-2	51	6.18	2.05e-2	39	4.22	1.4271
32x32	5.68e-4	21.57	5.45e-4	63	29.75	5.68e-4	72	28.87	1.6536
64x64	2.98e-5	266.83	9.05e-5	127	250.35	3.17e-5	150	240.70	1.8054

TABLE 6

The performance data of some solvers for solving the discrete equations obtained by applying GENCOL and GINCOL procedures to the equation $u_{xx}+u_{yy}=f$ *with Dirichlet boundary conditions on the rectangle* $(-1,1)\times(-1,1)$. *The function* f *is selected so that* $u(x,y)=10\phi(x)\phi(y)$, *where* $\phi(x)=e^{-100(x-0.1)^2}(x^2-x)$. *BAND GE is applied with partial pivoting and "natural ordering" of the equations and unknowns. The iterative solver is used to solve the linear system based using tensor-product ordering.*

GENCOL									
		BAND GE		GMRES(50)			Opt SOR		
mesh	neqn	error	time	error	iter	time	error	iter	time
2x2	36	2.99e-1	0.04	2.99e-1	8	0.05	2.99e-1	6	0.48
4x4	100	8.45e-1	0.39	8.45e-1	13	0.17	8.45e-1	10	0.93
8x8	324	1.34e-1	0.83	1.34e-1	36	4.30	1.34e-1	23	1.52
16x16	1156	2.33e-2	8.55	2.33e-2	53	9.883	2.33e-2	47	7.72
32x32	4356	5.68e-4	104.98	5.69e-4	73	49.95	5.69e-4	99	57.05
64x64	16900	2.91e-5	968.83	3.35e-5	191	589.633	3.09e-5	284	625.88

GINCOL									
		BAND GE		GMRES(50)			Opt SOR		
mesh	neqn	error	time	error	iter	time	error	iter	time
2x2	16	2.99e-1	0.02	2.99e-1	7	0.02	2.99e-1	6	0.48
4x4	64	8.45e-1	0.07	8.45e-1	12	0.08	8.45e-1	10	0.65
8x8	256	1.34e-1	0.40	1.34e-1	20	0.68	1.34e-1	22	1.15
16x16	1024	2.33e-2	8.88	2.33e-2	51	19.7	2.33e-2	43	6.30
32x32	4096	5.68e-4	41.68	5.86e-4	66	54.417	5.69e-4	92	49.3
64x64	16384	2.91e-5	648.75	5.98e-5	186	498.35	3.09e-5	246	540.93