A PRECONDITIONED CONJUGATE GRADIENT METHOD USING A SPARSE LINKED-LIST TECHNIQUE FOR THE SOLUTION OF FIELD PROBLEMS

by

RONALD L. NAKONECHNY

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
MASTER OF SCIENCE
in
ELECTRICAL ENGINEERING

Winnipeg, Manitoba, 1983

A

PRECONDITIONED CONJUGATE GRADIENT METHOD USING A SPARSE LINKED-LIST TECHNIQUE FOR THE SOLUTION OF FIELD PROBLEMS

BY

RONALD L. NAKONECHNY

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

RONALD L. NAKONECHNY

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

RONALD L. NAKONECHNY

ABSTRACT

This thesis presents a method for the solution of large, sparse, symmetric sets of linear equations employing the Incomplete Cholesky Conjugate Gradient (ICCG) method.

The incomplete Cholesky preconditioning technique is presented and modified to be compatible with a sparse Zollenkopf linked-list storage scheme. A windowed preconditioning technique was employed that resulted in a substantial reduction of the overall computational time.

A modification to the original algorithm makes it suitable for handling Dirichlet boundary conditions arising from a finite element discretization of electric field problems. Also, the preconditioned conjugate gradient method was able to obtain a solution to the pure Neumann field problem (a singular matrix problem).

ACKNOWLEDGEMENTS

The author would like to thank Dr. A. Wexler for his supervision, guidance and enthusiastic encouragement during the development of this work.

The author also wishes to thank his colleagues of the Numerical Applications Group at the University of Manitoba, particularly Mr. R. Allen, Mr. P. Giese and Mr. J. Shaw for their discussions related to this thesis topic. The author also appreciates the contributions and feedback received from Mr. B. Fry and Mr. B. Klimpke who implemented the sparse preconditioned conjugate gradient method into their own projects.

Finally, the author would like to thank his wife, Christine, whose encouragement and support aided in the completion of this thesis.

CONTENTS

LIST OF FIGURES

LIST OF PRINCIPAL SYMBOLS

$\underline{a}$	generic vector
$\underline{b}$	source vector of length N
BW	matrix bandwidth
D	diagonal matrix of order N
$\underline{e}$	error vector of length N
f	generic scalar source function
F	energy functional
g(s)	Dirichlet source
h	distance between finite difference nodes
H	large constant
$\underline{h}$	exact solution to system of equations
h(s)	Neumann source
K	generic matrix of order N
L	preconditioned matrix of order N
$\tilde{L}$	inverse factored matrix
M	generic matrix of order N
m(s)	mixed boundary source
N	number of unknowns
$\hat{n}$	unit normal
$\underline{p}$	direction vector of length N
q	variable window factor
R	domain
$\underline{r}$	residual vector of length N

$\bar{r}$	position vector
s	domain boundary
u	generic function or field
$\underline{u}$	generic vector
v	generic function or field
$\underline{v}$	generic vector
$\underline{x}$	generic vector
$\underline{y}$	generic vector
α	optimum scaling factor
$\alpha(\bar{r})$	interpolation functions
β	orthogonalization constant
ε	small constant
$\underline{\varepsilon}$	small vector quantity, approaching the zero vector
κ	medium characteristic
λ	eigenvalue
σ	conductivity
$\underline{\phi}$	solution vector of length N
Ψ	error functional
Ω	region of integration

Chapter I

INTRODUCTION

Numerical discretization of field problems using finite element or finite difference methods yields a system of simultaneous linear equations. The set of equations are usually written in matrix form as $S\underline{\phi}=\underline{b}$, where S is the system matrix of order N, $\underline{\phi}$ is a vector of N unknowns and $\underline{b}$ is the forcing vector of N terms.

The matrices which generally result from the use of a Finite Element Method (FEM) or Finite Difference Method (FDM) on field problems are usually:

1. symmetric, positive definite;
2. sparse; and
3. large (typically 500 to 10,000 Unknowns).

Iterative solution techniques (e.g. successive over-relaxation), though storage conscious, are often "subjective" in that they may require a few iterations before an optimum acceleration parameter for acceptable solution times can be determined. This thesis focuses on the Conjugate Gradient (CG) method, a hybrid direct-iterative process used to solve systems of linear equations. The CG method, discussed in Chapter II, was first developed by Hestenes and Stiefel [1] in 1952. The method, however, did not gain popularity due

to its excessive storage requirements as well as the limited storage resources of earlier computers. The method was recently recognized by Reid [2] as an effective equation solver for large sparse sets of linear equations. Chapter III presents a modification of the CG algorithm, known as preconditioning, which increases the efficiency of the CG method particularily in the case of ill-conditioned systems.

Chapter IV is a presentation of the sparsity technique that is adapted to conjugate gradients. Much of the current literature does not dwell on conjugate gradient sparsity techniques. This thesis presents a sparsity method, an adaptation of a Zollenkopf storage scheme, that appears to be ideal. Moveover, this thesis presents a modified sparse preconditioning technique which calculates no fill-ins. Storage techniques and system matrix topologies, with respect to solution convergence, are also discussed.

Chapters V and VI involve the FEM and FDM techniques used for modelling field problems and their resulting solutions with the preconditioned CG method. Some applications of the method to non-symmetric systems of linear equations are discussed. Benefits arising from the application of the preconditioned CG method to nonlinear iterative problems are revealed. A comparison with another sparse solution technique known as the Zollenkopf bi-factorisation method is presented in Chapter VI.

Chapter II

THE CONJUGATE GRADIENT METHOD

The conjugate gradient algorithm is formulated in the functional sense involving the minimization of a residual.[1] A direction vector is specified and its magnitude is scaled such that an error functional is minimized.

The solution is sought to the system of linear equations represented as

$$S\underline{\phi}=\underline{b} \tag{2.1}$$

where S is a square matrix of order N; $\underline{\phi}$ is the unknown vector of length N; and $\underline{b}$ is a vector of N coefficients. An iterative process of degree one [3] is defined as

$$\underline{\phi}^{(k+1)}=\underline{\phi}^{(k)}+\alpha_k\underline{p}^{(k)} \tag{2.2}$$

where α_k is a constant and $\underline{p}$ is a vector of length N. The iterative process has converged when

$$\alpha_k\underline{p}^{(k)}=\underline{0} \tag{2.3}$$

(In a practical computation, the right hand side of (2.3) would be a small, but arbitrary vector, $\underline{\varepsilon}$.) This implies

---

[1] The derivation of the method is purposely meant to be extensive since discussions on the subject of conjugate gradients are scattered throughout much of the literature.

that

$$\underline{\phi}^{(k)}=S^{-1}\underline{b} \tag{2.4}$$

The kth residual vector is defined as

$$\underline{r}^{(k)}=\underline{b}-S\underline{\phi}^{(k)} \tag{2.5}$$

Let $\underline{h}$ be the exact solution to (2.1), i.e.

$$\underline{h}=S^{-1}\underline{b} \tag{2.6}$$

The kth error vector is

$$\underline{e}^{(k)}=\underline{h}-\underline{\phi}^{(k)} \tag{2.7}$$

or

$$\underline{\phi}^{(k)}=\underline{h}-\underline{e}^{(k)} \tag{2.8}$$

Substituting (2.8) into (2.5)

$$\underline{r}^{(k)}=\underline{b}-S(\underline{h}-\underline{e}^{(k)}) \tag{2.9}$$

$$\underline{r}^{(k)}=\underline{b}-S\underline{h}+S\underline{e}^{(k)} \tag{2.10}$$

$$\underline{r}^{(k)}=\underline{b}-\underline{b}+S\underline{e}^{(k)} \tag{2.11}$$

$$\underline{r}^{(k)}=S\underline{e}^{(k)} \tag{2.12}$$

and

$$\underline{e}^{(k)}=S^{-1}\underline{r}^{(k)} \tag{2.13}$$

At this point an error measure is defined so that it may be minimized. Define

$$\Psi(\underline{\phi})=<\underline{e},K\underline{e}> \tag{2.14}$$

where K is an arbitrary N by N matrix. From (2.13)

$$\Psi(\underline{\phi})=<S^{-1}\underline{r},KS^{-1}\underline{r}> \tag{2.15}$$

Equation (2.14) is analagous to the square of the length of the error vector. If K is chosen to be a positive definite matrix, the quantity will be non-negative [3]. In this case, $\underline{h}$, the exact solution, will be the minimum point of Ψ. Expanding (2.15)

$$\Psi(\underline{\phi})=\underline{r}^{T}(S^{-1})^{T}K^{T}S^{-1}\underline{r} \tag{2.16}$$

A choice for K to simplify (2.16) is K = S. Equation (2.16) reduces to

$$\Psi(\underline{\phi})=\underline{r}^{T}(S^{-1})^{T}\underline{r} \tag{2.17}$$

and more conveniently

$$\Psi(\underline{\phi})=<\underline{r},S^{-1}\underline{r}> \tag{2.18}$$

At $\underline{\phi} = \underline{h}$, the gradient of $\Psi(\underline{\phi}) = 0$, i.e. the zero vector. The negative gradient of Ψ is the direction of steepest

descent at $\underline{\phi}^{(k)}$. Taking the gradient of (2.18) (Appendix A)

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = \frac{d(S^{-1}\underline{r})^T}{d\underline{\phi}}\underline{r} + \frac{d\underline{r}^T}{d\underline{\phi}}S^{-1}\underline{r} \tag{2.19}$$

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = \frac{d(\underline{r}^T)}{d\underline{\phi}}(S^{-1})^T\underline{r} + \frac{d(\underline{r}^T)}{d\underline{\phi}}S^{-1}\underline{r} \tag{2.20}$$

and since $(S^{-1})^T = S^{-1}$

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = 2\frac{d\underline{r}^T}{d\underline{\phi}}S^{-1}\underline{r} \tag{2.21}$$

Substituting (2.5) into (2.21) for $\underline{r}$ gives

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = 2\frac{d}{d\underline{\phi}}\{\underline{b}^T - \underline{\phi}^T S^T\}S^{-1}\underline{r} \tag{2.22}$$

(See Appendix A). Equation (2.22) becomes

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = -2S^T S^{-1}\underline{r} \tag{2.23}$$

which reduces to

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = -2\underline{r} \tag{2.24}$$

When $\underline{\phi} = \underline{h}$, the exact solution, (2.24) reduces to

$$\frac{d\Psi(\underline{\phi})}{d\underline{\phi}} = \underline{0}\Big|_{\underline{\phi}=\underline{h}} \tag{2.25}$$

or

$$-2\underline{r} = \underline{0} \Big|_{\underline{\phi}=\underline{h}} \tag{2.26}$$

which implies that $\underline{r} = \underline{0}$ at $\underline{\phi} = \underline{h}$. In the definition of an iterative process (2.2), α was chosen to be a constant. The next step is to minimize the error measure at $\underline{\phi}=\underline{\phi}+\alpha\underline{p}$, i.e. $\Psi(\underline{\phi}) = \Psi(\underline{\phi}+\alpha\underline{p})$. This may be done by taking the gradient of $\Psi(\underline{\phi}+\alpha\underline{p})$ with respect to α , and equating the gradient to zero to find the optimum α to minimize $\Psi(\underline{\phi}+\alpha\underline{p})$.

$$\Psi(\underline{\phi})=<\underline{r},S^{-1}\underline{r}> \tag{2.27}$$

Substituting (2.5) into (2.27) results in

$$\Psi(\underline{\phi})=<(\underline{b}-S\underline{\phi}),S^{-1}(\underline{b}-S\underline{\phi})> \tag{2.28}$$

Replacing $\underline{\phi}$ with $\underline{\phi}+\alpha\underline{p}$ in (2.28) gives

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{b}-S(\underline{\phi}+\alpha\underline{p}),S^{-1}(\underline{b}-S(\underline{\phi}+\alpha\underline{p}))> \tag{2.29}$$

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{b}-S\underline{\phi}-\alpha S\underline{p},S^{-1}(\underline{b}-S\underline{\phi}-\alpha S\underline{p})> \tag{2.30}$$

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{r}-\alpha S\underline{p},S^{-1}(\underline{r}-\alpha S\underline{p})> \tag{2.31}$$

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{r},S^{-1}\underline{r}>-<\underline{r},\alpha\underline{p}>-<\alpha S\underline{p},S^{-1}\underline{r}>+<\alpha S\underline{p},\alpha\underline{p}> \tag{2.32}$$

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{r},S^{-1}\underline{r}>-\alpha<\underline{r},\underline{p}>-\alpha<S\underline{p},S^{-1}\underline{r}>+\alpha^2<S\underline{p},\underline{p}> \tag{2.33}$$

Expanding the third term of (2.33)

$$<S\underline{p},S^{-1}\underline{r}>=\underline{r}^T(S^{-1})^TS\underline{p} \tag{2.34}$$

$$<S\underline{p},S^{-1}\underline{r}>=\underline{r}^TS^{-1}S\underline{p} \tag{2.35}$$

$$<S\underline{p},S^{-1}\underline{r}>=\underline{r}^TI\underline{p} \tag{2.36}$$

$$<S\underline{p},S^{-1}\underline{r}>=<\underline{p},\underline{r}> \tag{2.37}$$

and from Appendix A,

$$<S\underline{p},S^{-1}\underline{r}>=<\underline{r},\underline{p}> \tag{2.38}$$

Substituting (2.38) into (2.33) gives

$$\Psi(\underline{\phi}+\alpha\underline{p})=<\underline{r},S^{-1}\underline{r}>-2\alpha<\underline{r},\underline{p}>+\alpha^2<S\underline{p},\underline{p}> \tag{2.39}$$

Ψ is minimized when α is optimized. Taking the derivative of $\Psi(\underline{\phi}+\alpha\underline{p})$ with respect to α ,

$$\frac{d\Psi(\underline{\phi}+\alpha\underline{p})}{d\alpha} = 0 = -2<\underline{r},\underline{p}>+2\alpha<S\underline{p},\underline{p}> \tag{2.40}$$

and

$$\alpha = \frac{\langle \underline{r}, \underline{p} \rangle}{\langle \underline{p}, S\underline{p} \rangle} \tag{2.41}$$

For the kth iteration, (2.41) becomes

$$\alpha_k = \frac{\langle \underline{r}^{(k)}, \underline{p}^{(k)} \rangle}{\langle \underline{p}^{(k)}, S\underline{p}^{(k)} \rangle} \tag{2.42}$$

The vectors that remain to be derived are the $\underline{r}$ (residual) and p (direction) vectors.

The conjugate gradient method relies on the choice of the direction vectors being S-orthogonal,[2]

$$\langle \underline{p}^{(i)}, S\underline{p}^{(j)} \rangle = 0 \quad , i \neq j \tag{2.43}$$

while the residual vectors remain orthogonal, i.e.

$$\langle \underline{r}^{(i)}, \underline{r}^{(j)} \rangle = 0 \ , \ i \neq j \tag{2.44}$$

Since S is positive definite, then $\langle \underline{p}^{(i)}, S\underline{p}^{(j)} \rangle > 0$ for $\underline{p} \neq \underline{0}$. As a result, the direction vectors, $\underline{p}^{(i)}$ are linearly independent and the solution, $\underline{h}$, may be expressed as [3,45]

$$\underline{h} = \sum_{i=0}^{N-1} \alpha_i \underline{p}^{(i)} \tag{2.45}$$

Choosing $\underline{p}$ such that it is S-orthogonal (2.43) and $\underline{r}$ such that it is orthogonal (2.44) places a constraint on the number of iterations required for the CG method to converge. Since the $\underline{r}$ and $\underline{p}$ vectors enforce two orthogonality conditions simultaneously, it implies that they must both be zero

---

[2] Commonly referred to as "A-orthogonal" [3].

at the Nth iteration. Direction vectors constructed after the Nth iteration cannot be linearly independent since the system of equations is only of dimension N. Similarly, the residual vectors are forced to be orthogonal and therefore, linearly independent. The vector $\underline{r}^{(N)}$ would be orthogonal to all the N previous residual vectors ($\underline{r}^{(0)}, \underline{r}^{(1)}, \ldots, \underline{r}^{(N-1)}$) and therefore, $\underline{r}^{(N)}$ is forced to be the zero vector. The solution is obtained when $\underline{r}^{(N)} = \underline{0}$.

The solution of (2.1) is obtained in at most N iterations due to the orthogonality relationships incorporated into the algorithm. The algorithm is therefore a hybrid direct-iterative process, meaning that the solution is attained in a finite number of iterations like that of a direct solution technique provided that no round-off errors occur [3].
The new residual, $\underline{r}^{(k+1)}$ may be derived as follows:
The kth residual, from (2.5) is

$$\underline{r}^{(k)} = \underline{b} - S\underline{\phi}^{(k)} \tag{2.46}$$

and

$$\underline{r}^{(k+1)} = \underline{b} - S\underline{\phi}^{(k+1)} \tag{2.47}$$

Subtracting (2.46) from (2.47) results in

$$\underline{r}^{(k+1)} - \underline{r}^{(k)} = -\alpha_k S\underline{p}^{(k)} \tag{2.48}$$

and

$$\underline{r}^{(k+1)} = \underline{r}^{(k)} - \alpha_k S\underline{p}^{(k)} \tag{2.49}$$

The new direction vector, $\underline{p}^{(k+1)}$, is formed by the S-orthogonalization of the residual vectors utilizing a successive othogonalization process. The following orthogonalization process is shown by [4,444] to be identical to the Gram-Schmidt orthogonalization process. The direction vectors have the form:

$$\underline{p}^{(0)} = \underline{r}^{(0)} \tag{2.50}$$

$$\underline{p}^{(1)} = \underline{r}^{(1)} + \beta_0 \underline{p}^{(1)} \tag{2.51}$$

$$\underline{p}^{(2)} = \underline{r}^{(2)} + \beta_1 \underline{p}^{(1)} \tag{2.52}$$

. .

. .

. .

$$\underline{p}^{(k)} = \underline{r}^{(k)} + \beta_{k-1} \underline{p}^{(k-1)} \tag{2.53}$$

$$\underline{p}^{(k+1)} = \underline{r}^{(k+1)} + \beta_k \underline{p}^{(k)} \tag{2.54}$$

The p vectors must be S-orthogonal, i.e. $\langle \underline{p}^i, S\underline{p}^j \rangle = 0$, $i \neq j$, while the residual vectors must remain orthogonal, $\langle \underline{r}^i, \underline{r}^j \rangle = 0$, $i \neq j$. The $\underline{p}$ vectors represent (as nearly as possible) the directions of steepest descent of the error functional (2.18) subject to the over-riding condition that they are orthogonal with respect to S [5,142]. The constant, β_k,

is determined such that $\underline{p}^{(k)}$ is S-orthogonal with $\underline{p}^{(k+1)}$. For the kth direction vector

$$<\underline{p}^{(k)}, S\underline{p}^{(k+1)}> = 0 \tag{2.55}$$

From (2.54)

$$<\underline{p}^{(k)}, S\underline{r}^{(k+1)} + \beta_k S\underline{p}^{(k)}> = 0 \tag{2.56}$$

$$<\underline{p}^{(k)}, S\underline{r}^{(k+1)}> + \beta_k <\underline{p}^{(k)}, S\underline{p}^{(k)}> = 0 \tag{2.57}$$

$$\beta_k = - \frac{<\underline{p}^{(k)}, S\underline{r}^{(k+1)}>}{<\underline{p}^{(k)}, S\underline{p}^{(k)}>} \tag{2.58}$$

which is equivalent to

$$\beta_k = - \frac{<S\underline{p}^{(k)}, \underline{r}^{(k+1)}>}{<\underline{p}^{(k)}, S\underline{p}^{(k)}>} \tag{2.59}$$

Substituting (2.48) into (2.59) gives

$$\beta_k = - \frac{<\dfrac{\underline{r}^{(k)} - \underline{r}^{(k+1)}}{\alpha_k}, \underline{r}^{(k+1)}>}{<\underline{p}^{(k)}, S\underline{p}^{(k)}>} \tag{2.60}$$

$$\beta_k = \frac{1}{\alpha_k} \frac{<\underline{r}^{(k+1)} - \underline{r}^{(k)}, \underline{r}^{(k+1)}>}{<\underline{p}^{(k)}, S\underline{p}^{(k)}>} \tag{2.61}$$

$$\beta_k = \frac{1}{\alpha_k} \frac{\langle \underline{r}^{(k+1)}, \underline{r}^{(k+1)} \rangle - \langle \underline{r}^{(k)}, \underline{r}^{(k+1)} \rangle}{\langle \underline{p}^{(k)}, S\underline{p}^{(k)} \rangle} \tag{2.62}$$

and noting that the second term in the numerator of (2.62) is zero

$$\beta_k = \frac{1}{\alpha_k} \frac{\langle \underline{r}^{(k+1)}, \underline{r}^{(k+1)} \rangle}{\langle \underline{p}^{(k)}, S\underline{p}^{(k)} \rangle} \tag{2.63}$$

In order to obtain a form of the algorithm which minimizes the storage requirements for digital computer calculation, an additional orthogonality relationship must be obtained. The numerator of (2.42)

$$\langle \underline{r}^{(k)}, \underline{p}^{(k)} \rangle \tag{2.64}$$

is sought to be expanded. From (2.53)

$$\underline{p}^{(k)} = \underline{r}^{(k)} + \beta_{k-1} \underline{p}^{(k-1)} \tag{2.65}$$

Substituting (2.65) into (2.64) results in

$$\langle \underline{r}^{(k)}, \underline{p}^{(k)} \rangle = \langle \underline{r}^{(k)}, \underline{r}^{(k)} + \beta_{k-1} \underline{p}^{(k-1)} \rangle \tag{2.66}$$

$$\langle \underline{r}^{(k)}, \underline{p}^{(k)} \rangle = \langle \underline{r}^{(k)}, \underline{r}^{(k)} \rangle + \beta_{k-1} \langle \underline{r}^{(k)}, \underline{p}^{(k-1)} \rangle \tag{2.67}$$

The second term of (2.67) is zero since $\underline{p}^{(k-1)}$ is not a function of $\underline{r}^{(k)}$ and therefore all inner products, $\langle \underline{r}^{i}, \underline{r}^{j} \rangle$ with $j \neq i$, will vanish. Therefore

$$\langle \underline{r}^{(k)}, \underline{p}^{(k)} \rangle = \langle \underline{r}^{(k)}, \underline{r}^{(k)} \rangle \tag{2.68}$$

Utilizing (2.68), (2.42) may be rewritten as

$$\alpha_k = \frac{\langle \underline{r}^{(k)}, \underline{r}^{(k)} \rangle}{\langle \underline{p}^{(k)}, S\underline{p}^{(k)} \rangle} \tag{2.69}$$

Thus, substituting this result into (2.62) gives

$$\beta_k = \frac{\langle \underline{r}^{(k+1)}, \underline{r}^{(k+1)} \rangle}{\langle \underline{r}^{(k)}, \underline{r}^{(k)} \rangle} \tag{2.70}$$

Finally, the CG algorithm may be summed up by equations (2.2), (2.49), (2.54), (2.69) and (2.70) with $\underline{p}^{(0)} = \underline{r}^{(0)}$ initially,

$$\alpha_i = \frac{\langle \underline{r}^{(i)}, \underline{r}^{(i)} \rangle}{\langle \underline{p}^{(i)}, S\underline{p}^{(i)} \rangle} \tag{2.71}$$

$$\underline{\phi}^{(i+1)} = \underline{\phi}^{(i)} + \alpha_i \underline{p}^{(i)} \tag{2.72}$$

$$\underline{r}^{(i+1)} = \underline{r}^{(i)} - \alpha_i S\underline{p}^{(i)} \tag{2.73}$$

$$\beta_i = \frac{\langle \underline{r}^{(i+1)}, \underline{r}^{(i)} \rangle}{\langle \underline{r}^{(i)}, \underline{r}^{(i)} \rangle} \tag{2.74}$$

$$\underline{p}^{(i+1)} = \underline{r}^{(i+1)} + \beta_i \underline{p}^{(i)} \tag{2.75}$$

$$i=0,1,2,\ldots$$

The iteration may be terminated when the residual vector is essentially the zero vector.

The conjugate gradient method can obtain the solution to a system of N simultaneous linear equations, $S\underline{\phi}=\underline{b}$ (where S

is symmetric and positive definite), in at most N iterations if no rounding errors exist. In practice, for large sparse matrices, the solution is obtained in less than the N required amount of iterations. In this case, it is desirable to check the residual vector to determine whether or not it is less than a specified criterion, rather than iterating to the Nth step.

Chapter III

THE PRECONDITIONED CONJUGATE GRADIENT METHOD

In the course of investigation, the CG method has exhibited a slow convergence rate for some types of problems, although some reports indicate that the method faired well in the solution of some ill-conditioned finite difference problems [6]. Kershaw [7,46] attributes the slow behavior to problems where the system matrix has no multiple eigenvalues and contains an even spread of eigenvalues between the maximum and minimum eigenvalue. Jennings [8,218] also states that slow convergence corresponds to a large number of small but distinct eigenvalues. However, if the system matrix contains only $s<N$ distinct eigenvalues (N being the order of the S-matrix), then the conjugate gradient method obtains its solution in s-iterations [9,662].

In order to combat the slow convergence problem, one method (as suggested by Irons [9,661]) is to precondition or pre-multiply equation (2.1) on both sides by a matrix of the same order as S. The preconditioned matrix has the characteristic of approximating the inverse of the S-matrix which may (hopefully) better condition the orginal system of equations [10,157]. It is desirable not to spend much computational time calculating the preconditioned matrix but rather

to find an economical approximation to the inverse of the S-matrix.

3.1 PRECONDITIONING WITH AN INCOMPLETE CHOLESKY DECOMPOSITION

A familiar process known as the Cholesky decomposition method decomposes a symmetric S-matrix into the product of an upper and lower triangular matrix. In order to save time, an incomplete decomposition is perfomed such that the decomposition is approximate, i.e. the resulting inverse of the triangular matrices is approximate.[3] The incomplete decomposition has benefits both in hastening convergence of the system and complementing the sparse storage scheme of Chapter IV. The idea behind an incomplete decomposition is as follows:

1. Calculate an approximate inverse S-matrix by performing an incomplete Cholesky decomposition.
2. Pre-multiply (2.1) by this approximate inverse to the system matrix and solve the resulting system using the CG algorithm.
3. The resulting system will be "preconditioned", meaning that convergence to the solution will depend on the effectiveness of the approximate inverse matrix. Preconditioning has the effect of introducing multiple eigenvalues which speed up the convergence of the

---

[3] Axelsson [11] states that an incomplete Cholesky factorization is numerically stable.

CG process. To illustrate, if one preconditoned by the exact inverse, the resulting system eigenvalues would all be 1, i.e. convergence would be attained (trivially) in one iteration.

The Cholesky decomposition approach is used because it is one of the most efficient methods for solving a symmetric system of linear equations [12]. The symmetric S-matrix is decomposed to

$$S = LL^T \tag{3.1}$$

where L is lower triangular. Therefore, the solution to the problem may be written as

$$\underline{\phi} = (L^T)^{-1}L^{-1}\underline{b} \tag{3.2}$$

In order to avoid a complete Cholesky decompositon, the sparsity pattern for L is chosen to be the same as that for S [7]. In other words, if an S(i,j) is non-zero, then and only then, is a corresponding L(i,j) calculated. Therefore, an L matrix would only require the same amount of storage as the S-matrix (the S-matrix is stored in lower triangular form since S is symmetric). A major problem with sparse matrix techniques is that one must determine the amount of storage needed to hold the factorized form [13]. Since this particular method has no fill-ins, its storage requirements can be easily determined. The decomposition may be explic-

itly written as

$$S = LL^T + E \tag{3.3}$$

where E is an error matrix corresponding to the 'missing' entries of L which are exempted from calculation. Performing this form of decomposition may be done efficiently without requiring any significant amounts of storage since no fill-ins are calculated or stored. The incomplete decomposition may be interpreted as coupling a given region within the S-matrix most strongly to its nearest neighbors as an approximation to the inverse matrix of the system, namely S^{-1}.

Assuming that

$$S \simeq LL^T \tag{3.4}$$

$S\underline{\phi}=\underline{b}$ may be rewritten as

$$[L^{-1}S(L^T)^{-1}](L^T\underline{\phi}) = L^{-1}\underline{b} \tag{3.5}$$

Viewing the new system matrix as

$$S' = L^{-1}S(L^T)^{-1} \tag{3.6}$$

S' may be viewed as an approximate identity matrix which should make the CG convergence more rapid. Substituting (3.6) into equations (2.29) to (2.33) of the previous chapter with $\underline{r}^{(0)} = \underline{b} - S\underline{\phi}$ and $\underline{p}^{(0)} = K\underline{r}^{(0)}$ [1], results in [7]:

$$\alpha_i = \frac{\langle \underline{r}^{(i)}, K\underline{r}^{(i)} \rangle}{\langle \underline{p}^{(i)}, S\underline{p}^{(i)} \rangle} \tag{3.7}$$

$$\underline{\phi}^{(i+1)} = \underline{\phi}^{(i)} + \alpha_i \underline{p}^{(i)} \tag{3.8}$$

$$\underline{r}^{(i+1)} = \underline{r}^{(i)} - \alpha_i S\underline{p}^{(i)} \tag{3.9}$$

$$\beta_i = \frac{<\underline{r}^{(i+1)}, K\underline{r}^{(i+1)}>}{<\underline{r}^{(i)}, K\underline{r}^{(i)}>} \tag{3.10}$$

$$\underline{p}^{(i+1)} = K\underline{r}^{(i+1)} + \beta_i \underline{p}^{(i)} \tag{3.11}$$

where the matrix

$$K = (LL^T)^{-1} \tag{3.12}$$

In order to avoid the possiblity of negative square roots which may appear in the LL^T decomposition [7,46], an LDL^T decomposition may be performed, i.e.

$$L_{ji} = S_{ji} - \sum_{k=1}^{(i-1)} L_{jk} L_{ik} D_{kk} \tag{3.13}$$

$$j=i, i+1, \ldots, N$$

$$D_{ii} = \frac{1}{L_{ii}} \tag{3.14}$$

Therefore, K in equations (3.7) to (3.12) may be replaced with

$$K = (LDL^T)^{-1} \tag{3.15}$$

The preconditioned matrix has the same sparsity as the S ma-

trix, but the inverse of the preconditioned matrix is actually what is required for (3.15). In order to obtain $(LDL^T)^{-1}$, the L-matrix is treated as (N-1) stacked factor matrices (Appendix B), and the inverse is the the product of these factor matrices [14,22-24]. The stacked matrix form is convenient for the sparsity technique discussed in Chapter IV. It is important to note, however, that since the L-matrix is calculated with the omission of fill-ins, the product of all the factor matrices (i.e. the inverse) will be approximate.

In testing the preconditioned version of the CG algorithm, an improvement in the rate of convergence has been observed. Typically, the solution will converge within $\sqrt{N}$ CG iterations, (as tested for problem sizes ranging up to 10,000 unknowns), as opposed to nearly N with the CG method in its simplest form. The original CG algorithm is satisfactory for problems in which the number of unknowns does not exceed some interim value such as N=200 [15].

Chapter IV

CONJUGATE GRADIENT SPARSITY TECHNIQUE

One of the main points of this thesis involves the application of a conjugate gradient (symmetric) linear equation solution technique to solve large, sparse sets of linear equations.

4.1 A MODIFIED ZOLLENKOPF STORAGE SCHEME

To complement the conjugate gradient process, a Zollenkopf storage scheme was chosen for the following reasons:

1. The ability to randomly access and construct a system matrix without storing zero coefficients;
2. the scheme structure allows it to be efficiently modified for sparse symmetric storage;
3. the same storage scheme may be used to access the elements of both the preconditioned L and S-matrices. This feature provides an extremely economical (and efficient) method of storage for the largest of matrices incurred; and
4. computational efficiency is increased by performing only non-trivial arithmetic operations, e.g. no multiplications by zero are performed.

The Zollenkopf sparse storage method, based on a linked-list scheme, explained thoroughly by Wexler [14,45-50], is outlined in the example below. The Zollenkopf linked-lists are modified to store the symmetric matrix of Fig. 1 in a lower triangular form (Fig. 2), thus reducing the amount of storage from N^2 to $N(N+1)/2$.

$$\begin{bmatrix} 4 & 0 & 0 & -2 \\ 0 & 2 & -1 & -1 \\ 0 & -1 & 3 & 0 \\ -2 & -1 & 0 & 2 \end{bmatrix}$$

Figure 1: An example of a typical symmetric matrix.

INDEX	CE	ITAG	LNXT	LCOL
1	4	1	5	1
2	2	2	3	2
3	-1	3	6	4
4	3	3	0	7
5	-2	4	0	
6	-1	4	0	
7	2	4	0	

Figure 2: The example matrix stored (lower triangularly) in a linked-list.

The vector CE contains the matrix of coefficients stored columnwise; the parallel vector ITAG gives the row number of the associated entry; the parallel vector LNXT gives the location of the next position of the next column entry; and

the vector LCOL (of length N) contains the location of the start of each column. An LNXT(i)=0 signifies that no more entries exist in that column.

The above vectors contain the relevant information to reconstruct the above matrix without the storage of non-zero matrix terms. Note too, that the ITAG, LNXT and LCOL vectors are integer vectors, which require half the storage of a floating point vector with the same length. When the number of unknowns becomes large, the matrix is sparsely populated with non-zero coefficients and the amount of storage required by the auxiliary vectors ITAG, LNXT and LCOL would be small by comparison. Note that since the matrix is symmetric, only the lower triangular portion of the matrix is actually stored, thus reducing the amount of sparse storage by approximately half.

The Zollenkopf storage scheme is quite flexible if an additional entry is suddenly introduced. For example, consider the element S(2,1) being replaced by a -1.1. The result is depicted in Fig. 3.

Notice that the addition of a matrix entry corresponded to changing a few pointers (*-above) without re-ordering the lists. The random accumulation of the system matrix is extremely beneficial to finite difference and finite element methods where it is usually convenient to construct the matrix in an arbitrary fashion.

INDEX	CE	ITAG	LNXT	LCOL
1	4	1	8 *	1
2	2	2	3	2
3	-1	3	6	4
4	3	3	0	7
5	-2	4	0	
6	-1	4	0	
7	2	4	0	
8	-1.1	2	5 *	

Figure 3: The linked-list after the introduction of a new matrix entry.

4.2 PRECONDITIONING WITH ZOLLENKOPF SPARSITY

The preconditioning of the S-matrix is attained quite easily since the sparsity of the Cholesky L-matrix was chosen to have the same sparsity as the S-matrix. The choice for this particular sparsity pattern was chosen by Kershaw [7] and this thesis exploits this choice by conveniently using the linked-list vectors for a dual purpose: accessing both the S and L matrices. The dual access feature results in a storage saving -- no auxiliary linked-lists are required. The only drawback stems from the fact that the calculation of a particular L(i,j) requires the search through each column (k) to extract the correct row coefficient for multiplication, for k=1 to (i-1) columns, (3.13) and (3.14). This can be a serious problem since in practice, the amount of time spent preconditioning rivals the time spent for the CG method to iterate to the solution. This may be avoided by employing yet a more crude approximation to the inverse.

Variable Window Preconditioning

The use of a variable window method was implemented to reduce the searching times accompanying the calculation of the preconditioned matrix. An incomplete factorization has the advantage that it may be made as accurate as wanted [13] and the variable window is a method that makes the preconditioned matrix as accurate as required. The amount of time required for preconditioning was found to be dependent upon the topology of the S-matrix. Two types of matrix topologies which commonly arise from finite element field formulations are the banded and arbitrary symmetric positive definite matrices.

Banded Positive Definite Symmetric System Matrix

Banded symmetric matrices often arise from finite difference and finite element problems corresponding to special geometries and tactical node-numbering schemes. An example of a symmetric matrix is shown in Fig. 4 which is the result of a linear 2-D finite element regular grid problem.

Reviewing equation (3.13), it was noted that the calculation of each L(j,i) corresponds to retrieving the previous L(j,k) and L(i,k) terms for k spanning column 1 to column (i-1). The matrix elements of interest in this calculation are located in the row=i and row=j rows. Although preconditioning poses no problem when the matrix is fully stored (zero entries stored also), there are drawbacks to using a

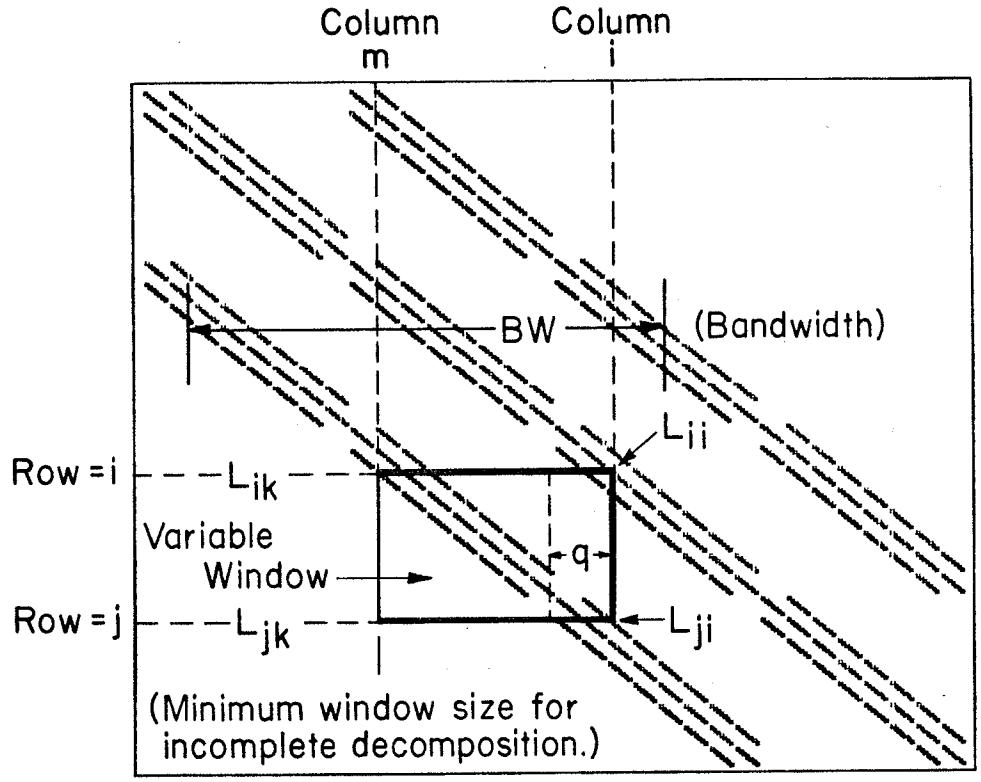


Figure 4: A banded sparse symmetric matrix.

sparsity linked-list scheme. The difficulty arises because the linked-list vectors cannot access any S-matrix entry directly. One must scan down a chosen column, constantly checking for the row location of a particular entry. Unfortunately, a time consuming search down the column must be performed to determine whether a particular entry lies in a given row. However, a saving may be obtained by noting that there are no contibutions to an L(j,i) preceding column m (Fig. 4). Therefore, equation (3.13) may be modified to be

$$L_{ji} = S_{ji} - \sum_{k=i-1-q}^{(i-1)} L_{jk} L_{ik} D_{kk} \qquad (4.1)$$

$$j=i,i+1,\ldots,N$$

with

$$D_{ii} = \frac{1}{L_{ii}} \qquad (4.2)$$

Experimentation with the new variable, q, led to some interesting results. If q was chosen to be the value

$$q \simeq i - \frac{BW}{2} \qquad (4.3)$$

where BW is the bandwidth of the S-matrix, then an incomplete Cholesky decomposition would be performed identical to Kershaw's method [7] (equivalent to choosing q=(i-2)). However, q was varied to the extent where some contributory entries were purposely omitted. Two important points arose from this test:

1. preconditioning was speeded up immensely; and
2. omission of these terms did not adversely affect the convergence rate of the conjugate gradient algorithm.

During testing, q was chosen to be small, e.g. from 1 to 3, for the calculation of the L-matrix. Although these values for q corresponded to a small window for the elimination of roughly 90% of the matrix contributions into the summation of (4.1), the results were extremely favorable. The resulting savings were due to minimizing the time spent preconditioning. The variable window factor, q, enabled preconditioning to be performed quickly by omitting contributory columns and thus eliminating costly searches. The reason

as to why the windowed preconditioned matrix is satisfactory for CG convergence may be due to the nearest neighbor coupling and/or the diagonal dominance of the S-matrix which enables a satisfactory approximate inverse to be calculated.

Arbitrary Positive Definite Symmetric System Matrix

Field problems involving arbitrarily sparse (symmetric) matrices, an example of which is depicted in Fig. 5, usually arise when finite elements are applied to regions of irregular boundaries or when arbitrary element node-numbering schemes are used.

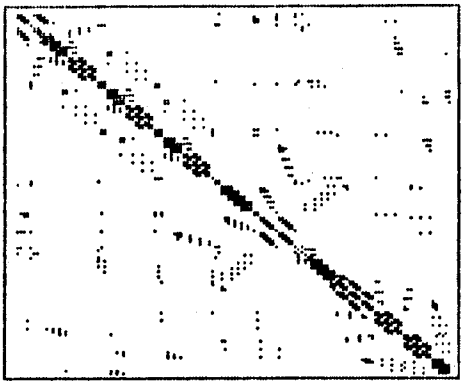


Figure 5: An arbitrary sparse symmetric matrix.

At first glance, variable window preconditioning would not appear to be applicable to an arbitrary sparse matrix since "nearest" neighbor coupling might be degraded. However, tests performed on arbitrary sparse matrices indicate that the identical method to calculate a variable window

preconditioned matrix for banded matrices may be used successfully. This result may suggest that diagonal dominance may play a greater role in the calculation of a suitable approximate inverse necessary for the CG process as opposed to neighboring row and column coupling.

In summary, all tests indicate that the variable window preconditioning technique produces a satisfactory approximate inverse matrix (for CG) quickly and efficiently. This particular scheme is ideally tailored to the Zollenkopf linked-list storage scheme and eliminates costly (sparsity) searches with no visible deterioration of CG convergence. The variable window factor, together with the Zollenkopf sparse storage method, minimizes the amount of time required to calculate preconditioned matrix.

4.3 STORAGE CONSIDERATIONS

The Zollenkopf storage scheme permits the solution of large sparse sets of linear equations to be solved efficiently while conserving the amount of computer storage due to sparseness and symmetry. As an example, consider a typical finite element system matrix of order N=500 with an average of 9 entries per row. The amount of computer storage necessary for the full storage of the S-matrix (including the zero terms) would be approximately 1 MByte. With the Zollenkopf scheme, the amount of storage required is approximately 40 KBytes including storage for the preconditioned

matrix and linked-list vectors. Although the full storage mode is convenient, the sparsity scheme enables very large (sparse) systems to be solved in a reasonable amount of computer memory.

Zollenkopf sparsity storage also allows for the (convenient) random accumulation of entries into the system matrix. The linked-list vectors are adjusted automatically with the addition or insertion of any matrix entry. Re-ordering of the linked-list vectors is unnecessary [16] since the linked-list addressing scheme is easily built into the CG algorithm.

The most important aspect of the application of a Zollenkopf sparsity scheme to conjugate gradients arises from the fact that the preconditioned matrix retains exactly the same sparsity as the system matrix. This immediately allows the same set of linked-list Zollenkopf vectors used to access S-matrix entries to be used to access the preconditioned matrix entries as well. Kershaw's choice for the preconditioned matrix to retain the identical sparsity as the S-matrix [7] provided the impetus to implement a sparse Zollenkopf preconditioned conjugate gradient process. The later development of a variable window preconditioning technique for the efficient calculation of the L-matrix led to a more efficient conjugate gradient linear equation solver.

The total amount of computer memory required for the incomplete Cholesky conjugate gradient method depends on two

parameters; N, the number of unknowns, and NR, the average number of non-zero entries per S-matrix row. Assuming that integer vectors require two bytes of storage while real vectors require four, the amount of computer storage required is approximately

$$\text{Store} \cong \frac{N}{512} (3NR + 11) \quad \text{(Kbytes)} \tag{4.4}$$

Equation (4.4) includes the storage required by the linked-list vectors (including L) and five vectors of length N from equations (3.7) to (3.11) including the vector formed by the multiplication of S with the direction vector. In addition, the $\underline{b}$ vector is assumed to be stored in the residual vector initially.

The Zollenkopf sparsity technique coupled with the conjugate gradient method provides a practical solution technique to tackle large sparse sets of simultaneous linear equations.

Chapter V

FINITE DIFFERENCE AND FINITE ELEMENT MODELLING TECHNIQUES

In this thesis, a major emphasis has been placed on the application of preconditioned congjugate gradients for the solution of sparse linear systems of finite difference and finite element equations originating from the discretization of elliptic partial differential equations. Chandra [17] points out that the conjugate gradient method is suitable for the solution of linear systems arising from these methods. Some of the more common types of partial differential equations of the elliptic class, namely the Laplace and Poisson equations, are considered here.

5.1 FINITE DIFFERENCE MATRIX ACCUMULATION

Consider a general two-dimensional boundary value problem (Fig. 6) where the potential field is governed by Laplace's equation,

$$-\nabla^2\phi = 0 \tag{5.1}$$

or more explicitly (in Cartesian coordinates),

$$-\frac{\partial^2\phi}{\partial x^2} - \frac{\partial^2\phi}{\partial y^2} = 0 \tag{5.2}$$

In order to model the region of Fig. 6 with finite differ-

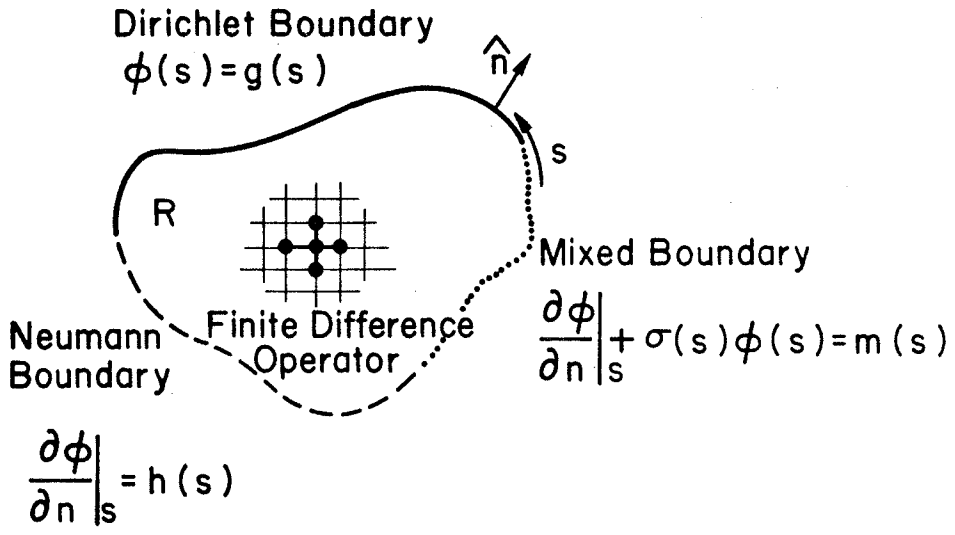


Figure 6: A boundary value problem with finite difference discretization.

ences, Laplace's equation (5.2) must be discretized. The discretization is performed by assuming a Taylor series expansion for ϕ. Assuming a one-dimensional expansion,

$$\phi_{i+1} = \phi_i + h\phi_i' + \frac{h^2}{2}\phi_i'' + \ldots \qquad (5.3)$$

$$\phi_{i-1} = \phi_i - h\phi_i' + \frac{h^2}{2}\phi_i'' - \ldots \qquad (5.4)$$

where h is the (fixed) distance between the node points. The addition of (5.3) and (5.4) produces

$$\phi_{i+1} + \phi_{i-1} = 2\phi_i + h^2\phi_i'' + O(h^2) \qquad (5.5)$$

where $O(h^2)$ denotes the error due to the truncation of the

high-order derivative terms. Re-arranging (5.5) and ignoring the error term,

$$\phi_i'' = \frac{\phi_{i-1} - 2\phi_i + \phi_{i+1}}{h^2} \qquad (5.6)$$

results in an expression for the second derivative. Extending (5.6) to the two-dimensional case with the substitution of (5.6) into (5.2) using double subscripts results in

$$-\phi_{i-1,j} - \phi_{i,j+1} + 4\phi_{i,j} - \phi_{i+1,j} - \phi_{i,j-1} = 0 \qquad (5.7)$$

Equation (5.7) corresponds to the application of Laplace's equation to the interior of Fig. 6. The Laplacian finite difference operator is shown in Fig. 7.

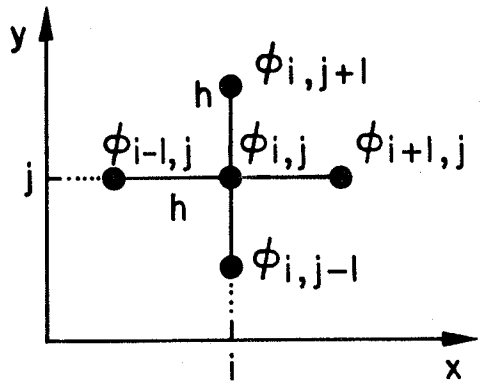


Figure 7: The five-point Laplacian finite difference operator.

A Dirichlet boundary condtion requires a modification to five-point operator. Consider

$$\phi_{i+1,j} = g(s) \qquad (5.8)$$

where g(s) is a known boundary potential. Equation (5.7) is

modified (via the substitution of (5.8) into (5.7)) to be

$$-\phi_{i,j-1}-\phi_{i,j+1}+4\phi_{i,j}-\phi_{i,j-1} = g(s) \qquad (5.9)$$

If one considers the application of equations (5.7) and (5.9) to a Dirichlet boundary value problem with equi-distant nodes, the result is a linear system of equations having the form

$$S\underline{\phi}=\underline{b} \qquad (5.10)$$

where S is symmetric and positive definite and $\underline{b}$ consists of Dirichlet source terms.

The finite difference technique also has applications for different types of boundary conditions, namely of the Neumann and mixed type. These boundary conditions produce a non-symmetric system matrix and the form of the conjugate gradient algorithm cannot be used. However, if equation (5.10) is multiplied by the system transpose matrix, i.e.

$$S^T S\underline{\phi} = S^T\underline{b} \qquad (5.11)$$

the resulting system can be solved by the conjugate gradient algorithm. This is possible because the product of S with its transpose results in a symmetric positive definite matrix (Appendix B). Although the multiplication of a matrix by its transpose is a costly computational process, Beaubien [18] had developed a symmetric thirteen-point symmetric finite difference operator which is numerically equivalent to

(5.11). This operator has the benefit of accumulating the (symmetric) system matrix directly without excessive storage requirements, especially for the case of large sparse systems. For these large matrices, the density of the non-zero terms is approximately doubled, however, only the lower triangular portion need be stored with the sparsity scheme mentioned previously.

5.2 THE FINITE ELEMENT METHOD

The finite element method is a numerical technique whereby a continuum problem, whose field behavior is governed by a partial differential equation, is discretized resulting in a large number of simultaneous linear equations. For the Laplacian and Poisson partial differential equations, the set of linear equations turns out to be symmetrical, enabling the use of the conjugate gradient method. The finite element formulation has many benefits for the solution of the continuum problem. One advantage is the ease of accommodating Dirichlet boundaries regardless of the geometry of the problem. Also, curved boundaries may be approximated using specialized elements; a task much more cumbersome with finite differences. Another advantage involves governing the topology of the system S-matrix through element layout for solution schemes dependent on matrix topologies. Generally, the flexibility of the finite element method makes it a desirable favorite over finite differences.

The Variational Approach

The variational approach is explained by Wexler [14] and briefly discussed here with particular attention to the matrix accumulation of the finite element method. Consider an elliptic second order partial differential equation that corresponds to a field distribution. Generally,

$$L\phi = f \tag{5.12}$$

where L denotes a linear operator, $\phi(\bar{r})$ the solution and $f(\bar{r})$, the known source distribution. Again, let L assume the Laplacian form,

$$L = -\nabla\cdot\kappa\nabla\phi \tag{5.13}$$

within an inhomgeneous, isotropic medium, κ being the permeability or permittivity of the medium. The boundary conditions are of the form:

$$\phi(s) = g(s) \tag{5.14}$$

$$\left.\frac{\partial\phi}{\partial n}\right|_s = h(s) \tag{5.15}$$

$$\left.\frac{\partial\phi}{\partial n}\right|_s + \sigma(s)\phi(s) = m(s) \tag{5.16}$$

known as the Dirichlet, Neumann, and mixed boundary conditions, which may be applied separately over segments on the boundary for a closed region.

The integral inner product of u and v (functions in the domain of the operator L) is defined to be

$$(u,v) = \int_R uv^* \, d\Omega \tag{5.17}$$

where R corresponds to an enclosed region and * denotes the complex conjugate. For simplicity, only the real case is considered so that

$$(u,v) = \int_R uv \, d\Omega \tag{5.18}$$

Considering a scalar field, the operator L can be shown to be self-adjoint, i.e.

$$(Lu,v) = (u,Lv) \tag{5.19}$$

and positive definite, i.e.

$$\begin{aligned} (Lu,u) &> 0 \quad , u \neq 0 \\ (Lu,u) &= 0 \quad , u = 0 \end{aligned} \tag{5.20}$$

when u and v satisfy homogeneous boundary conditions and are sufficiently differentiable but are otherwise arbitrary (with the exception to the application of a totally homogeneous Neumann boundary) [14]. If the (linear) operator L can be shown to be self-adjoint and positive definite under homogeneous boundary conditions, the minimization of the

functional

$$F = (L\phi,\phi) - (\phi,f) - (f,\phi) \qquad (5.21)$$

may be attained for the ϕ which is the solution to (5.12) [19,6/7]. A modification of (5.21) to include inhomogeneous Neumann boundary conditions after the substitution of (5.13) into (5.21) results in

$$F = \int_R \phi\nabla\cdot\kappa\nabla\phi \; d\Omega - 2\int_R \phi f \; d\Omega - 2\int_C \phi h \; ds \qquad (5.22)$$

Using Green's identity, (5.22) becomes

$$F = \int_R \kappa\nabla\phi\cdot\nabla\phi \; d\Omega - 2\int_R \phi f \; d\Omega - 2\int_C \phi h \; ds \qquad (5.23)$$

This functional is valid for inhomogeneous Dirichlet (by fixing the boundary nodes to their stated potential) and Neumann boundaries (where specified) and furnishes the homogeneous Neumann boundary condition where no other boundary conditions are specified.

The next step is to minimize the functional (5.23) and obtain the solution, ϕ . A direct method for minimizing the functional is known as the Rayleigh-Ritz method. In this procedure, the unknown $\phi(\bar{r})$ may be approximated as a linear combination of interpolation functions $\alpha_i(\bar{r})$ and node-point (discretized) field values ϕ_i, i.e.

$$\phi(\bar{r}) = \underline{\phi}^T\underline{\alpha} = \underline{\alpha}^T\underline{\phi} \qquad (5.24)$$

where $\underline{\phi}$ and $\underline{\alpha}$ are column vectors [20,5-43]. The interpolation (shape) functions are usually polynomials with the requirement that $\alpha_i(\bar{r})$ has a unit value at its associated node (at $\bar{r}_i$) and vanishes at all other nodes [14]. The substitution of (5.24) into (5.23) yields

$$F = \underline{\phi}^T \int_R \kappa \nabla \underline{\alpha} \cdot \nabla \underline{\alpha}^T \, d\Omega \underline{\phi} - 2\underline{\phi}^T \int_R \underline{\alpha} f \, d\Omega - 2\underline{\phi}^T \int_C \underline{\alpha} h \, ds \qquad (5.25)$$

The integrand of the first integral of (5.25) has the form of a square matrix, S, whose typical entry is

$$S_{ij} = \int_R \kappa \nabla \alpha_i \cdot \nabla \alpha_j d\Omega \qquad (5.26)$$

The two remaining integrals combine to form a column vector, $\underline{b}$, with entries such as

$$b_i = \int_R \alpha_i f \, d\Omega + \int_C \alpha_i h \, ds \qquad (5.27)$$

Therefore, the functional of (5.25) is a matrix equation having the form

$$F = \underline{\phi}^T S \underline{\phi} - 2\underline{\phi}^T \underline{b} \qquad (5.28)$$

To find the $\underline{\phi}$ which minizes F, the derivative of F with respect to $\underline{\phi}$ is performed yielding

$$\frac{\partial F}{\partial \phi} = 2S\underline{\phi} - 2\underline{\phi}^T \underline{b} = \underline{0} \qquad (5.29)$$

resulting in

$$S\underline{\phi} = \underline{b} \qquad (5.30)$$

where S is a symmetric[4] positive definite matrix, which also has the property of being quite sparse. The resulting set of equations are then quite easily solved with the conjugate gradient process.

5.3 CG MODIFICATION FOR DIRICHLET BOUNDARY CONDITIONS

Ultimately, a finite element formulation must resort to the modification of the S-matrix to transfer the Dirichlet nodes to the b vector, thus reducing the S-matrix size. This may prove to be a costly procedure and a way to avoid the node transferal was suggested by Irons.[5] The procedure involves no re-dimensioning of the S-Matrix, but only a modification to the diagonal elements of the matrix and corresponding $\underline{b}$ vector. Consider

$$\phi_i = g(s) \tag{5.31}$$

the known Dirichlet potential. If the corresponding entry on the diagonal of the S-matrix, S(i,i) is relatively small, this diagonal entry may be replaced by an entry, H, where H is 4 or 5 orders of magnitude greater than the value of the original entry. At the same time, the b(i) entry is re-

---

[4] The first integrand of (5.25) contained a row by column vector multiplication where the row vector was the transpose of column vector, thus obtaining a square symmetric matrix.

[5] In conversation with A. Wexler.

placed with

$$b_i = g(s)H \tag{5.32}$$

This process has the effect of de-emphasizing the off-diagonal terms of the particular row and places more emphasis on the diagonal term. Consider the ith equation explicitly

$$s_{i1}\phi_1 + \ldots + s_{ii}\phi_i + \ldots + s_{iN}\phi_N = b_i \tag{5.33}$$

Since the off-diagonal terms are small in comparison to the diagonal term, (5.33) may be written as

$$s_{ii}\phi_i + \varepsilon = b_i \tag{5.34}$$

where ε is the error term. Solving (5.34) for $\phi(i)$ ignoring the error term results in

$$\phi_i \simeq \frac{b_i}{s_{ii}} \tag{5.35}$$

Substituting (5.32) into (5.35) produces

$$\phi_i = \frac{b_i}{s_{ii}} = \frac{g(s)H}{H} = g(s) \tag{5.36}$$

the desired Dirichlet potential.

The Irons technique requires little time to perform and reduces the program complexity in comparison to the transferal of the Dirichlet nodes to the right hand vector. The trade-off is reflected in the accuracy of the value for the Dirichlet potential obtained. The larger H becomes, the better the accuracy of the potential.

5.4 CG SOLUTION FOR THE "PURE" NEUMANN PROBLEM

The ability to solve the pure Neumann problem presents a problem to many linear equation solution techniques. This occurs because the system matrix is singular and not readily solvable by direct solution techniques. However, Forsythe and Wasow [21,240,376-377] state that the Successive Over-Relaxation method (S.O.R.) can solve the Neumann system of equations and produce a solution which is correct with respect to an arbitrary additive constant.

When the Neumann problem is solved with the preconditioned conjugate gradient method, the result is equivalent to the S.O.R. solution (to within an additive constant).

5.5 OBTAINING QUICK CONVERGENCE WITH A STRATEGIC STARTING SOLUTION

In problems that require the successive solution to a linear system of equations, e.g. nonlinear magnetic field problems, a direct solution technique is often impractical. The amount of work involved in inverting a matrix requires a large amount of computational time, especially if many stages are required. The conjugate gradient method has an advantage in this aspect since the solution from the previous stage may be used as the starting point for the current stage. This means that the new solution may require only a few iterations for convergence (much less than starting with an arbitrary starting vector) with significant time savings, as opposed to inverting the current system matrix, as is the case with a direct solution technique.

In some nonlinear problems, the topology of the system matrix remains unchanged from stage to stage (while the values of the matrix entries change). Since the sparsity pattern remains constant, there is no need to reconstruct the linked-list vectors for the new matrix at each stage. The contents of the S-matrix are zeroed and the new entries are accumulated into the (old) zeroed positions. This technique is more efficient than rebuilding the linked-list vectors at each matrix accumulation stage. For these problems, preconditioning at each stage may be avoided and performed after several nonlinear iterations. It may be possible that the previously calculated preconditioned matrix is sufficient for the CG process after several nonlinear iterations provided the system matrix does not change radically at each iteration. Eliminating the preconditioning step at each nonlinear stage results in a time savings.

Chapter VI

EXAMPLES OF FIELD SOLUTIONS

This chapter provides some examples of the use of conjugate gradients for the solution of field problems arising from finite difference and finite element formulations. The examples that follow exemplify three different properties of conjugate gradients that are discussed in this thesis:

1. The speed of the sparse CG method in comparison with another well-known sparsity technique;
2. The effect of the windowed preconditioning factor on the convergence of the CG method for arbitrary sparse symmetric systems of linear equations; and
3. The ability of the conjugate gradient method to solve a singular system of linear equations resulting from a finite element pure Neumann boundary value problem.

6.1 2-D FINITE DIFFERENCE REGULAR GRID PROBLEM

Consider a homogeneous square region with Dirichlet boundary conditions (Fig. 8). Since the regular grid tested employed a simple node-numbering scheme, the system matrix representing the resulting set of linear equations is banded and symmetric.

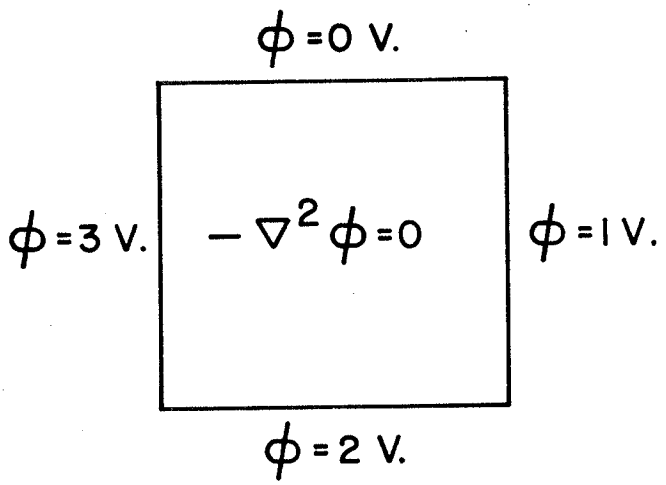


Figure 8: A finite difference problem with Dirichlet boundary conditions.

Comparison of ICCG with Zollenkopf Bi-Factorisation

A comparison between a sparse, symmetric Zollenkopf bi-factorisation routine (SYMPAK) (Appendix B) and the incomplete Cholesky conjugate gradient method was performed for a variable number of unknowns within the region of Fig. 8. The test was performed with a nearly identical vector storage allocation since both techniques are based on a Zollenkopf linked-list sparsity storage scheme. The identical vector storage requirements would also ensure (approximately) the same amount of computational overhead involved in accessing and passing vectors between program subroutines. This test was chosen to provide a fair comparison with two sparsity techniques as opposed to a comparison with a direct matrix inversion method. The testing was done on an AMDAHL 470/V8 computer using FORTRAN.

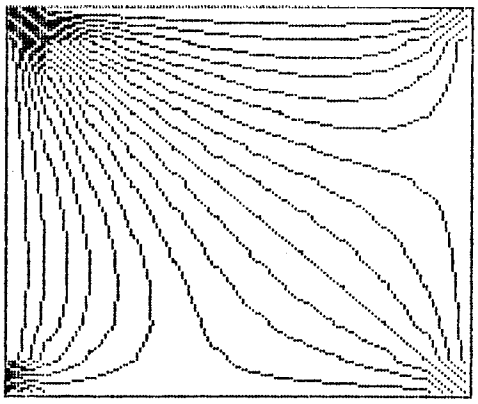


Figure 9: The scalar potential field for the finite difference problem.

The resulting field solution for the finite difference problem is shown in Fig. 9.

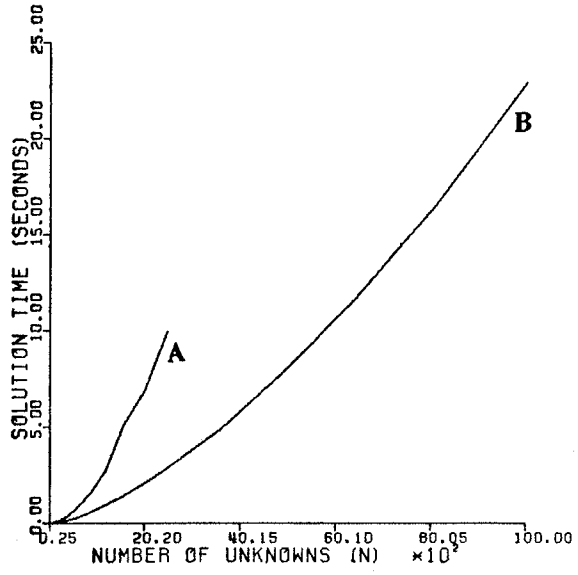


Figure 10: Solution times of SYMPAK (A) and ICCG (B) versus N.

The solution times for the SYMPAK and ICCG methods are depicted in Fig. 10. Curve A corresponds to the SYMPAK solution time while curve B corresponds to the ICCG solution. The SYMPAK program was not run for a number of unknowns beyond N=2500 due to storage exhaustion while ICCG was able to solve a problem containing 10,000 unknowns for the same storage allocation. The storage requirements are different since the Zollenkopf bi-factorisation technique produces fill-ins which require a greater storage allocation (Appendix B).

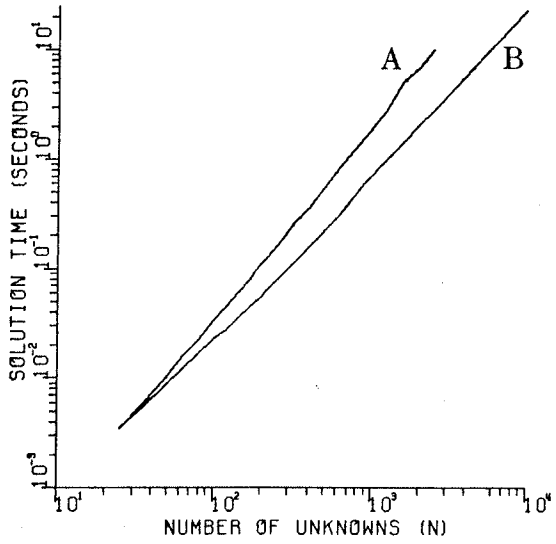


Figure 11: A logarithmic plot of the solution time versus N.

In order to establish a relationship between the solution time and the number of unknowns (N) for the two methods, the corresponding log-log plot is presented in Fig. 11. The results of Fig. 11 indicate that ICCG has a solution time

which is proportional to $N^{(1.5)}$ while SYMPAK has a relationship proportional to $N^{(1.75)}$.

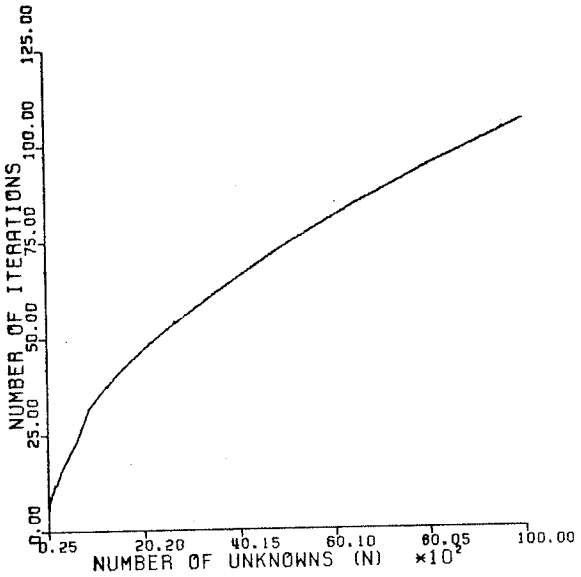
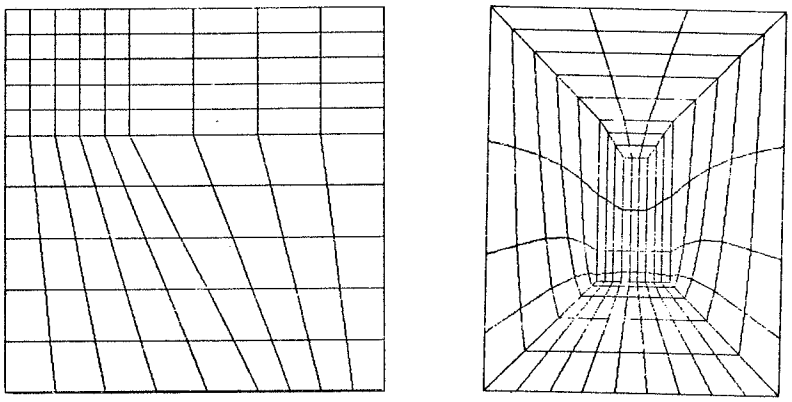


Figure 12: The number of ICCG iterations versus the number of unknowns.

Fig. 12 demonstrates the number of iterations required for the ICCG process to converge for various sizes of N. The result indicate that the number of iterations needed for the method to converge is approximately $\sqrt{N}$.

The results of this test indicate that the ICCG method has a definite time savings advantage as well as a more effective storage scheme than the Zollenkopf bi-factorisation method.

6.2 2-D FINITE-ELEMENT PROBLEM



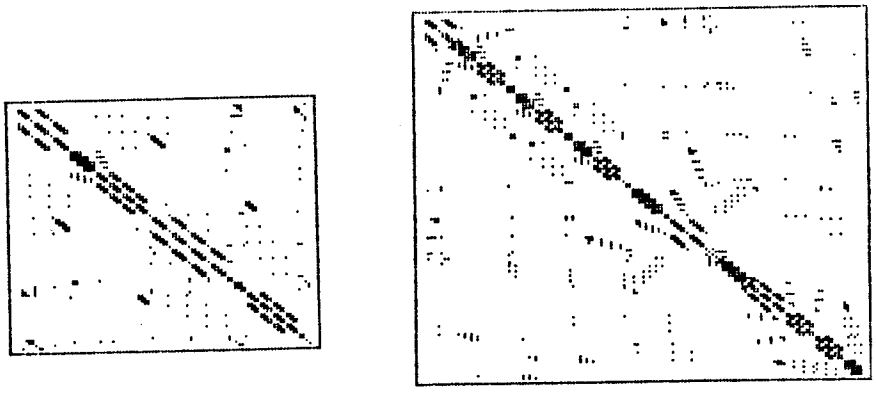
(a) (b)

Figure 13: Two FEM problems with arbitrarily numbered nodes.

Consider a two-dimensional finite element field problem for the solution of Laplace's equation which contains an arbitrary node numbering scheme such as those of Fig. 13 [23]. The resulting system matrix will be symmetric, positive definite and arbitrarily sparse (Fig. 14), since the sparsity pattern is directly related to the node-numbering scheme.

Windowed Preconditioning of an Arbitrary Symmetric Matrix

This test is designed to determine the effect of varying the window factor used for the preconditioning technique. In Chapter IV, it was shown that choosing q=(i-2) for equation (4.1) was equivalent to performing a "Kershaw" Cholesky decomposition, but any other value would produce an even less

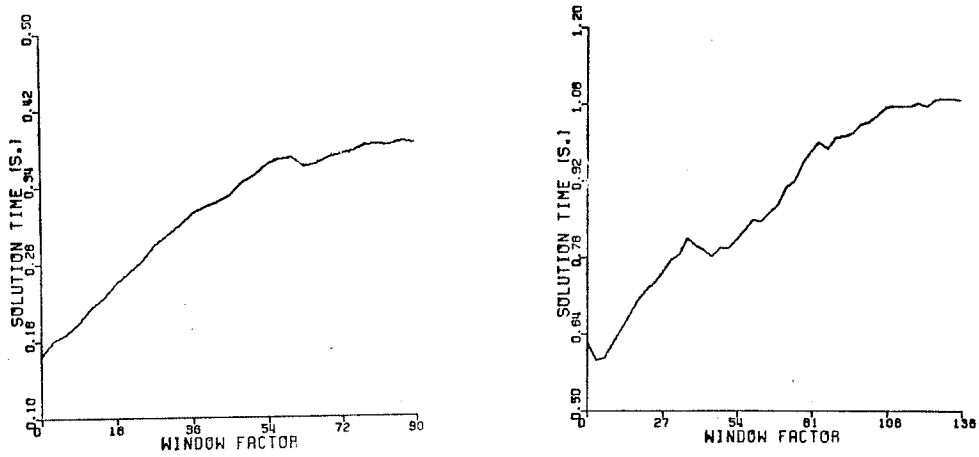


(a) (b)

Figure 14: The matrix sparsity of the two FEM problems.

accurate decomposition. For banded matrices, it was noted that the window factor could be varied in proportion to the bandwidth of the matrix (or less) with the rate of the ICCG convergence equivalent to using q=1 (the most inaccurate factorization). However, for arbitrary sparse matrices where there is no distinguishable bandwidth, a variable window factor will produce a less accurate factorization (since some contributory S-matrix terms are purposely omitted). Since it is desirable to minimize the calculation of the preconditioning matrix, the window factor for the finite element problems was varied from q=1 to q=(i-2).

The results from varying the window factor are shown in Fig. 15. From Fig. 15, it may be seen that keeping the window



(a) (b)

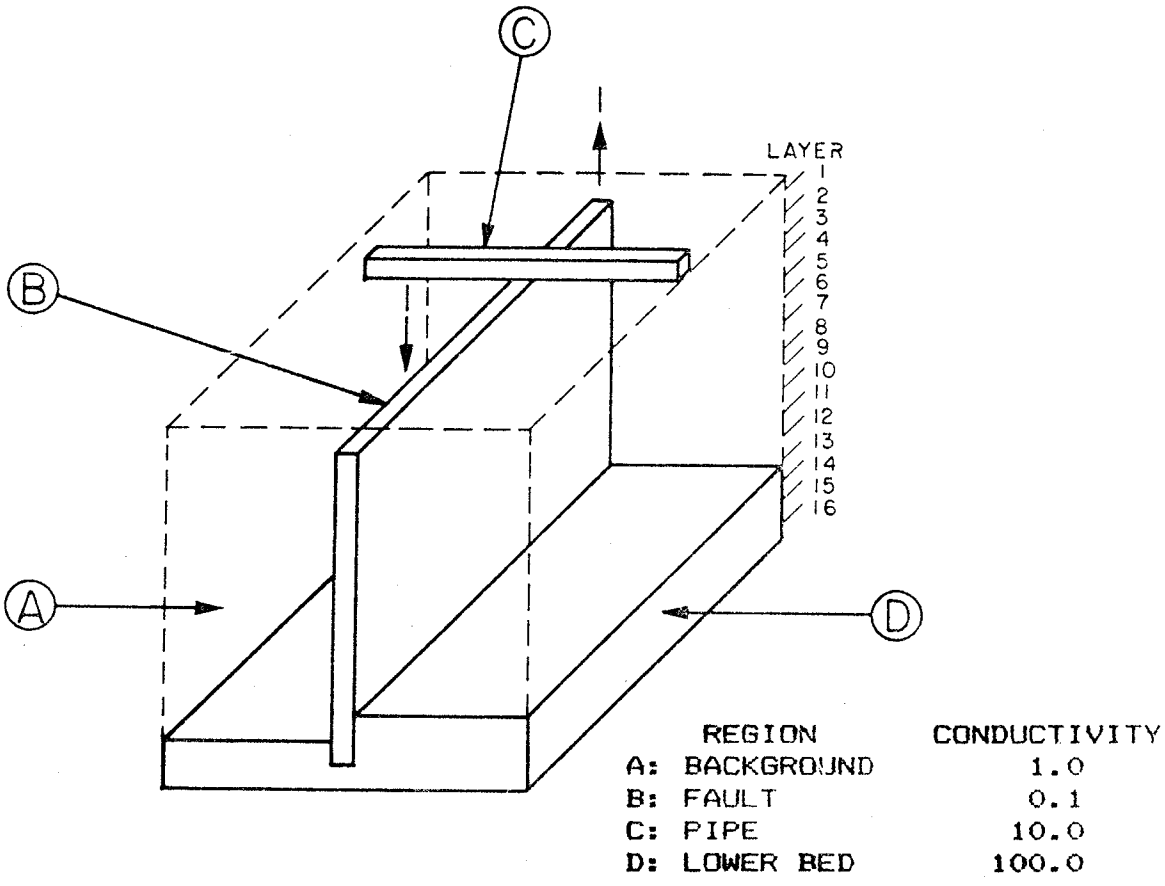
Figure 15: The solution time versus the variable window factor.

factor to a minimum reduces the amount of time spent preconditioning with no adverse effects to solution convergence. The amount of iterations necessary for convergence for the two problems of Fig. 13 remained constant for the complete range of window factors, and any time savings were due to the window factor monitoring the preconditioning process.

From the results, one may conclude that a small window factor is sufficient for determining an adequate preconditoned matrix for the conjugate gradient process. The smaller window factor reduces the searching required for sparse preconditioning, realizing a savings in the overall solution time with little or no perturbations to the number of iterations required for the conjugate gradient process to converge.

6.3 3-D FINITE-ELEMENT PROBLEM

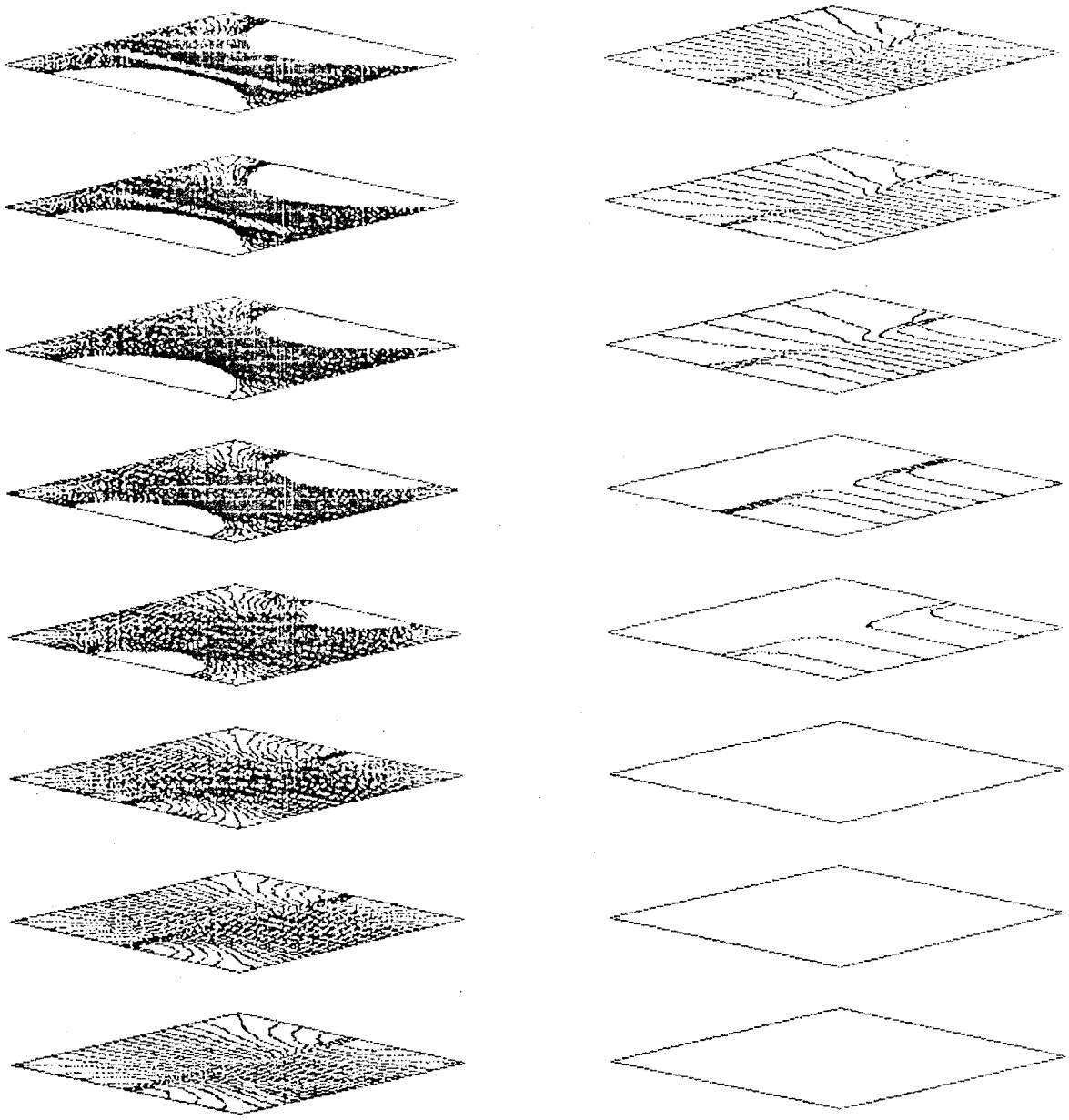
This example is one that is typical of geophysical techniques where a subsurface scalar electric potential need be determined. The applications may range from the detection of ore bodies, oil and gas deposits or the effects of ground current return paths of DC transmission lines [22]. The problem that is considered deals with the effects of a pipeline which is situated in the path of a ground return line for a high voltage DC transmission line.



REGION	CONDUCTIVITY
A: BACKGROUND	1.0
B: FAULT	0.1
C: PIPE	10.0
D: LOWER BED	100.0

Figure 16: A three-dimensional FEM representation of underground strata.

The underlying strata have conductivities that have been previously determined (Fig. 16). The return ground current is modelled by the injection of current at two sites on the surface, representing the ground electrodes. In the finite element formulation, the resulting set of linear equations turns out to be singular. This poses a problem for many direct solution techniques since they are unable to solve the resulting singular system matrix. The conjugate gradient process, however, has the ability to solve the singular system of linear equations in much the same way that S.O.R. solves this system. The resulting field plot for the various layers is depicted in Fig. 17.



LAYER LEGEND

1	9
2	10
3	11
4	12
5	13
6	14
7	15
8	16

Figure 17: The scalar potential at different depths.

Chapter VII

CONCLUSIONS AND RECOMMENDATIONS

This thesis has shown that the incomplete Cholesky conjugate gradient method is an efficient technique for the solution of sparse sets of (symmetric) linear equations. The incomplete Cholesky decomposition process served to accelerate the convergence of the CG method and was shown to be stable for a wide range of problems and a large number of unknowns.

The introduction of a Zollenkopf linked-list sparsity scheme into the CG algorithm increased both the storage and computational efficiency. The computational efficency was further improved by the introduction of the variable window technique into the preconditioning computation. The variable window factor was shown to be stable for the ICCG process when the system matrix is diagonally dominant, whether it be banded or otherwise. In addition, the storage and computational savings of the sparsity scheme are ideal for the solution of medium-sized problems on smaller, memory restricted computers.

The ICCG method was shown to be effective for a wide range of electric field problems involving finite element and finite difference formulations. The Dirichlet boundary conditions occurring in a finite element problem are easily

accommodated. Also, ICCG was able to obtain a solution for the pure Neumann problem, i.e. the solution to a singular system of linear equations. The Neumann problem cannot be solved by direct inversion solution techniques.

In the case of multi-iterative (e.g. nonlinear magnetic) problems, which require successive system matrix solutions, the solution from the previous iteration may be used as a starting estimate to the current system of equations. As a result, the solution to this system of equations would be attained much faster since the starting estimate is near the solution point for the current system matrix. For problems where the system matrix topology remains unchanged from one iteration to the next, a further saving may be realized by not rebuilding the linked-list vectors after each matrix accumulation. For these problems, preconditioning need not be performed at each iterative step, resulting in yet another time saving.

The ICCG method does not store any true form of the inverse system matrix. If problems arise where several source (right-hand) vectors require solutions for the same matrix, ICCG would have to iterate right from the beginning for each vector. For this situation, it is reccommended that one employ a scheme such as Zollenkopf bi-factorisation, which retains the inverse matrix form.

The Zollenkopf sparsity method as presented is not ideally appropriate for specialized parallel computer hardware.

Much of the information is scattered throughout the linked-list vectors (due to the random accumulation of matrix entries). Since parallel techniques usually require a rigidly ordered data structure, an implementation using the Zollenkopf storage scheme would likely be complex and subject to significant overhead costs. The original ICCG form though quite vectorizable (if full storage for the matrices is assumed), would not be feasible for the solution of large matrices due to storage limitations. The exception may be the banded matrix. A parallel scheme may be implemented such that operations may be performed block-sparsely, thus reducing the amount of main store.

In dealing with nonsymmetric system matrices, a scheme may be developed similar to one implemented by Beaubien [18] making use of the fact that a matrix multiplied by its transpose results in a symmetric, positive definite matrix. It may be possible to develop a finite element matrix accumulation scheme which enters the correct terms into the transposed product matrix on a point accumulative basis rather than brute force multiplication of the system matrix. A side-effect due to the multiplication by the transposed matrix produces a system which is more ill-conditioned than the original system matrix. In this case, the ICCG (symmetric) method can still be used to solve the resulting set of equations.

Appendix A

VECTOR ALGEBRA AND IDENTITIES

1. Inner (scalar) product of two vectors $\underline{x}$ and $\underline{y}$.

$$<\underline{x},\underline{y}> = \underline{y}^T\underline{x} \tag{A.1}$$

If $\underline{y}=\underline{x}$ then $<\underline{x},\underline{x}> \geq 0$ and $<\underline{x},\underline{x}>=0$ if $\underline{x}=0$.

2. Symmetric Identity for an inner product of two vectors $\underline{x}$ and $\underline{y}$.

$$<\underline{x},\underline{y}> = <\underline{y},\underline{x}> \tag{A.2}$$

3. Bi-linear Property of vector inner products.

$$<(a_1\underline{x}_1+a_2\underline{x}_2),\underline{y}> = a_1<\underline{x}_1,\underline{y}> + a_2<\underline{x}_2,\underline{y}> \tag{A.3}$$

for any real numbers a_1 and b_2.

4. Orthogonal Vectors. Two vectors are orthogonal if

$$<\underline{x},\underline{y}> = 0 \quad \underline{x}\neq\underline{y} \tag{A.4}$$

5. S-Orthogonal vectors. The vector $\underline{p}$ is defined to be S-orthogonal if the inner product with $\underline{p}$ satisfies

$$<\underline{p},S\underline{p}> = 0 \quad \underline{p}\neq\underline{0} \tag{A.5}$$

6. Vector Differentiation. Let $\underline{y}$, $\underline{b}$, and $\underline{c}$ be column vectors, with

$$\underline{y} = \underline{b}^T\underline{c} \tag{A.6}$$

Then,

$$\frac{\partial \underline{y}}{\partial \underline{a}} = \frac{\partial (\underline{b}^T \underline{c})}{\partial \underline{a}} = \frac{\partial \underline{b}^T}{\partial \underline{a}} \underline{c} + \frac{\partial \underline{c}^T}{\partial \underline{a}} \underline{b} \qquad (A.7)$$

Appendix B

MATRIX PROPERTIES

1. Positive Definite Matrix. A matrix is positive definite if it satisfies the property

$$\langle S\underline{u},\underline{u}\rangle > 0 \quad \underline{u}\neq\underline{0} \tag{B.1}$$

and

$$\langle S\underline{u},\underline{u}\rangle = 0 \quad \underline{u}=\underline{0} \tag{B.2}$$

2. Self-Adjoint Property for Matrices. The matrix S is self-adjoint if it satisfies the following property

$$\langle S\underline{u},\underline{v}\rangle = \langle \underline{u},S\underline{v}\rangle \tag{B.3}$$

Expanding (B.3) results in

$$\underline{v}^T S\underline{u} = \underline{v}^T S^T \underline{u} \tag{B.4}$$

Therefore, S is self-adjoint if and only if S is symmetric.

3. Symmetrizing a Non-Symmetric System of Linear Equations. A non-symmetric matrix system of linear equations may be pre-multiplied by its matrix transpose to produce a symmetric positive definite system of linear equations, providing the matrix is nonsingular. Consider

$$M^T M\underline{x} = M^T \underline{b} \tag{B.5}$$

The system matrix is symmetric since

$$(M^TM)^T = M^TM \tag{B.6}$$

To establish positive definiteness, consider

$$M\underline{x} = \lambda\underline{x} \tag{B.7}$$

The transposed form of (B.7) is

$$\underline{x}^TM^T = \lambda\underline{x}^T \tag{B.8}$$

Pre-multiplying (B.7) by the transpose results in

$$M^TM\underline{x} = \lambda M^T\underline{x} \tag{B.9}$$

Taking the scalar vector product of (B.9) with $\underline{x}$ and expanding produces

$$\underline{x}^TM^TM\underline{x} = \lambda(\underline{x}^TM^T)\underline{x} \tag{B.10}$$

Substituting (B.8) into the quantity in parenthesis of (B.10) produces

$$\underline{x}^TM^TM\underline{x} = \lambda^2\langle\underline{x},\underline{x}\rangle \tag{B.11}$$

Since the right-hand side of (B.11) is positive, the product of M with its transpose is therefore positive definite.

4. LDL Factorization for Symmetric Matrices. A symmetric nonsingular matrix of order N may be factored as

$$S=LDL^T \tag{B.12}$$

where L is a unique unit lower triangular matrix and D is a diagonal matrix. Consider the matrix of Fig. 1. The factorized L and D matrices are

$$L=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -.5 & 1 & 0 \\ -.5 & -.5 & -.2 & 1 \end{bmatrix} \qquad D=\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2.5 & 0 \\ 0 & 0 & 0 & .4 \end{bmatrix}$$

The inversion of the $\tilde{L}$-matrix is easily obtained as a product of (N-1) L matrices. Each column of the L matrix corresponds to a factored inverse $\tilde{L}$ matrix of order N containing this column (with its terms negated), ones on the diagonal, and zeros elsewhere. The inverse of L, in its product form is given by

$$L^{-1} = \tilde{L}^{(3)}\tilde{L}^{(2)}\tilde{L}^{(1)} \qquad (B.13)$$

or more explicitly,

$$L^{-1}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & .2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & .5 & 1 & 0 \\ 0 & .5 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ .5 & 0 & 0 & 1 \end{bmatrix}$$

5. Bi-Factorisation Method for Systems of Linear Equations. This is a factorization technique whereby a matrix, S, is decomposed via a column and row elimination operation. Given

$$A\underline{x} = \underline{b} \qquad (B.14)$$

a column and row elimination is equivalent to a pre-

multiplication of S by a matrix, L, and similarly, a post-multiplication of S by R. Performing the first column and row operation is equivalent to

$$L^{(1)}AR^{(1)} \tag{B.15}$$

The elimination of the subsequent column and row is represented as

$$L^{(N)}L^{(N-1)}\ldots L^{(1)}AR^{(1)}\ldots R^{(N-1)}R^{(N)} \tag{B.16}$$

such that (B.16) is equivalent to the identity matrix,

$$LAR = I \tag{B.17}$$

where $L=L^{(N)}L^{(N-1)}\ldots L^{(2)}L^{(1)}$ and $R=R^{(1)}R^{(2)}\ldots R^{(N-1)}R^{(N)}$. Therefore, from (B.17)

$$A^{-1}= RL \tag{B.18}$$

and

$$\underline{x} = RL\underline{b} \tag{B.19}$$

where each factored matrix $L^{(i)}$ and $R^{(i)}$ are trivially inverted [14]. This method is employed in the SYMPAK package for the solution of simultaneous equations [24] which employs a Zollenkopf linked-list sparsity scheme.

REFERENCES

1. Hestenes, M.R. and Stiefel, E., Methods of Conjugate Gradients for Solving Linear Systems, Journal of Research of the National Bureau of Standards, Vol. 49, No. 6, 1952.

2. Reid, J.K., Large Sparse Sets of Linear Equations, Mathematics Branch, Atomic Energy Research Establishment, Harwell, England, 1971.

3. Westlake, J.R., A Handbook of Numerical Matrix Inversion and Solutions of Linear Equations, John Wiley & Sons, Inc., New York, 1968.

4. Ralston, A., A First Course in Numerical Analysis, McGraw-Hill Book Company, New York, 1965, 425-426, 442-445.

5. Jennings, A. and Malik, G.M., The Solution of Sparse Linear Equations by the Conjugate Gradient Method, International Journal for Numerical Methods in Engineering, Vol. 12, 1978, 141-158.

6. Engeli, M. et al, Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems, Brickhauser-Verlag, Basle, 1959.

7. Kershaw, D.S., The Incomplete Cholesky-Conjugate Gradient Method for the Iterative Solution of Systems of Linear Equations, Journal of Computational Physics 26, 1978, 43-65.

8. Jennings, A., Matrix Computation for Engineers and Scientists, John Wiley & Sons, New York, 1977, 212-221, 316-319.

9. Gambolati, G., Fast Solution to Finite-Element Flow Equations by Newton Iteration and Modified Conjugate Gradient Method, International Journal for Numerical Methods in Engineering, Vol. 15, 1980, 661-675.

10. Evans, D.J., Alternating Direction Implicit Preconditioning Methods for Self Adjoint Elliptic Differential Equations, Comp. & Maths. with Appls., Vol. 7, Pergamon Press Ltd., 1981, 151-158.

11. Axelsson, O. and Gustafsson, I., Note on the Use of Preconditioned Conjugate Gradient Methods for Red-Black Ordered Five-Point Difference Schemes, Journal of Computational Physics 35, 1980, 284-289.

12. James, M.L., Smith, G.M., and Wolford, J.C., Applied Numerical Methods for Digital Computation, Harper & Row, Publishers, Inc., 1977, 194-199, 343-347.

13. Munksgaard, N., Solving Sparse Symmetric Sets of Linear Equations by Preconditioned Conjugate Gradients, ACM Transactions on Mathematical Software, Vol. 6, No. 2, 1980, 206-219.

14. Wexler, A., Perspectives on the Solution of Simultaneous Equations, Department of Electrical Engineering Technical Report #TR79-2, University of Manitoba, Winnipeg, Manitoba, 1979.

15. Classen, R.G., A Note on the use of the Conjugate Gradient Method in the Solution of a Large System of Sparse Equations, The Computer Journal, Vol. 20, No. 2, 1975, 185-186.

16. Zollenkopf, K., "Bi-Factorisation -- Basic computational algorithm and programming techniques," In Large Sparse Sets of Linear Equations, Reid, J.K. (ed.), Academic Press: New York, 1971, 75-96.

17. Chandra, R. et al, The Modified Conjugate Residual Method for Partial Differential Equations, Research Report No. 107, Department of Computer Science, Yale University, 1977.

18. Beaubien, M.J. and Wexler, A., "An Accurate Finite Difference Method for Higher Order Waveguide Modes", IEEE Trans. Microwave Theory Tech., MTT-16, pp. 1007-1011, 1968.

19. Tottenham, H., "Direct Variational Techniques", In Variational Methods in Engineering Vol.1, C. A. Brebbia and H. Tottenham (eds.), Southhampton Unversity Press, 1973, 6/1-6/15.

20. Wexler, A., Finite-Elements for Technologists, Department of Electical Engineering Technical Report #TR80-4, University of Manitoba, Winnipeg, Manitoba, 1980.

21. Forsythe, G.E. and Wasow, W.R., Finite-Difference Methods for Partial Differential Equations, John Wiley & Sons, Inc., New York, 1967, 240,376-377.

22. Telford, W.M., Geldart, L.P., Sheriff, R.E., and Keys, D.A., Applied Geophysics, London: Cambridge University Press, Chapter 8, 1976.

23. Klimpke, B.W., A Simple Finite Element Program, Electrical Engineering Department, TR#83-5, University of Manitoba, Winnipeg, Canada, 1983.

24. Shaw, J. and Wexler, A., SYMPAK/ASYMPAK A Program Package for Solving Large Sparse Systems of Linear Algebraic Equations, Department of Electrical Engineering Technical Report #TR79-1, University of Manitoba, Winnipeg, Manitoba, 1979.