CORRECTIONS.

1. Page ii. The last sentence should read "...in solving a Stefan problem...".

2. Page 7. Equation (21) should obviously be

$$\gamma_{j+1} v_j^T v_{j+1}^* = -\beta_j^* v_j^T v_{j-1}^* + O(\varepsilon).$$

3. Page 8 Equations (22) and (23) are each missing a factor $O(\varepsilon)$ in the $\sum$ term.

4. Page 30. The quantity p_2 requires defining. The third line of this page should read $02 = \Delta r.p_2$.

5. Page 59. Line 3 : ..(e.g. if $Dy(x) = x\, y(x) + y(x)$)...
Line 15: ..here $Q_o(x)$ is either $-\frac{1}{2}$ or $\frac{1}{2}x$....

6. Page 62. Line 10. A factor $\frac{1}{2}$ should be attached to the first term.
Lines 7,8,9,10. The term y^2 appearing in the denominator should be Δy^2.

7. Page 88. Cases g) and h) at the bottom of this page should be deleted.

NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS

by Colin John Wright

A thesis submitted to the Imperial College of Science and Technology of the University of London for the degree of Master of Philosophy.

August 1976

ACKNOWLEDGEMENTS

I wish to express my gratitude to Drs. A. Fonda and E.L. Ortiz for their advice and encouragement during the preparation of this Thesis.

ABSTRACT

This research started out as an attempt to find the eigenvalues of the Laplacian operator on a certain domain. In order to do this a variation of the well-known Lanczos minimised iteration technique was devised for isolating the eigenvalues of large sparse non-symmetric matrices. The required eigenvalues were then found via the usual finite difference approach.

The Laplace and Poisson equations were then solved on domains of a certain type by means of the method of lines and the Lanczos-tau method. Some error analyses are given. The errors incurred by a previous author in solving a Stef~~f~~an equation by a similar technique are considered.

CONTENTS

Introduction . . . page 1

PART ONE

Chapter 1: The method of Lanczos for non-symmetric matrices

1.1 Introduction . . . page 3

1.2 Variants of Lanczos' algorithm 4

1.3-1.6 Analysis of techniques 7

1.7 Examples 16

1.8 Conclusions 25

Chapter 2: Eigenvalues of a Differential Operator

2.1 Statement of problem and outline of finite difference approach . . . page 27

2.2-2.4 Necessary background - evaluating determinant of tridiagonal matrix; Laguerre's method; scaling 31

2.5-2.6 Results and comments 34

2.7 Experiences in root finding 51

Final Comments on Part One . . . page 55

PART TWO

Chapter 1: Introduction to Tau Methods

1.1 Introduction . . . page 56

1.2 The tau method 56

1.3-1.4 Canonical polynomials and their construction 57

1.5 Eigenvalue problems 59

Chapter 2: Some Lanczos-tau method of lines techniques

2.1 Introduction . . . page 61

2.2-2.3 Techniques 61

2.4 Error Analyses 65

2.5-2.7 Numerical results and comments 71

2.7-2.8 Collocation, prior integration and canonical polynomial techniques 73

Chapter 3: Matrix Lines Tau Method

3.1 Introduction . . . page 79

3.2 Method applied to Laplace's equation 79

3.3–3.6 Examples, results and comments 83

3.7–3.8 Extension of method to Poisson's equation with some examples 97

3.9–3.11 Examples with more complex boundary conditions handled 101

Chapter 4: Extensions of Method

4.1 Introduction . . . page 111

4.2–4.3 Eigenvalue problem 111

4.4 General elliptic boundary value problem 114

Chapter 5: An Error Analysis . . . page 118

Chapter 6: An Error Analysis of the Method of Wragg . . . page 122

Bibliography . . . page 126

INTRODUCTION.

The original purpose of this work was to find the eigenvalues, λ, of smallest modulus of

$$\nabla^2 \psi + \lambda \psi = 0 \qquad (1)$$

on the domain Γ of figure 1, subject to the conditions

$$\psi = 0 \quad \text{on } S_1 \qquad (2)$$

and $$\frac{\partial \psi}{\partial \nu} + c\psi = 0 \quad \text{on } S_2 \;. \qquad (3)$$

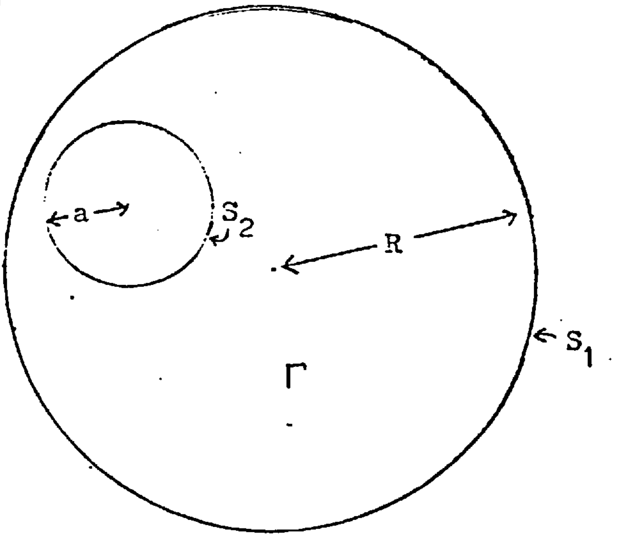

figure 1

The equation defined by (1), (2) and (3) occurs in the study of the physical theory of neutron chain reactors (see e.g. Weinberg and Wigner [31]). The simplest case is that of a bare homogeneous reactor having the shape of an infinitely high cylinder of radius R with an off-centre control rod of radius a (the centre of the control rod is at a distance b from the centre of the reactor). The method of Nordheim and Scaletter ([31] p 770) for this problem entails replacing the control rod by a point singularity in the neutron density - such a point singularity corresponds to an absorber of a certain strength. Polar coordinates are introduced, the eigenfunctions are approximated using Bessel functions of both the first and second kinds and the eigenvalues obtained. Control rods of non-zero radius only were considered in this work. First, the differential operator was approximated in the usual manner by means of a difference operator (see e.g. Fox [9] , Collatz [5]). The resulting large sparse banded non-symmetric matrix was reduced to tridiagonal form by means of a modified Lanczos (minimized iteration) method.

In the first chapter of part one we give an account of this modified method. Some examples illustrating various aspects of the behaviour of this algorithm are given. The roots of the resulting tridiagonal matrix are isolated by the method of Laguerre - this method being chosen because of its superior convergence properties - see chapter two for this. An abortive attempt at finding the eigenvalues using a variational method (Mikhlin [22] , Mikhlin and Smolitsky [23]) was made. This technique was abandoned because of the vast amount of computational effort required. At the time this work was done (1969-1971) the finite element techniques were not yet fully in vogue, hence their non-appearance.

Originally it was planned to extend the Lanczos tau method (in the form proposed by Ortiz [24]) to find the eigenvalues of the given problem. Inspired by Wragg's [37] solution to the Stefan problem, a combination of the method of lines (Berezin and Zhidkov [2] p 580) and various Lanczos tau methods were used to solve the equation of Laplace. Initially we met with little success, but then developed a matrix type technique which works extremely well on domains of a certain type. Error equations were set up and solved for most cases (including that of Wragg).

The layout of the second part, briefly, is as follows : We first give a general introduction to tau methods. In chapter two some unsuccessful attempts at solving Laplace's equation are outlined. The successful matrix type technique is then given for both the Laplace and Poisson equations and also the eigenvalue problem. Examples are given.

Some points regarding notation are necessary. Frequently we use [r], where r is real, this indicates the largest integer less than or equal to r. Entries aaaa and $.qqqq_p$ in tables mean .aaaa and $.qqqq \times 10^p$ respectively. An integer n appearing in the body of a table means 10^n. Entries in the bibliography are referred to by [m], the different uses made of square brackets are always clear. The Chapters in each of the two parts are numbered consecutively from one. Different numbering systems operate for equations, tables and figures in each of the several chapters, for example equation 21 of chapter 2 of part two is referred to as equation (21) in that chapter and equation (2.21) in other chapters of that part. There are no references from one part to equations etc. of the other.

PART 1

CHAPTER 1 : THE METHOD OF LANCZOS FOR NON-SYMMETRIC MATRICES

1.1 Wilkinson [35] describes satisfactory methods of computing eigensystems of non-symmetric matrices of reasonable order. For very large systems the situation is somewhat different in that available methods are not entirely satisfactory.

Tewarson [28, 29] does however describe a variation of the Gaussian similarity transformation by means of which the number of zero elements that become non-zero in reducing a very large sparse matrix to Hessenberg form is minimized.

Lanczos [18] suggested a method of minimized iterations for reducing a matrix to tri-diagonal form. This method has been further expounded by Wilkinson ([33],[35]). Paige [25] has described a variation of this method specifically suited to large sparse symmetric matrices. We here propose an extension of Paige's algorithm aimed at producing the eigenvalues of large arbitrary matrices.

The usual general Lanczos (minimized iteration) algorithm is:- Choose v_o and v_o^* to be null vectors and select v_1 and v_1^* arbitrarily (but not orthogonal), then compute for j=1,2,...,n:

$$\gamma_{j+1} v_{j+1} = A v_j - \alpha_j v_j - \beta_j v_{j-1} \;, \quad \alpha_j = (v_j^*)^T A v_j / (v_j^*)^T v_j$$

$$\beta_j = (v_{j-1}^*)^T A v_j / (v_{j-1}^*)^T v_{j-1}$$

$$\gamma^*_{j+1} v^*_{j+1} = A^T v^*_j - \alpha_j v^*_j - \beta^*_j v^*_{j-1} \;, \quad \alpha_j = v_j^T A^T v_j^* / (v_j^*)^T v_j$$

$$\beta^*_j = v^T_{j-1} A^T v^*_j / (v^*_{j-1})^T v_{j-1} \;.$$

The constants γ_j and γ_j^* are suitable scaling factors. In the absence of rounding and cancellation errors this algorithm ensures that the two sequences of vectors, viz. $v_1, v_2, \ldots, v_1^*, v_2^*, \ldots$ are biorthogonal.

For some value of $j \leq n$ (say s) it may happen that the scalar product $v_s^T v_s^*$ vanishes - this always occurs when the matrix is derogatory. Causey and Gregory [4] describe how to restart the algorithm in such circumstances. When the algorithm does fail in this way it follows that $AV=VT$, where $V=[v_1, \ldots v_s]$, T is tridiagonal and every eigenvalue of T is also an eigenvalue of A. In this work we will not be interested in restarting the algorithm if failure occurs, as T will locate some of the extreme eigenvalues of A. Also, under certain circumstances the eigenvalues of T_k , the leading $k \times k$

part of T, are likely to be good approximations to some of the extreme eigenvalues of A (see 1.7 and also chapter 2). Lanczos indicated this and Kaniel [12] and Paige [25] have given some results for symmetric matrices.

Several authors, including Lanczos [18] , Wilkinson [33] and Gregory [11] have pointed out that the orthogonality of the two sets of vectors is soon lost completely as a result of the cancellation errors which occur in the implementation of the algorithm. As a cure for this ill Wilkinson has suggested re-orthonormalizing each vector (as it is computed) against the previously computed vectors, while Gregory has proposed the retention of further non-zero terms in the recurrence relations - theoretically these terms should be zero, but in practice turn out not to be so. Neither of these techniques provides an efficient cure to the ills of the algorithm when it is applied to very large sparse matrices. In the next section we propose a new variation of the Lanczos algorithm which goes a long way toward solving the orthogonality problems encountered in its application to the problems of finding the eigenvalues of large general matrices.

1.2 The generalized Lanczos algorithm of 1.1 may be phrased somewhat differently as :-

1) Choose v_1 and v_1^* arbitrarily, but such that $v_1^{*T} v_1 = \pm 1$ $(=S_1)$ and compute $u_1 = A v_1$, $u_1^* = A^T v_1^*$.

2) For j=1,2,....,k , compute :-

$$\alpha_j = \frac{v_j^{*T} A v_j}{v_j^{*T} v_j} \qquad (1)$$

or

$$\alpha_j = \frac{v_j^{*T} u_j}{v_j^{*T} v_j} \qquad (2)$$

$$w_j = u_j - \alpha_j v_j \qquad (3)$$

$$w_j^* = u_j^* - \alpha_j v_j^* \qquad (4)$$

$$\phi_{j+1} = w_j^{*T} w_j \qquad (5)$$

$$S_{j+1} = \text{sign}(\phi_{j+1}) \qquad (6)$$

$$\gamma_{j+1} = (\phi_{j+1})^{\frac{1}{2}} \qquad (7)$$

$$v_{j+1} = w_j / \gamma_{j+1} \tag{8}$$

$$v^*_{j+1} = w^*_j / \gamma_{j+1} \tag{9}$$

$$\beta_{j+1} = \frac{v_j^{*T} A \, v_{j+1}}{v_j^{*T} \, v_j} \tag{10}$$

$$\beta^*_{j+1} = \frac{v_j^T \, A^T \, v^*_{j+1}}{v_j^{*T} \, v_j} \tag{11}$$

$$u_{j+1} = A \, v_{j+1} - \beta_{j+1} \, v_j \tag{12}$$

$$u^*_{j+1} = A^T \, v^*_{j+1} - \beta^*_{j+1} \, v^*_j \tag{13}$$

This defines the algorithm.

Using (8) and (9)

$$v_j^{*T} v_j = \frac{w^{*T}_{j-1}}{\gamma_j} \cdot \frac{w_{j-1}}{\gamma_j} = \frac{\phi_j}{\gamma_j^2} = \frac{1}{S_j} = S_j \quad (= \pm 1) \tag{14}$$

Also, by (13)

$$v_j^{*T} \, A = u_j^{*T} + \beta^*_j \, v^{*T}_{j-1}$$

and, substituting for u_j^{*T} from (4)

$$v_j^{*T} \, A = w_j^{*T} + \alpha_j \, v_j^{*T} + \beta^*_j \, v^{*T}_{j-1} \; . \tag{15}$$

Hence, by (10), (14) and (15)

$$\beta_{j+1} = S_j \, v_j^{*T} \, A \, v_{j+1} = S_j (w_j^{*T} + \alpha_j \, v_j^{*T} + \beta^*_j \, v^{*T}_{j-1}) \, v_{j+1} \; .$$

Using the bi-orthogonality property of the vectors, and also (8) it follows that

$$\beta_{j+1} = S_j \, S_{j+1} \, \gamma_{j+1} \; . \tag{16}$$

Similarly $\beta^*_{j+1} = S_j \, S_{j+1} \, \gamma_{j+1}$

$$= \beta_{j+1} \; . \tag{17}$$

We now have the algorithm (compare Paige [25]) :

1) Choose v_1 and v^*_1 arbitrarily, but such that $v_1^{*T} \, v_1 = S_1 \; (= \pm 1)$. Compute $u_1 = A \, v_1$ and $u^*_1 = A^T \, v^*_1$.

2) For $j=1,2,\ldots,k$, compute

$$\alpha_j = S_j v_j^{*T} A v_j \qquad (A1)$$

or

$$\alpha_j = S_j v_j^{*T} u_j \qquad (A2)$$

$$w_j = u_j - \alpha_j v_j \qquad (A3)$$

$$w_j^* = u_j^* - \alpha_j v_j^* \qquad (A4)$$

$$\phi_{j+1} = w_j^{*T} w_j \qquad (A5)$$

$$S_{j+1} = \mathrm{sign}(\phi_{j+1}) \qquad (A6)$$

$$\gamma_{j+1} = (|\phi_{j+1}|)^{\frac{1}{2}} \qquad (A7)$$

$$v_{j+1} = w_j / \gamma_{j+1} \qquad (A8)$$

$$v_{j+1}^* = w_j^* / \gamma_{j+1} \qquad (A9)$$

$$\beta_{j+1} = S_j v_j^{*T} A v_{j+1} \qquad (A10)$$

or

$$\beta_{j+1} = S_j S_{j+1} \gamma_{j+1} \qquad (A11)$$

$$\beta_{j+1}^* = S_j v_j^T A v_{j+1}^* \qquad (A12)$$

or

$$\beta_{j+1}^* = S_j S_{j+1} \gamma_{j+1} \qquad (A13)$$

$$u_{j+1} = A v_{j+1} - \beta_{j+1} v_j \qquad (A14)$$

$$u_{j+1}^* = A^T v_{j+1}^* - \beta_{j+1}^* v_j^* \qquad (A15)$$

The choices lie between (1) and (2) , (10) and (11) and (12) and (13). Denote these 8 algorithms by A(i,j,l) ,i=1 or 2, j=10 or 11 , l=12 or 13. Although theoretically identical these algorithms differ vastly computationally.

Using the results of Wilkinson [34] and assuming that $\|A\| = 1$ the equivalents of (A1) – (A15) in the presence of rounding errors are (where ε denotes the rounding error of the machine used) :

$$\alpha_j = S_j v_j^{*T} A v_j + O(\varepsilon) \qquad (R1)$$

or

$$\alpha_j = S_j v_j^{*T} u_j + O(\varepsilon) \qquad (R2)$$

$$w_j = u_j - \alpha_j v_j + O(\varepsilon) \qquad (R3)$$

$$w_j^* = u_j^* - \alpha_j v_j^* + O(\varepsilon) \qquad (R4)$$

$$\phi_{j+1} = w_j^{*T} w_j + O(\varepsilon) \qquad (R5)$$

$$S_{j+1} = \mathrm{sign}(\phi_{j+1}) \qquad (R6)$$

$$\gamma_{j+1} = [1+O(\varepsilon)] \; (|\phi_{j+1}|)^{\frac{1}{2}} \qquad (R7)$$

$$v_{j+1} = w_j/\gamma_{j+1} + O(\varepsilon) \qquad (R8)$$

$$v^*_{j+1} = w^*_j/\gamma_{j+1} + O(\varepsilon) \qquad (R9)$$

$$\beta_{j+1} = S_j \, v_j^{*T} A \, v_{j+1} + O(\varepsilon) \qquad (R10)$$

or $$\beta_{j+1} = S_j \, S_{j+1} \, \gamma_{j+1} \qquad (R11)$$

$$\beta^*_{j+1} = S_j \, v_j^T A^T v^*_{j+1} + O(\varepsilon) \qquad (R12)$$

or $$\beta^*_{j+1} = S_j \, S_{j+1} \, \gamma_{j+1} \qquad (R13)$$

$$u_{j+1} = A \, v_{j+1} - \beta_{j+1} \, v_j + O(\varepsilon) \qquad (R14)$$

$$u^*_{j+1} = A^T v^*_{j+1} - \beta^*_{j+1} \, v^*_j + O(\varepsilon) \qquad (R15)$$

Factors, such as n, have been omitted here for the sake of simplicity.

1.3 Loss of orthogonality occurs when either (or both) of w_j and w^*_j are small, in which case, as a result of cancellation γ_{j+1} will be small in (R7) and the O(ε) errors in w_j and w^*_j will be greatly magnified in (R8) and (R9) causing v^*_{j+1} and v_{j+1} to be very different from the expected vectors. This loss of orthogonality is simply unavoidable in any of the algorithms. However, even in this case, as in the symmetric, some noteworthy results involving $v_j^{*T} v_{j+1}$ still hold. In particular, from (R1), (R3), (R14) and (R1), (R4), (R15) and (R8) and (R9) it follows that

$$\gamma_{j+1} \, v_{j+1} = A \, v_j - \alpha_j \, v_j - \beta_j \, v_{j-1} + O(\varepsilon) \qquad (18)$$

and $$\gamma_{j+1} \, v^*_{j+1} = A^T v^*_j - \alpha_j \, v^*_j - \beta^*_j \, v^*_{j-1} + O(\varepsilon) \qquad (19)$$

so that $$\gamma_{j+1} \, v_j^{*T} v_{j+1} = -\beta_j \, v_j^{*T} v_{j-1} + O(\varepsilon) \qquad (20)$$

and also $$\gamma_{j+1} \, v_j^T v^*_{j+1} = -\beta^*_j \, v_j^T v^*_{j-1} \quad . \qquad (21)$$

It follows easily, then, that

$$\gamma_{j+1} v_j^{*T} v_{j+1} = \sum_{r=2}^{j} \frac{\beta_j \beta^*_{j-1} \beta_{j-2} \beta^*_{j-3} \cdots\cdots \beta_r^{(*)}}{\gamma_j \gamma_{j-1} \gamma_{j-2} \gamma_{j-3} \cdots\cdots \gamma_r} + O(\varepsilon) \tag{22}$$

and also that

$$\gamma_{j+1} v_j^{T} v^*_{j+1} = \sum_{r=2}^{j} \frac{\beta^*_j \beta_{j-1} \beta^*_{j-2} \beta_{j-3} \cdots \beta_r^{(*)}}{\gamma_j \gamma_{j-1} \gamma_{j-2} \gamma_{j-3} \cdots \gamma_r} + O(\varepsilon) \tag{23}$$

The symbol (*) indicates that an asterisk is to be inserted on β_r iff $j+1-r$ is even in (22) and odd in (23).

Using (R2), (R3) and also (R2), (R4) we have that

$$\gamma_{j+1} v_{j+1} = u_j - \alpha_j v_j + O(\varepsilon) \tag{24}$$

and $$\gamma_{j+1} v^*_{j+1} = u^*_j - \alpha_j v^*_j + O(\varepsilon) \tag{25}$$

so that $$\gamma_{j+1} v_j^{*T} v_{j+1} = O(\varepsilon) \tag{26}$$

and $$\gamma_{j+1} v_j^{T} v^*_{j+1} = O(\varepsilon) \ . \tag{27}$$

(26) and (27) hold for both $A(2,10,12)$ and $A(2,11,13)$. The algorithm $A(1,11,13)$ results in (22) and (23) having the form of (26, and (27) respectively, as here $\beta_j = \beta^*_{j_*} = S_{j-1} S_j \gamma_j$. Hence, if $\gamma_{j+1} = O(1)$ here, the orthogonality of v_j and v_{j+1}, and also of v_j and v^*_{j+1} is quite satisfactory, regardless of any previous cancellation. The algorithm $A(1,10,12)$ is not as satisfactory.

If we assume that everything up to and including Av_{j-1}, $A^T v^*_{j-1}$, α_{j-1}, β_{j-1}, β^*_{j-1} is known exactly, then rounding errors occur in the subtractions (R3), (R4) and (R14), (R15). Let $\bar{\gamma}_j$, $\bar{v}^*_j$ $\bar{v}_j$, $\bar{\beta}^*_j$, $\bar{\beta}_j$ represent the computed values and γ_j, v^*_j, v_j, β^*_j, β_j the exact values. Then

$$\begin{aligned} \bar{\gamma}_j \bar{v}_j &= A v_{j-1} - \alpha_{j-1} v_{j-1} - \beta_{j-1} v_{j-2} + O(\varepsilon) = \gamma_j v_j + O(\varepsilon) \\ \bar{\gamma}_j \bar{v}^*_j &= A^T v^*_{j-1} - \alpha_{j-1} v^*_{j-1} - \beta^*_{j-1} v^*_{j-2} + O(\varepsilon) = \gamma_j v^*_j + O(\varepsilon) \ . \end{aligned} \tag{28}$$

If no errors occur in computing

$$\bar{\beta}_j = s_{j-1}\, v_{j-1}^{*T}\, A\, \bar{v}_j$$

and $\bar{\beta}_j^* = s_{j-1}\, v_{j-1}^T\, A^T\, \bar{v}_j^*$

then $\bar{\gamma}_j \bar{\beta}_j = \gamma_j \beta_j + O(\varepsilon) = s_{j-1}\, s_j \gamma_j^2 + O(\varepsilon)$

$$(29)$$

and $\bar{\gamma}_j \bar{\beta}_j^* = \gamma_j \beta_j^* + O(\varepsilon) = s_{j-1}\, s_j \gamma_j^2 + O(\varepsilon)$.

Since $\bar{\gamma}_j^2 = \gamma_j^2 + \gamma_j\, O(\varepsilon) + O(\varepsilon^2)$ it follows that

$$\frac{\bar{\beta}_j}{\bar{\gamma}_j} = \frac{s_j\, s_{j-1} \gamma_j^2 + O(\varepsilon)}{\gamma_j^2 + \gamma_j\, O(\varepsilon) + O(\varepsilon^2)} = \frac{s_j\, s_{j-1} + O(\varepsilon)/\gamma_j^2}{1 + O(\varepsilon)/\gamma_j + O(\varepsilon^2)/\gamma_j^2} \qquad (30)$$

and

$$\frac{\bar{\beta}_j^*}{\bar{\gamma}_j} = \frac{s_j\, s_{j-1} + O(\varepsilon)/\gamma_j^2}{1 + O(\varepsilon)/\gamma_j + O(\varepsilon^2)/\gamma_j^2} \quad . \qquad (31)$$

If $\gamma_j < O(\varepsilon)$ the algorithm still performs satisfactorily, but if $\gamma_j^2 \ll O(\varepsilon)$ the numerator in each of (30) and (31) could be far greater than 1 and the bounds (22) and (23) are unsatisfactory.

The algorithms A(.,11,12) and A(.,10,13) produce obvious combinations of the above results. For example, using A(.,11,12) we see that (26) and (27) apply. These two algorithms therefore have the same shortcomings as A(.,10,12).

Returning to A(.,10,12), the factors (30) and (31) appear in all subsequent expressions (22) and (23) respectively for orthogonality, hence, once orthogonality has been lost it is unlikely that it will be recovered.

1.4 Notice that the tridiagonal matrix obtained in the above manner, viz.

$$\begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \gamma_2 & \alpha_2 & \beta_3 & & \\ & \gamma_3 & \alpha_3 & & \\ & & \cdots\cdots\cdots & & \\ & & & \gamma_{k-1} & \alpha_k \end{bmatrix} ,$$

has already been equilibrated (or balanced) in the sense of Wilkinson ([35] , chap. 6 section 10) and Parlett and Reinsch (in Wilkinson and Reinsch [36] , Contribution II/11), since the p-norm of any row of this matrix is identical to the same norm of the corresponding column (as $|\gamma_r| = |\beta_r|$, r=2,...,k-1).

If the matrix A has real eigenvalues only, then, commencing with suitable v_1 and v_1^* , a symmetric T_k will be obtained - as in example 2.

1.5 Wilkinson [35] describes the usual Lanczos method as a special case of the Generalized Hessenberg process, which may be described by :-

$$\begin{aligned} \gamma_{r+1} v_{r+1} &= w_r = A v_r - \sum_{i=1}^{r} h_{ir} v_r \\ \gamma^*_{r+1} v^*_{r+1} &= w^*_r = A^T v^*_r - \sum_{i=1}^{r} h^*_{ir} v^*_r \end{aligned} \qquad (32)$$

where the h_{ir} and h^*_{ir} are chosen so that v_{r+1} is orthogonal to $v^*_1, \ldots\ldots, v^*_r$ and v^*_{r+1} is orthogonal to $v_1, \ldots\ldots, v_r$ respectively. Applying the algorithm exactly he shows that

$$h_{ir} = h^*_{ir} = 0 \quad (i=1,\ldots r-2), \qquad (33)$$

from which it is easily inferred that if Av_r is orthogonalized with respect to v^*_{r-1} and v^*_r it is automatically orthogonal with respect to all earlier v^*_i - similarly for $A^T v^*_r$. Further

$$h_{rr} = h^*_{rr}$$

$$h_{r,r+1} \gamma_{r+1} = h^*_{r,r+1} \gamma^*_{r+1} \quad .$$

In our work $\gamma^*_{r+1} = \gamma_{r+1}$. The notation may be simplified in view of (33) so that the algorithm reads

$$\begin{aligned} \gamma_{r+1} v_{r+1} &= A v_r - \alpha_r v_r - \beta_r v_{r-1} \\ \gamma_{r+1} v^*_{r+1} &= A^T v^*_r - \alpha_r v^*_r - \beta^*_r v^*_{r-1} \quad . \end{aligned} \qquad (34)$$

Gregory [11] pointed out that in the application of this to large matrices the factors h_{ir} and h^{*}_{ir} $(i=1,\ldots,r-2)$ are not exactly zero as the biorthogonality of the vectors is not preserved. In fact, Gregory advocates the computation of the exact values of these h_{ir}, h^{*}_{ir} and their subsequent retention in the recursion relationship (32). We now establish some interesting relationships for the h_{ir}, $i=1,\ldots,r-2$.

Assume that all computations are performed exactly in the first r-1 steps of the algorithm.

Noting that in the context of this work

$$h_{ir} = v_i^{*T} A\, v_r \qquad (35)$$

it is easily established from our algorithm that if rounding occurs during the execution of the r-th step:

$$\begin{aligned} h_{ir} = {} & \frac{\gamma_{i+1}}{\gamma_r} v_{i+1}^{*T} A\, v_{r-1} - \frac{\gamma_{i+1}}{\gamma_r} \beta_{r-1} v_{i+1}^{*T} v_{r-2} - \frac{\gamma_{i+1}}{\gamma_r} \alpha_{r-1} v_{i+1}^{*T} v_{r-1} + \\ & + \frac{\alpha_i}{\gamma_r} v_i^{*T} A\, v_{r-1} - \frac{\alpha_i}{\gamma_r} \beta_{r-1} v_i^{*T} v_{r-2} - \frac{\alpha_i}{\gamma_r} \alpha_{r-1} v_i^{*T} v_{r-1} + \\ & + \frac{\beta_i^{*}}{\gamma_r} v_{i-1}^{*T} A\, v_{r-1} - \frac{\beta_i^{*}}{\gamma_r} \beta_{r-1} v_{i-1}^{*T} v_{r-2} - \frac{\beta_i^{*}}{\gamma_r} \alpha_{r-1} v_{i-1}^{*T} v_{r-1} + \\ & + \frac{0(\varepsilon)}{\gamma_r} \end{aligned} \qquad (36)$$

for $i = 1,2,\ldots,r-2$. All vectors with negative subscripts are to be taken as null vectors.

For i=1,...,r-4 this gives

$$h_{ir} = \frac{\gamma_{i+1}}{\gamma_r} h_{i+1,r-1} + \frac{\alpha_i}{\beta_r} h_{i,r-1} + \frac{\beta^*_i}{\gamma_r} h_{i-1,r-1} -$$

$$- \frac{\gamma_{i+1}}{\gamma_r} \beta_{r-1} v^{*T}_{i+1} v_{r-2} - \frac{\gamma_{i+1}}{\gamma_r} \alpha_{r-1} v^{*T}_{i+1} v_{r-1} - \frac{\alpha_i}{\gamma_r} \beta_{r-1} v^{*T}_i v_{r-2} -$$

$$- \frac{\alpha_i}{\gamma_r} \alpha_{r-1} v^{*T}_i v_{r-1} - \frac{\beta^*_i}{\gamma_r} \beta_{r-1} v^{*T}_{i-1} v_{r-2} - \frac{\beta^*_i}{\gamma_r} \alpha_{r-1} v^{*T}_{i-1} v_{r-1} +$$

$$+ \frac{O(\varepsilon)}{\gamma_r} \; . \qquad (37)$$

When i=r-3 (36) is

$$h_{r-3,r} = \frac{\gamma_{r-2}}{\gamma_r} v^{*T}_{r-2} A v_{r-1} + \frac{\alpha_{r-3}}{\gamma_r} v^{*T}_{r-3} A v_{r-1} + \frac{\beta^*_{r-3}}{\gamma_r} v^{*T}_{r-4} A v_{r-1} -$$

$$- \frac{\gamma_{r-2}}{\gamma_r} \beta_{r-1} v^{*T}_{r-2} v_{r-2} - \frac{\gamma_{r-2}}{\gamma_r} \alpha_{r-1} v^{*T}_{r-2} v_{r-1} \quad -$$

$$- \frac{\alpha_{r-3}}{\gamma_r} \beta_{r-1} v^{*T}_{r-3} v_{r-2} - \frac{\alpha_{r-3}}{\gamma_r} \alpha_{r-1} v^{*T}_{r-3} v_{r-1} - \frac{\beta^*_{r-3}}{\gamma_r} \beta_{r-1} v^{*T}_{r-4} v_{r-2} -$$

$$- \frac{\beta^*_{r-3}}{\gamma_r} \alpha_{r-1} v^{*T}_{r-4} v_{r-1} + \frac{O(\varepsilon)}{\gamma_r} \; . \qquad (38)$$

With i=r-2, (36) is

$$h_{r-2,r} = \frac{\gamma_{r-1}}{\gamma_r} v^{*T}_{r-1} A v_{r-1} + \frac{\alpha_{r-2}}{\gamma_r} v^{*T}_{r-2} A v_{r-1} + \frac{\beta_{r-2}}{\gamma_r} v^{*T}_{r-3} A v_{r-1} -$$

$$-\frac{\gamma_{r-1}\beta_{r-1}}{\gamma_r} v^{*T}_{r-1} v_{r-2} - \frac{\gamma_{r-1}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-1} v_{r-1} - \frac{\alpha_{r-2}\beta_{r-1}}{\gamma_r} v^{*T}_{r-2} v_{r-2} -$$

$$-\frac{\alpha_{r-2}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-2} v_{r-1} - \frac{\beta_{r-2}\beta_{r-1}}{\gamma_r} v^{*T}_{r-3} v_{r-2} - \frac{\beta^*_{r-2}}{\gamma_r} v^{*T}_{r-3} v_{r-1} +$$

$$+\frac{O(\varepsilon)}{\gamma_r} . \qquad (39)$$

Assume now that β_{r-1} and α_{r-1} have been computed exactly, i.e. that $\beta_{r-1} = S_{r-2} v^{*T}_{r-2} A v_{r-1}$ and $\alpha_{r-1} = S_{r-1} v^{*T}_{r-1} A v_{r-1}$. Under these circumstances (38) and (39) are, respectively

$$h_{r-3,r} = \frac{\alpha_{r-3}}{\gamma_r} h_{r-3,r-1} + \frac{\beta^*_{r-3}}{\gamma_r} h_{r-4,r-1} - \frac{\gamma_{r-2}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-2} v_{r-1} -$$

$$-\frac{\alpha_{r-3}\beta_{r-1}}{\gamma_r} v^{*T}_{r-3} v_{r-2} - \frac{\alpha_{r-3}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-3} v_{r-1} - \frac{\beta^*_{r-3}\beta_{r-1}}{\gamma_r} v^{*T}_{r-4} v_{r-2}$$

$$-\frac{\beta^*_{r-3}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-4} v_{r-1} + \frac{O(\varepsilon)}{\gamma_r} \qquad (40)$$

and $h_{r-2,r} = \frac{\beta_{r-2}}{\gamma_r} h_{r-3,r-1} - \frac{\gamma_{r-1}\beta_{r-1}}{\gamma_r} v^{*T}_{r-1} v_{r-2} - \frac{\alpha_{r-2}\alpha_{r-1}}{\gamma_r} v^{*T}_{r-2} v_{r-1} -$

$$-\frac{\beta_{r-2}\beta_{r-1}}{\gamma_r} v^{*T}_{r-3} v_{r-2} - \frac{\beta^*_{r-2}}{\gamma_r} v^{*T}_{r-3} v_{r-1} + \frac{O(\varepsilon)}{\gamma_r} . \qquad (41)$$

This semi-ideal state therefore leads to

$$h_{ir} = \frac{O(\mathcal{E})}{\gamma_r} \quad , \; i=1,\ldots,r-2 \; . \qquad (42).$$

This result is not valid when either a catastrophic deterioration in the biorthogonality occurs or when "previous" h_{ir}'s are small. Again, the danger of a small γ is highlighted.

In the examples of 1.7 the values of h_{ir}, $i=1,\ldots,r-2$ were actually computed.

Several authors strongly recommend intermediate reorthogonalization of the theoretically biorthogonal sets of vectors (e.g. Wilkinson [35]). This work has not convinced us of the need for reorthogonalization in this particular algorithm.

1.6 It is interesting to obtain results analogous to (22) and (23) and (26) and (27) when the vectors are reorthogonalized in our algorithms. In the presence of rounding the algorithms may be formulated as :-

1) Choose v_1 and v_1^* arbitrarily, but such that $v_1^{*T} v_1 = S_1$ (= ±1).

Compute $u_1 = A\, v_1$ and $u_1^* = A^T v_1$.

2) For $j=1,2,\ldots,k$ compute

$$\alpha_j = S_j\, v_j^{*T} A\, v_j + O(\varepsilon) \qquad \text{(RR1)}$$

or $$\alpha_j = S_j\, v_j^{*T} u_j + O(\varepsilon) \qquad \text{(RR2)}$$

$$\bar{w}_j = u_j - \alpha_j\, v_j + O(\varepsilon) \qquad \text{(RR3)}$$

$$\bar{w}_j^* = u_j^* - \alpha_j\, v_j + O(\varepsilon) \qquad \text{(RR4)}$$

$$w_j = \bar{w}_j - \sum_{i=1}^{j} e_{ji}\, v_i + O(\varepsilon) \qquad \text{(RR5)}$$

$$w_j^* = \bar{w}_j^* - \sum_{i=1}^{j} e_{ji}^*\, v_i^* + O(\varepsilon) \qquad \text{(RR6)}$$

where $$e_{ji} = v_i^{*T} w_j \qquad \text{(RR7)}$$

and $$e_{ji}^* = v_i^T w_j^* \qquad \text{(RR8)}$$

$$\phi_{j+1} = w_j^{*T} w_j + O(\varepsilon) \qquad \text{(RR9)}$$

$$S_{j+1} = \text{sign}\ (\phi_{j+1}) \qquad \text{(RR10)}$$

$$\gamma_{j+1} = (\ 1 + O(\varepsilon)\ )\ |\phi_{j+1}|^{1/2} \qquad \text{(RR11)}$$

$$v_{j+1} = w_j / \gamma_{j+1} \ + \ O(\varepsilon) \qquad \text{(RR12)}$$

$$v^*_{j+1} \equiv w^*_j / \gamma_{j+1} \ + \ O(\varepsilon) \qquad \text{(RR13)}$$

$$\beta_{j+1} = S_j\ v_j^{*T}\ A\ v_{j+1} + \ O(\varepsilon) \qquad \text{(RR14)}$$

$$\text{or } \beta_{j+1} = S_j\ S_{j+1}\ \gamma_{j+1} \qquad \text{(RR15)}$$

$$\beta^*_{j+1} = S_j\ v_j^T\ A^T\ v^*_{j+1} \ + \ O(\varepsilon) \qquad \text{(RR16)}$$

$$\text{or } \beta^*_{j+1} = S_j\ S_{j+1}\ \gamma_{j+1} \qquad \text{(RR17)}$$

$$u_{j+1} = A\ v_{j+1} - \beta_{j+1}\ v_j + O(\varepsilon) \qquad \text{(RR18)}$$

$$u^*_{j+1} = A^T\ v^*_{j+1} - \beta^*_{j+1}\ v^*_j + O(\varepsilon)\ . \qquad \text{(RR19)}$$

As before, factors such as n have been omitted for simplicity.

Using (RR1), (RR3), (RR18) and (RR1), (RR4), (RR19) leads to

$$\gamma_{j+1}\ v_j^T\ v^*_{j+1} = \sum_{r=2}^{j} \frac{\beta_j\ \beta^*_{j-1}\ \beta_{j-2}\ \beta^*_{j-3}\ \ldots\ldots \beta_r^{(*)}}{\gamma_j\ \gamma_{j-1}\ \gamma_{j-2}\ \gamma_{j-3}\ \ldots\ldots\ \gamma_r}\ O(\varepsilon) \quad +$$

$$+ \sum_{r=2}^{j} (\pm) \frac{\beta_j\ \beta^*_{j-1}\ \ldots\ldots \beta_r^{(*)}}{\gamma_j\ \gamma_{j-1} \ldots\ldots \gamma_r} \sum_{i=1}^{r} e_{ri}\ v_r^{*T} v_i \ - \sum_{i=1}^{j} e_{ji}\ v_j^{*T} v_i + O(\varepsilon) \qquad (43)$$

and

$$\gamma_{j+1}\ v_j^{*T}\ v_{j+1} = \sum_{r=2}^{j} \frac{\beta^*_j\ \beta_{j-1}\ \beta^*_{j-2}\ \beta_{j-3}\ \ldots \beta_r^{(*)}}{\gamma_j\ \gamma_{j-1}\ \gamma_{j-2}\ \gamma_{j-3}\ \ldots \gamma_r}\ O(\varepsilon) \ +$$

$$+ \sum_{r=2}^{j} (\pm) \frac{\beta^*_j\ \beta_{j-1} \ldots \beta_r^{(*)}}{\gamma_j\ \gamma_{j-1} \ldots \gamma_r} \sum_{i=1}^{r} e_{ri}\ v_j^T\ v^*_i \ - \sum_{i=1}^{j} e_{ji} v_j^T v^*_i \ + O(\varepsilon) \qquad (44)$$

The symbol (*) indicates that an asterisk is to be attached to β_r iff j+1-r is even in (43) and odd in (44).

Using (RR2), (RR3) and (RR2), (RR4) leads to the more satisfactory result

$$\gamma_{j+1} \; v_j^{*T} \; v_{j+1} = -\sum_{i=1}^{j} e_{ji} \; v_j^{*T} \; v_i + O(\varepsilon) \qquad (45)$$

and also to

$$\gamma_{j+1} \; v_j^T \; v_{j+1}^* = -\sum_{i=1}^{j} e_{ji}^* \; v_j^T \; v_i^* + O(\varepsilon) \; . \qquad (46)$$

The results (29), (30) and (31) still hold here, as do the comments following them.

The algorithm (RR1), (RR3), (RR4), (RR18) and (RR19) can produce surprising results even when all the γ's are O(1). We have found that the algorithm (RR2), (RR3), (RR4) can be less satisfactory than the non-reorthogonalized form when there is a serious deterioration in the bi-orthogonality. In fact an improvement in the bi-orthogonality (and incidentally in the upper hessenberg form) was rare when intermediate reorthogonalization was used.

In computing the examples of 1.7 we computed the $v_i^* \; v_j$'s and, where appropriate, have tabulated these.

1.7 The real eigenvalues of several matrices were approximated in order to illustrate various features of the algorithm A(2,11,13). The features illustrated are :- a suitable set of initial vectors leads in the case of a non-symmetric matrix with real roots to a symmetric tridiagonal matrix; situations where roots are obtained using our method without regular reorthogonalization while these roots could not be found when reorthogonalization was used; the extreme roots being determined after fewer than n Lanczos steps (for an nxn matrix), leading consequently to a tridiagonal matrix of order less than n; ill-conditioned roots being obtained when more than n steps were applied and not otherwise.

The computed eigenvalues were found using both reorthogonalization and also without any subsequent reorthogonalization, both of these processes on several different initial vectors - the eigenvalues obtained from these procedures have been tabulated. The complete upper heesenberg form obtained by applying the Lanczos algorithm to each of the matrices has sometimes been tabulated - see (32). Where appropriate, some of the values of $v_i^{*T} v_j$ have been tabulated in order to give an indication of whether the resulting vectors are adequatly biorthogonal. In the tables (only) any fixed point integer entry (n) is to be understood as 10^n.

<u>Example 1</u> : (Wilkinson [35] p 392)

$$\begin{bmatrix} 4 & 1 & 3 & 2 \\ 2 & 1 & 2 & 5 \\ 1 & 3 & 3 & 4 \\ 4 & 1 & 2 & 1 \end{bmatrix}$$

The real eigenvalues of this matrix were found using the sets of initial vectors

$$v_{1(p)}^{T} = \frac{\sqrt{10^{p}}}{4} \left[1 , 2 \times 10^{-p} , 0 , 0 \right]$$

and

$$v_{1(p)}^{*T} = \frac{\sqrt{10^{p}}}{4} \left[2 \times 10^{-p} , 1 , 0 , 0 \right] ,$$

$p=0(1)8.$

We tabulate the results below (reorthogonalization being used in the first table and not the second) :-

p	0	1	2	3	4	5
1	-1.362444	-1.362444	-1:362444	-1.362444	-1.362444	-1.362444
2	9.703378	9.703378	9.703378	9.703378	9.703378	9.703378

p	6	7	8
1	?	?	?
2	9.730741	9.639500	9.562491

<u>table 1</u>

p	0	1	2	3	4	5
1	-1.362444	-1.362444	-1.362444	-1.362444	-1.362444	-1.362444
2	9.703378	9.703378	9.703378	9.703378	9.703378	9.703378

p	6	7	8
1	-1.362444	-1.359950	-1.519
2	9.703375	9.704292	9.528485

<u>table 2</u>

A question mark indicates that the relevant root could not be found.

Upper hessenberg form - without reorthogonalization

$$\begin{bmatrix} 4.000 & 5.431 & -14 & -14 \\ 5.431 & 4.458 & 8.747 & -14 \\ & 8.747 & -.363 & -1.782 \\ & & 1.782 & .906 \end{bmatrix} \qquad \begin{bmatrix} 7.600 & 6.735 & -14 & -13 \\ 6.735 & -5.102 & -6.939 & -13 \\ & 6.939 & 3.217 & -2.038 \\ & & 2.038 & 3.286 \end{bmatrix}$$

$p=0$ $\qquad$ $p=1$

$$\begin{bmatrix} 52.51 & -44.00 & -12 & -11 \\ 44.00 & -35.79 & -6.022 & -11 \\ & 6.022 & -11.44 & .923 \\ & & .923 & 1.725 \end{bmatrix} \qquad \begin{bmatrix} 502.5 & -494.5 & -10 & -9 \\ 494.5 & -486.3 & -1.521 & -9 \\ & 1.521 & -8.841 & 1.103 \\ & & 1.103 & 1.629 \end{bmatrix}$$

$p=2$ $\qquad$ $p=3$

$$\begin{bmatrix} 5002. & -4994. & -7 & -6 \\ 4994. & -4986. & -.4717 & -6 \\ & .4717 & -8.641 & 1.021 \\ & & 1.021 & 1.620 \end{bmatrix} \qquad \begin{bmatrix} 5(4) & -5(4) & -5 & -4 \\ 5(4) & -5(4) & -.1489 & -4 \\ & .1489 & -8.621 & 1.022 \\ & & 1.022 & 1.619 \end{bmatrix}$$

$p=4$ $\qquad$ $p=5$ ≠

$$\begin{bmatrix} 5(5) & -5(5) & -3 & -1 \\ -5(5) & -5(5) & -.0478 & -1 \\ & -.0478 & -8.619 & 1.022 \\ & & 1.022 & 1.619 \end{bmatrix} \qquad \begin{bmatrix} 5(7) & -5(6) & -1 & 1 \\ 5(6) & -5(6) & -.01489 & 1 \\ & .01489 & -8.619 & 1.022 \\ & & 1.022 & 1.619 \end{bmatrix}$$

$p=6$ $\qquad$ $p=7$

$$\begin{bmatrix} 5(7) & -5(7) & 3 & 4 \\ 5(7) & -5(7) & -.004707 & 4 \\ & .004707 & -8.618 & 1.044 \\ & & 1.044 & 2.167 \end{bmatrix}$$

$p=8$

table 3

≠ - a(n) means a x 10^n.

Upper hessenberg form - with reorthogonalisation

$$\begin{bmatrix} 4.000 & 5.431 & -15 & -16 \\ 5.431 & 4.458 & 8.747 & 0.0000 \\ & 8.747 & -3.632 & -1.782 \\ & & 1.782 & .906 \end{bmatrix} \qquad \begin{bmatrix} 7.6000 & 6.733 & -15 & -15 \\ 6.733 & -5.102 & -6.939 & -14 \\ & 6.939 & 3.217 & -2.038 \\ & & 2.038 & 3.286 \end{bmatrix}$$

p=0 p=1

$$\begin{bmatrix} 52.51 & -44.00 & -12 & -11 \\ 44.00 & -33.79 & -6.022 & -11 \\ & 6.022 & -11.44 & .923 \\ & & .923 & 1.725 \end{bmatrix} \qquad \begin{bmatrix} 502.5 & -494.5 & -10 & -7 \\ 494.5 & -486.3 & -1.521 & -7 \\ & 1.521 & -8.841 & 1.013 \\ & & 1.021 & 1.620 \end{bmatrix}$$

p=2 p=3

$$\begin{bmatrix} 5003. & -4994. & -8 & -4 \\ 4994. & -4986. & -.4717 & -5 \\ & .4717 & -8.641 & 1.021 \\ & & 1.021 & 1.620 \end{bmatrix} \qquad \begin{bmatrix} 5(4) & -5(4) & -7 & -2 \\ -5(4) & -5(4) & -.1489 & -2 \\ & -.1489 & -8.621 & 1.022 \\ & & 1.022 & 1.619 \end{bmatrix}$$

p=4 p=5

$$\begin{bmatrix} 5(5) & -5(5) & -2 & 3 \\ 5(5) & -5(5) & -.04708 & 3 \\ & .04708 & -8,619 & 1.022 \\ & & 1.022 & 4.072 \end{bmatrix} \qquad \begin{bmatrix} 5(6) & -5(6) & 0 & 7 \\ 5(6) & -5(6) & -.01489 & 7 \\ & .01489 & -8.619 & -.9522 \\ & & .9522 & 8 \end{bmatrix}$$

p=6 p=7

$$\begin{bmatrix} 5(7) & -5(7) & 3 & 8 \\ 5(7) & -5(7) & -.004707 & 8 \\ & .004707 & -8.619 & -252.0 \\ & & 252.0 & -5(7) \end{bmatrix}$$

p=8

table 4

For p=0(1)4 the $v_i^{*T} v_j$'s $(i \neq j)$ are all less than $.26 \times 10^{-7}$ in absolute value, for p≥5 the situation is not as favourable. In table 5 we tabulate the orders of magnitude of the $v_i^{*T} v_j$'s for $p \geq 1$.

		1	2	3	4	5	6	7	8
NON REORTHO	max $v_i^{*T} v_j$	-14	-12	-10	-7	-5	-3	0	3
	max $v_i^{*T} v_{j+1}$	-15	-14	-13	-11	-10	-9	-8	-5
REORTHO	max $v_i^{*T} v_j$	-15	-12	-9	-7	-5	-2	1	1
	max $v_i^{*T} v_{j+1}$	-15	-14	-12	-11	-10	-8	-6	-5

table 5

<u>Example 2</u> : $A = (a_{ij}) \quad 1 \leq i,j \leq 20$, with $a_{ii}=i$; $a_{ik}=1$, $k=i+1(1)i+4$; and all other elements zero.

This matrix has $\prod_{i=1}^{20} (\lambda - i) = 0$ as its characteristic equation. Wilkinson ([34] pp 41-43) discusses this equation at some length. He shows that the root of greatest sensitivity is $\lambda = 15$ and also that if the coefficient of λ^{19} is perturbed by 2^{-23} then ten of the roots become complex with substantial imaginary parts. In fact, to 9D the roots of the perturbed polynomial are :-

1.000000000	6.000006944	10.095266145 ± 0.643500904 i
2.000000000	6.999697234	11.793633881 ± 1.652329728 i
3.000000000	8.007267603	13.992358137 ± 2.518830070 i
4.000000000	8.917250249	16.730737466 ± 2.812624894 i
4.999999928	20.846908101	19.502439409 ± 1.940330347 i .

The real roots of the above matrix were determined in the first place by using two different sets of initial vectors and not reorthogonalising the resultant vectors and in the second place by applying the reorthogonalization process to the vectors obtained from both sets of initial vectors.

a) <u>No reorthogonalization</u>

i) As initial vectors here we used

$v_1^T = (t_i)$; $t_i=1$ $i=1(2)19$; $t_i=0$ $i=2(2)20$

$v_1^{*T} = (t_i^*)$; $t_i^*=1$ $i=2(2)20$; $t_i^*=0$ $i=3(2)19$; $t_1^*=1$.

These vectors do not produce a symmetric tridiagonal matrix. The real roots found are as follows :-

<u>20 Lanczos steps</u> : All 20 roots were found correct to 6 decimal places.

<u>19 Lanczos steps</u> : Roots 1 through to 19 were found to 6D, while the 20-th root was not found.

<u>18 Lanczos steps</u> : The only roots obtained were 1.000001 ; 1.999780 ; 3.012135 ; 3.854078 ; 16.234479 ; 16.983875 ; 18.000175 and 18.999999.

A check on the biorthogonality of the vectors v_i and v_k^* showed that the worst case was $v_2^T v_{19}^* = -.103 \times 10^{-9}$ – an adequate result. A check on the values of the off-tridiagonal h_{ir} in the upper hessenberg form showed that all were less than 10^{-8}, except for $h_{i,20}$ which lay between .01 and 10. The accuracy of the 19-step solution occurs because $\gamma_{19} = .216 \times 10^{-9}$.

ii) The next set of initial vectors used was

$$v_1^T = (t_i) ; \qquad t_i = .01 \quad i=1(1)20$$

$$v_1^{*T} = (t_i^*) ; \qquad t_i = .5 \quad i=1(1)20 .$$

These lead to a symmetric tridiagonal matrix. We may summarise the results as follows :-

20 Lanczos steps : The 20 roots were found correct to 6 decimal places.

19 Lanczos steps : Roots 1 through to 15 and 17 to 20 were obtained correct to 6D.

18 Lanczos steps : The roots 1 to 15 were found correct to 6D. The three other roots are 17.175238 ; 18.999902 and 20.000000.

17 Lanczos steps Roots 1 to 15 and also 20 were obtained to 6D. A further root of 18.998542 was found.

16 Lanczos steps : The following roots were found :-

1.000000	2.000000	3.000014	4.000207	5.001518
6.006404	7.016258	8.025796	9.025707	10.015711
11.005770	12.000958	13.000078	14.000003	15.000000
20.000000 .				

Notice that the roots not obtained in using fewer than 20 Lanczos iterations are, roughly speaking, the ill-conditioned roots.

In all columns except the last the entries in the upper hessenberg matrix are less than 10^{-4} in modulus, while in the last they range between 10^{-6} and 10. In this case the only small γ is γ_{19}, which has the value $.283 \times 10^{-4}$.

b) Using reorthogonalization

i) The same initial vectors as in (a) (i) above were used. Although the resulting vectors were satisfactorily biorthogonal, the only real roots found after applying 20 Lanczos steps were :-

1.000002	1.999667	3.025791	5.758775	7.063477
10.000016	18.000955	18.999997 .		

As expected, the tridiagonal elements here differ vastly from those in (a) (i). The surprise is that the off-tridiagonal elements in the hessenberg form are generally larger than the corresponding elements in the previous case, they range as follows in fact :-

h_{ir}, r=3,...,14, are all less than 10^{-4} (in modulus);

$10^{-3} \geqslant h_{i,15} \geqslant 10^{-15}$;

$10^{-2} \geqslant h_{i,16} \geqslant 10^{-15}$;

$10^{-1} \geqslant h_{i,17} \geqslant 10^{-14}$;

$10 \geqslant h_{i,18} \geqslant 10^{-14}$;

$10^{2} \geqslant h_{i,19} \geqslant 10^{-13}$;

$10^{2} \geqslant h_{i,20} \geqslant 10^{-13}$.

ii) Using the initial vectors of (a)(ii) above it was found that $\gamma_{20} < 10^{-10}$ and so only 19 Lanczos steps were applied. The roots $\lambda = 1(1)15$ and $\lambda = 17(1)20$ (all correct to 6D) were found. The biorthogonality of the vectors is satisfactory, as expected.

Applying 18 steps of the algorithm the roots $\lambda = 1(1)15, \lambda = 20$, correct to 6D, $\lambda = 17.831556$ and $\lambda = 18.99574$ were found. The only unsatisfactory elements in the upper hessenberg form occur in the 20-th column, where $h_{17,20} = 10^{-1}$ and $h_{18,20} = 10^{2}$. The tridiagonal form is not symmetric.

The sections (a)(i) and b(i) above illustrate a situation in which the algorithm is superior without intermediate reorthogonalization. Two facts illustrate this : a) Without reorthogonalization we were able to find all the eigenvalues correct to 6D, while when using the intermediate reorthogonalization procedure approximations to only eight of the roots could be found (the extreme ones to 4 and 5D and the middle ones very inacurately) ;
b) Without reorthogonalizing it was found that $\max\limits_{i \neq j} v_i^T v_j^* = .936 \times 10^{-10}$ and $\max v_j^T v_{j+1}^* = .110 \times 10^{-12}$, while when reorthogonalizing $v_{20}^T v_{18}^* = 1.26$ and $v_{20}^T v_{19}^* = .443$. This catastrophic deterioration is <u>not</u> associated with a small γ as no γ is less than 2 in (b)(i).

Example 3 : Frank's matrix (Westlake [32] p 153)

$$\begin{bmatrix} 1 & 1 & & & & & & \\ 1 & 2 & 2 & & & & & \\ 1 & 2 & 3 & 3 & & & & \\ 1 & 2 & 3 & 4 & 4 & & & \\ & \multicolumn{5}{c}{\cdots\cdots\cdots\cdots} & & \\ 1 & 2 & 3 & 4 & 5 & \cdots\cdots & (n-1) & (n-1) \\ 1 & 2 & 3 & 4 & 5 & \cdots\cdots & (n-1) & n \end{bmatrix}$$

We choce $n=12$, $v_1^T = (t_i)$, $t_i = 1/\sqrt{84}$, $i=1(1)12$,

$v_1^{*T} = (t_i^*)$, $t_i^* = 7/\sqrt{84}$, $i=1(1)12$.

	No reortho.		Reortho.	
True roots	12 steps	16 steps	12 steps	16 steps
0.031028060644010		0.031030		
0.049507429185278	0.045836	0.049486		
0.081227659240405	0.081074	0.081227		
0.143646519769220	0.143644	0.143647	0.136028	0.136028
0.284749720558478	0.284750	0.284750	0.284102	0.284102
0.643505319004856	0.643505	0.643505	0.643493	0.643493
1.553988709132108	1.553989	1.553989	1.553989	1.553989
3.511855948580757	3.511856	3.511856	3.511855	3.511856
6.961533085567122	6.961533	6.961533	6.961533	6.961533
12.311077408868526	12.311077	12.311077	12.311077	12.311077
20.198988645877079	20.198989	10.198989	20.198989	20.198989
32.228891501572161	32.228892	32.228891	32.228892	32.228892

table 6

The values of the condition numbers $x_i^T y_i = s_i$ (where x_i and y_i are the right and left hand vectors associated with the eigenvalue λ_i respectively) (see Wilkinson [34]) are extremely small for the small eigenvalues and close to one for the larger ones.

The off-tridiagonal vectors in both the non-reothogonalized and orthogonalized forms vary from 10^{-15} to 10^2 in modulus.

Summary of results

Define the symmetry ratio as the number of positive β_j's, obtained after applying k steps of the Lanczos algorithm,divided by k-1.

In both the previous application of the Lanczos algorithm and in the later applications in chapter 2 we are interested only in the real eigenvalues of the given matrix. Several of the previous examples have only real roots in fact. The value of the symmetry ratio is a useful a priori indicator of the accuracy of these real eigenvalues.

1) Denote the symmetry ratio by S.R.
2) A cross in the column headed "h" indicates that some or all of the off-tridiagonal elements in the upper hessenberg form are large, while a "v" indicates that they are all small.
3) A cross in the column headed "r" indicates that the roots found are not satisfactory, while a "v" indicates the roots as satisfactory.

Example 1 : (4x4 matrix)

No reorthogonalization:-

p=	0	1	2	3	4	5	6	7	8
S.R.	0.67	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
h	v	v	v	v	v	x	x	x	x
r	v	v	v	v	v	v	v	x	x

Reothogonalization used:-

p=	0	1	2	3	4	5	6	7	8
S.R.	0.67	0.33	0.33	0.33	0.33	0.33	0.33	0.00	0.00
h	v	v	v	v	x	x	x	x	x
r	v	v	v	v	v	v	x	x	x

Example 2 : (20x20 matrix)

a) No reorthogonalization

i)

steps	20	19	18
S.R.	0.45	0.47	0.44
h	x	v	v
r	v	v	x

ii)

steps	20	19	18	17	16
S.R.	1.00	1.00	1.00	1.00	1.00
h	x	x	x	v	v
r	v	v	v	v	v

b) Reorthogonalizing

i) 20 Lanczos steps : S.R. = 0.53 ; h=x ; r=x throughout.

ii) 20 Lanczos steps : S.R. = 0.84 ; h=v except in last element of last column ; r=v.

Example 3 : (12x12 matrix)

no of steps		12	16
no reortho	S.R.	0.75	0.66
	h	x	x
	r	small roots x large roots v	v
reortho	S.R.	0.50	0.40
	h	x	x
	r	no small roots large roots v	no small roots large roots v

1.8 Conclusions

1) Reorthogonalization does not necessarily improve the algorithm in the sense of preserving biorthogonality, neither does it always assist in determining the eigenvalues more accurately.

2) Fewer than n applications of the Lanczos algorithm are sufficient to isolate the extreme roots when the symmetry ratio is comparatively large - this is particularly advantageous when dealing with large sparse matrices.

3) When the symmetry ratio is particularly low, even after n applications of the Lanczos algorithm, the eigenvalues can still be poorly determined - see chapter two for a particularly clear example.

4) Severe non-biorthogonality always has disastrous consequences, as in example 1.

5) On occasions more than n Lanczos iterations were performed - see example 3 for example. Note that in this example the smaller eigenvalues, which are badly conditioned, are improved by using more than n iterations without reorthogonalizing, while no improvement occurred when the two sets of vectors were continuously reorthogonalized. Notice too that the symmetry ratio of the former case is consistently greater than that of the latter. Paige, in applying more than 8 iterations of the symmetric Lanczos algorithm to the 8x8 Rosser matrix, found that the additional roots generated also converged to the Rosser matrix roots, [25]. We did not always find this to be the case with non-symmetric matrices.

Final conclusion : we have found a technique of some promise for isolating the real roots of large sparse non-symmetric matrices, in particular the extreme roots.

CHAPTER 2 : EIGENVALUES OF A DIFFERENTIAL EQUATION

2.1 We require the eigenvalues of

$$\nabla^2\psi + \lambda\psi = 0 \qquad (1)$$

defined on the domain Γ of figure 1.

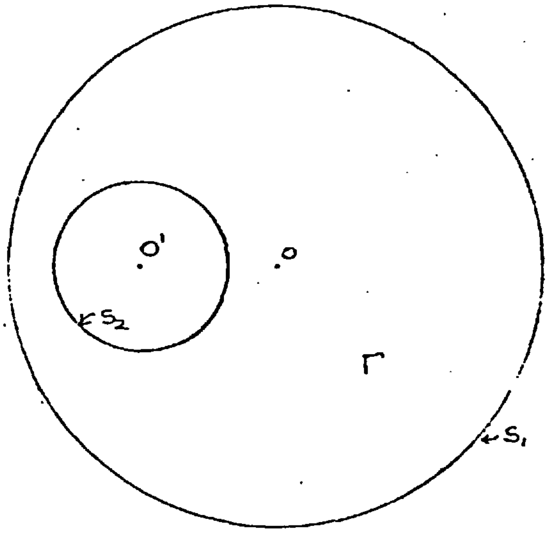

figure 1

The boundary conditions are:-

$$\psi = 0 \text{ on } S_1: x^2 + y^2 = ro^2 \qquad (2)$$

$$\frac{\partial\psi}{\partial r} + c\psi = \text{ on } S_2: (x + xc)^2 + y^2 = ri^2 \quad . \qquad (3)$$

Relocating the origin at 0', transforming to polar coordinates and making the substitution

$$\phi(r,\theta) = r^{\frac{1}{2}}\psi(r,\theta) ,$$

(1) becomes

$$\frac{\partial^2\phi}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2\phi}{\partial\theta^2} + \left(\frac{1}{4r^2} + \lambda\right)\phi = 0 \qquad (4)$$

subject to $\phi = 0$ on $r^2 - 2\,r\,xc\cos\theta = ro^2 - xc^2$ (5)

and $r\dfrac{\partial\phi}{\partial r} + (cr - \frac{1}{2})\,\phi = 0$ on $r = ri$. (6)

xc and yc are the coordinates of 0.

Using the usual finite-difference notation; defining

$$\phi_{ij} = \phi(r_j, \theta_i) ,$$

where $r_j = ri + j\Delta r$,

$\theta_i = i\Delta\theta$;

and using the simplest second-order central difference approximation

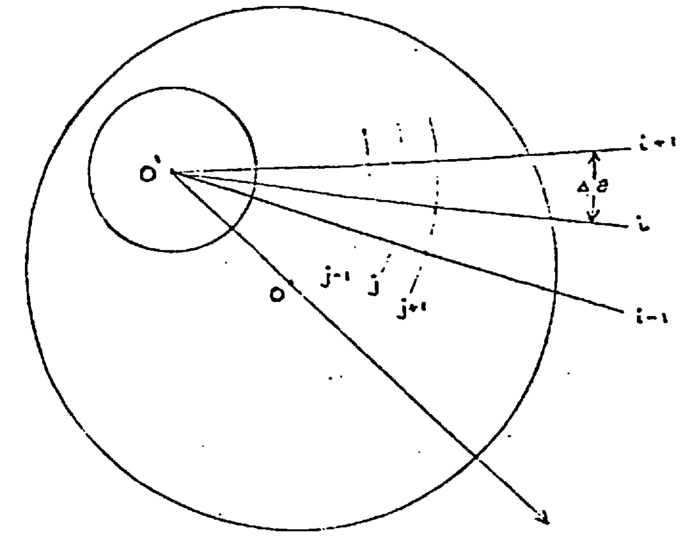

figure 2

to the partial derivatives, (4) may be approximated by

$$\frac{1}{r^2_. \Delta\theta^2}\,\phi_{i-1,j} + \frac{1}{\Delta r^2}\,\phi_{i,j-1} + \frac{1}{r^2_. \Delta\theta^2}\,\phi_{i+1,j} + \frac{1}{\Delta r^2}\,\phi_{i,j+1} +$$

$$+\left\{ \frac{-2}{\Delta r^2} - \frac{2}{r^2_.\Delta\theta^2} + \frac{1}{4r^2} + \lambda \right\}\phi_{ij} = 0 \quad , \qquad (7)$$

where i and j range over appropriate values.

Applying the usual Taylor expansion approach the truncation error in (7) is easily seen to be

$$O(\Delta r^2) + O(\frac{\Delta\theta^2}{r^2}) \quad . \qquad (8)$$

Differentiating the boundary condition (8) with respect to r gives

$$r\,\frac{\partial^2\phi}{\partial r^2} + (c.r + \tfrac{1}{2})\,\frac{\partial\phi}{\partial r} + c\,\phi = 0 \text{ on } r{=}ri \,. \qquad (9)$$

Substitute for $\frac{\partial\phi}{\partial r}$ from (6) into (9) to obtain

$$\frac{\partial^2\phi}{\partial r^2} = \left[4.c^2_. r^2 - 4.c.r - 1\right] / \,4\,r^2 \quad . \qquad (10)$$

Putting (10) into (4) yields

$$\left[\frac{c}{r}\left(c.r - 1\right) + \lambda\right]\phi + \frac{1}{r^2}\frac{\partial^2\phi}{\partial\theta^2} = 0 \quad \text{on } r=ri.$$

Again use the second-order central difference approximation to the derivative, here obtaining

$$\frac{1}{ri^2_{.}\Delta\theta^2}\,\theta_{i-1,1} + \frac{1}{ri^2\Delta\theta^2}\,\theta_{i+1,1} + \left[\frac{c}{ri}\left(c.ri - 1\right) - \frac{2}{ri^2_{.}\Delta\theta^2} + \lambda\right]\phi_{i,1} = 0. \tag{11}$$

The truncation error is $0\left(\frac{\theta^2}{ri^2}\right)$.

The situation on the other boundary is somewhat more complicated. The typical situation is illustrated in figure 3.

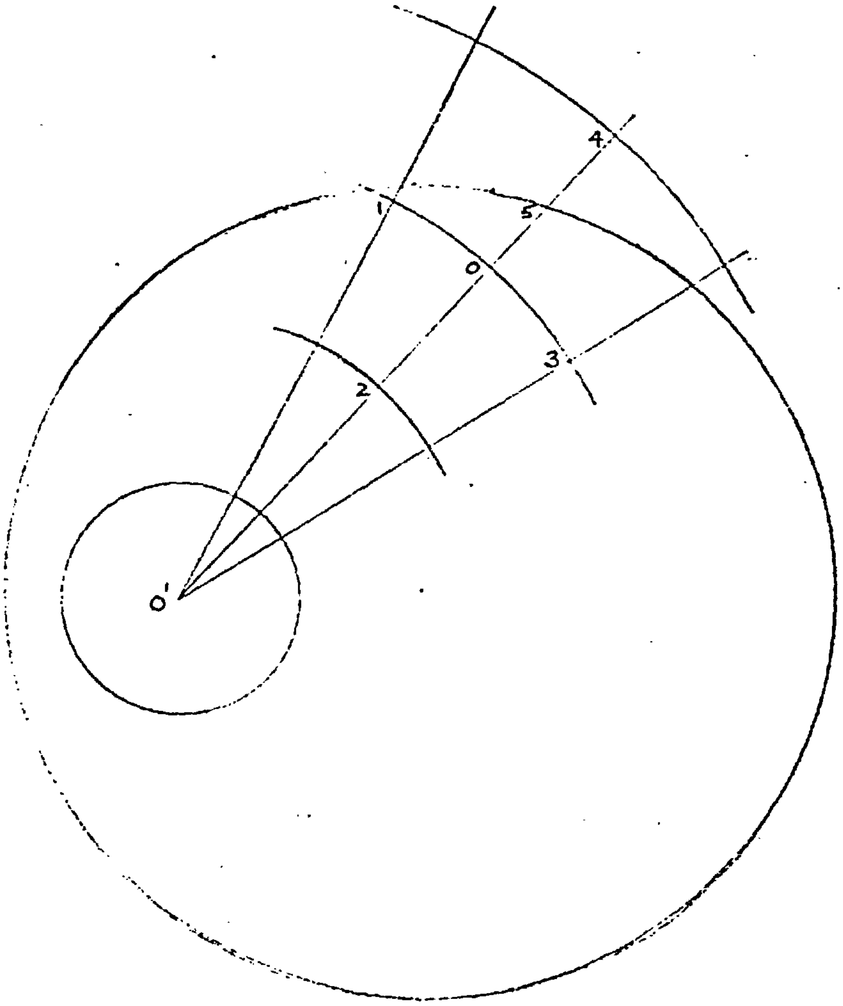

figure 3

Label the nodes as illustrated.

Let $05 = p_5 . \Delta r$; $03 = p_3 . \Delta \theta$;

$02 = \Delta r$; $01 = p_1 . \Delta \theta .$

Easily $$\frac{p_2\, p_5 (\, p_2 + p_5\,)}{2}\, \Delta r^2\, \frac{\partial^2 \phi_o}{\partial r^2} = p_2\, \phi_5 + p_5\, \phi_2 - (p_2 + p_5)\phi_o + + 0(\Delta r^3) \qquad (12)$$

and $$\frac{p_1\, p_3\, (\, p_1 + p_3\,)\Delta \theta^2}{2}\, \frac{\partial^2 \phi_o}{\partial \theta^2} = p_1\, \phi_3 + p_3\, \phi_1 - (p_1 + p_3)\phi_o + + 0(\Delta \theta^3)\ . \qquad (13)$$

Substituting from (12) and (13) into (4) and remembering that $\phi_5 = 0$ leads to

$$\frac{2}{p_1\, (p_1 + p_3)\Delta\theta^2 . \, r^2}\, \phi_1 + \frac{2}{(1 + p_5)\, \Delta r^2}\, \phi_2 + \frac{2}{p_3\, (p_1 + p_3)\Delta\theta^2 . r^2}\, \phi_3 +$$

$$+ \left[\frac{-2}{p_1\, p_3 \Delta \theta^2 . \, r^2} - \frac{2}{p_5 \Delta r^2} + \frac{1}{4r^2} + \lambda \right] \phi_o$$

$$+ 0(\Delta r) + 0(\Delta \theta) = 0.$$

In our more familiar notation this is

$$\frac{2}{p_1\, p_3\, (p_1 + p_3)\Delta\theta^2 . \, r^2}\, \phi_{i+1,j} + \frac{2}{(1 + p_5)\Delta r^2}\, \phi_{i,j-1} +$$

$$+ \frac{2}{p_3\, (1 + p_3)\Delta\theta^2 . \, r^2}\, \phi_{i-1,j} +$$

$$+ \left[\frac{-2}{p_1\, p_3\, \Delta \theta^2\, r^2} - \frac{2}{p_5\, \Delta r^2} + \frac{1}{4r^2} + \lambda \right] \phi_{i,j} + 0(\Delta r) + 0(\Delta \theta) = 0 \qquad (14)$$

at points neighbouring the boundary S_1.

We tacitly assumed that $p_3 = 1$.

The order of the matrix involved was approximately halved by using the symmetry of the problem at the time of discretization. The program used to set up the matrix is reproduced in the appendix. The matrix is sparse, banded and non-symmetric. The method of Lanczos, as described in chapter 1, was used to transform this matrix into tridiagonal form for various values of the radius of the control rod and of the distance between the centres. This method was chosen because of its suitability for finding extreme roots - here we sought the smallest ; because of its rapid convergence ; because of storage limitations we had to use a method which did not require the storage of the full coefficient matrix (or its equivalent) at any time.

2.2 The evaluation of det$(T - \lambda I)$, with T a tridiagonal matrix was performed by the usual algorithm, [34] p 423 , where, if we write $t_{ii}=\alpha_i$, $t_{i,i+1}=\beta_{i+1}$, $t_{i+1,i}=\gamma_{i+1}$ and denote the leading principal minor of $(T - \lambda I)$ of order r by $p_r(\lambda)$, then

$$p_r(\lambda) = (\alpha_r - \lambda)\, p_{r-1}(\lambda) - \beta_r \gamma_r\, p_{r-2}(\lambda) \quad (r=2,..,n)$$

where $p_o(\lambda) = 1$, $p_1(\lambda) = \alpha_1 - \lambda$.

$$\text{Also } p'_r(\lambda) = (\alpha_r - \lambda)\, p'_{r-1}(\lambda) - \beta_r \gamma_r\, p'_{r-2}(\lambda) - p_{r-1}(\lambda) \qquad (15)$$

$$\text{and } p''_r(\lambda) = (\alpha_r - \lambda)\, p''_{r-1}(\lambda) - \beta_r \gamma_r\, p''_{r-2}(\lambda) - 2\, p'_{r-1}(\lambda) ,$$

where $p'_o(\lambda) = p''_o(\lambda) = p''_1(\lambda) = 0$

$p'_1(\lambda) = -1.$

The rounding errors inherent in this algorithm are satisfactorily small.

2.3 The method of Laguerre (Laguerre [16] pp 87-103, Bodewig [3] , van der Corput [30], Parlett [26] , Wilkinson [54] pp443-445) was used to find the zeros of the characteristic polynomial defined by (15). This method was chosen since:-

a) if the characteristic polynomial, $p_n(\lambda)$, has real roots and the real line is divided into as many intervals as there are distinct roots, then from any initial point in such an interval the successive Laguerre iterates converge monotonically to the root therein;

b) locally, if the root is simple, convergence is cubic, otherwise it is linear.

The first above does not extend to the complex plane, while the second does ([26]) - this is not of great importance here as only real roots will be sought.

Parlett describes the algorithm as follows:-

Let the polynomial $p_n(\lambda)$ have roots $\lambda_1, \ldots \lambda_n$. Given an approximation λ to one of the roots, say λ_n.

Define

$$S_1(\lambda) = \frac{p_n'(\lambda)}{p_n(\lambda)} = \sum_{i=1}^{n} \frac{1}{\lambda - \lambda_i}$$

$$S_2(\lambda) = \frac{p_n'(\lambda)^2 - p_n(\lambda)\, p_n''(\lambda)}{p_n(\lambda)^2} = \sum_{i=1}^{n} \frac{1}{(\lambda - \lambda_i)^2} .$$

The next approximation, λ', to λ_n is obtained from

$$\lambda' = \lambda - \frac{n}{S_1 \pm \sqrt{(n-1)(n\, S_2 - S_1^2)}}$$

$$= \lambda - \frac{n}{S_1 \pm W^2} \quad \text{say.} \qquad (16)$$

Choose that square root of W which maximizes $|S_1 \pm W|$. Also, $|S_1 \pm W|^2 = |S_1|^2 + |W|^2 \pm 2\,\mathrm{Re}(\bar{S}_1 . W)$, so choose W to make $\mathrm{Re}(\bar{S}_1 . W)$ non-negative; when it is zero, arbitrarily take $0 \leq \arg(W) \leq \pi$.

Accepted roots may be suppressed by eliminating their influence on S_1 and S_2, this is done by noting that

$$S_1^{(p)} = \sum_{i=1}^{p} \frac{1}{\lambda - \lambda_i} = S_1^{(n)} - \sum_{i=p+1}^{n} \frac{1}{\lambda - \lambda_i} \qquad (17)$$

$$S_2^{(p)} = \sum_{i=1}^{p} \frac{1}{(\lambda - \lambda_i)^2} = S_2^{(n)} - \sum_{i=p+1}^{n} \frac{1}{(\lambda - \lambda_i)^2}$$

Parlett suggests the following numerical criteria for deciding whether a computed number is an acceptable approximation to a zero of the characteristic polynomial:-

Let λ be the current iterate, $\Delta\lambda$ the computed increment and $|\lambda| = |\mathrm{Re}(\lambda)| + |\mathrm{Im}(\lambda)|$. In the following α represents the base of the number system used by the machine and t the word length. In the work of Parlett $\alpha = 10$ and $t=8$.

Test 1: $|p_n(\lambda)| < \alpha^{-t} |\lambda| \, |p_n'(\lambda)|$. This test was designed to catch zero or small values of $p_n(\lambda)$ relative to $p_n'(\lambda)$, in practice it may be preferable to use $\alpha^{-t/2}$ or $\alpha^{-t/2-1}$ rather than α^{-t}. This test should also catch values of S_1 so large that there will be no change observable in λ to t decimal places.

Test 2: Let c be the modulus of the largest root found.

$$|\Delta\lambda| < \alpha^{-t/2} \max(|\lambda| , \alpha^{-t/2+1} . c).$$

For linear convergence this test is not fine enough, and

$$|\Delta\lambda| < \alpha^{-t/2-2} \max(|\lambda| , \alpha^{-t/4} . c)$$ is more appropriate.

Test 3: $|\Delta\lambda| < \alpha^{-t} . c$

Two further cases may occur as a result of complex eigenvalues causing cycles - we are not overly interested in these as we seek real roots, however see Parlett [26] for details.

Peters and Wilkinson [27] have also addressed themselves to the problem of deciding when a computed number is an adequate approximation to one of the roots of a polynomial.

2.4 In the implementation of the above algorithm it was found necessary to scale the matrix and/or the characteristic polynomial in order to ensure that the computed values of the characteristic polynomial and its first and second derivitives were always within the allowable range. As a hexadecimal machine was used, the coefficient matrix was scaled by a suitable power of 16.

2.5 Numerical results.

The twelve eigenvalues of smallest modulus were computed for the following cases:-

Outer radius equal to 1.000 in all cases.

$\Delta r = 0.1$, $\Delta\theta = \pi/64$.

	Inner radius	
	0.400	0.300
distances between centres	0.200	0.200
	0.175	
	0.150	0.150
	0.125	
	0.100	0.100
	0.075	
	0.050	0.050
	0.025	
	0.000	0.000

table 1

The order of the coefficient matrices used in the above varied from 390 in the case of the 0.400 hole, 0.000 between centres case to 487 for the 0.300 hole, 0.050 between centres situation.

The eigenvalues over domains with holes of smaller radius were not computed as it was felt that very little which was new, except the actual values of the eigenvalues, would be obtained. These cases would also have required the use of smaller values for Δr and $\Delta\theta$ and correspondingly larger matrices - together with very much more computer time.

It will be remembered that in the modified form of the Lanczos algorithm, as discussed in section 1.2, the upper and lower diagonal elements of the resulting tridiagonal matrix were denoted by γ and β respectively. All of the γ's are non-negative, while some (or all) of the β's are negative (cf algorithm A1-A15 of 1.1). It was found, in computing the eigenvalues of Laplace's equation on the region defined above that a reliable indicator of the accuracy of the results of the algorithm is given by what we have here termed the Symmetry Ratio (S.R.) at a particular point of the Lanczos algorithm, viz. the ratio of the number of positive β's in the tridiagonal matrix calculated thus far to the order of this tridiagonal matrix. In the cases where the S.R. was greater than 0.60 after 2n/3 Lanczos steps, it was found that the smallest roots could be computed accurately from this (2n/3) x (2n/3) tridiagonal matrix - see in particular the cases with inner radius of 0,400 and distances between centres of 0.200, 0.175, 0.150, 0.125. When the S.R. fell below this threshold value the roots were less accurately determined - see the cases with inner radius of 0.400 and distances between centres of 0.075 and 0.050. The choice of initial vectors affects the S.R. ratio profoundly - see as examples the cases where the inner radius is 0.400 and the distances between the centres are again 0.075 and 0.050.

The resulting eigenvalues are tabulated on the following pages. Besides the eigenvalues, the order of the matrix, the bandwidth, the symmetry ratio and the initial vectors usedare also tabulated for each case. Other information, where available and relevant, has also been tabulated. Where any doubt whatsoever exists about the accuracy of any of the lower order figures in the approximation to a root of the relevant tridiagonal matrix these have been underlined - note that this does not imply that the remaining figures represent a perfectly accurate representation of that eigenvalue. Very close or repeated roots, where indistinguishable, have been bracketed. The title "n-iterations" means n Lanczos iterations.

1) Outer radius 1.000; inner radius .400; distance between centres .200.

2) $\Delta r = .1$; $\Delta \theta = \pi/64 = .04908738$

3) Matrix has order 415 and bandwidth 17.

4) Symmetry ratio is .77 for 415, 311 and 276 Lanczos iterations.

5) $\max\limits_{i} \left(|v_1^T v_i^*|, |v_1^{*T} v_i| \right) = 13.16.$

6) 415 Lanczos iterations required from 4 to 12 Laguerre iterations, while 311 and 276 required from 3 to 10 and 3 to 11 respectively.

7) Initial vectors of $[1, 0, 1, 0, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$ were used.

415 its	311 its	276 its
$-.0019211\underline{57926}$		
$-.001922\underline{390160}$	$-.00192206\underline{6619}$	$-.0019220\underline{64204}$
	$-.0105637\underline{1195}$	
		$.35512\underline{34992}$
$6.2468\underline{5259}$ }	$5.95402\underline{5529}$	
$6.24\ 68\underline{7499}$ }	$6.2468653\underline{94}$	$6.24686539\underline{4}$
$12.5314\underline{5505}$		
$17.655\underline{67959}$		
$17.756\underline{04923}$	$17.75604\underline{923}$	$17.75604\underline{923}$
$24.9780\underline{5455}$	$24.9780545\underline{3}$	$24.97\underline{805474}$
25.67559720		
$30.71\underline{181545}$	$30.71\underline{181543}$	$30.711\underline{81593}$
$44.125\underline{32415}$	$44.125\underline{32286}$	$44.12534\underline{589}$
$55.684066\underline{10}$	$55.6518475\underline{5}$	
	$56.1912669\underline{5}$	
	$63.717\underline{37142}$	
		$65.924\underline{19245}$
	$66.1924\underline{2042}$	
		77.08921537

table 2

In addition with 276 iterations 4 complex roots, namely $55.87776990 \pm .3025067\underline{319}\, i$ and $88.62199\underline{263} \pm 4.32180\underline{5241}\, i$ were found.

1) Outer radius 1.000; inner radius .400; distance between centres .175.

2) $\Delta r = .1; \Delta\theta = \pi/64 = .04908738$

3) Matrix has order 418 and bandwidth 17.

4) Symmetry ratio is .54.

5) $\underset{i}{\mathrm{Max}}(|v_1^T v_i^*|, |v_1^{*T} v_i|) = 28.14$.

6) 418, 313 and 278 Lanczos iterations required between 3 and 13, 3 and 12 and 4 and 8 Laguerre iterations respectively.

7) Initial vectors: $[1, 0, 1, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$.

418 its	313 its	278 its
-.001921493$\underline{618}$	-.0019214$\underline{91051}$	-.001908690$\underline{459}$
-.0019224$\underline{38361}$	-.0019224$\underline{37427}$	-.0019222$\underline{14831}$
-.0046563759$\underline{96}$		
6.2468$\underline{44780}$)	6.24686$\underline{0068}$	6.246860068
6.2468$\underline{65798}$ }	6.24752$\underline{1416}$	7.09598$\underline{0427}$
18.66$\underline{039816}$)		
18.660$\underline{44173}$ }	18.66043376	18.660$\underline{43876}$
24.977$\underline{89644}$)	24.89437$\underline{624}$	
24.9780$\underline{3904}$ }?	24.97800751	24.97800747
31.024$\underline{82002}$	31.0252$\underline{4226}$	31.02524226
	36.46506$\underline{385}$	
	43.2784$\underline{5174}$	43.278$\underline{45173}$
	52.9945$\underline{6153}$	52.99$\underline{454615}$
	56.1471908$\underline{1}$	56.1472309$\underline{2}$
		64.46906775
		68.25$\underline{812674}$

table 3

1) Outer radius 1.000; inner radius .400; difference between centres .150.

2) Δr and $\Delta\theta$ as before.

3) Matrix is of order 419 and has bandwidth 17.

4) Symmetry ratio is .71.

5) Max($|v_1^T v_i^*|,|v_1^{*T} v_i|$)=22.04 .

6) 419, 314 and 279 Lanczos iterations required between 4 and 11, 3 and 12 and 3 and 9 Laguerre iterations respectively.

7) Initial vectors: $[1, 0, 1, 0,\ldots]^T$ & $[1, 1, 0, 1,\ldots]^T$.

419 its	314 its	279 its
-.001921852169	-.001922325657	-.001922321634
-.001922499148	-.001907408710	
		-.001073167734
6.246857736)	6.246868779	6.246859126
6.246884075 }	6.247105191	6.253645840
19.62478269)		
19.62597703 }?	19.62597702	19.62597702
24.96657408		
24.97811440	24.97811417	24.97811417
31.26051027		
31.29320758	31.29320712	31.29320715
	36.15119855	
	42.02771776	42.02771760
	50.52694585	50.52696381
	56.14217250	56.14224669
	63.06217236	63.06201531
		71.98414771

table 4

1) Outer radius 1.000; inner radius .400; distance between centres .125.

2) Δr and $\Delta \theta$ as before.

3) Matrix has order 420 and bandwidth 17.

4) Symmetry ratio is .67.

5) $\text{Max}(\ |v_1^T v_i^*|, |v_1^{*T} v_i|\ ) = 23.34$

6) 420, 315 and 280 Lanczos iterations require between 5 and 12, 5 and 11 and 5 and 1 Laguerre iterations respectively.

7) Initial vectors as in the previous two cases.

420 its	315 its	280 its
-.001922837949	-.002402849159	-.002402907173
-.001924196992	-.002403508532	-.002403510258
6.246792091)	6.186788760	
6.246850138 }	6.246382769	6.246382769
20.06221815		
20.66019056	20.66032398	20.66032398
24.97806017	24.97757281	24.97757281
26.07235397		
31.45575050	31.45595703	31.45595703
40.24055707		
40.32459958	40.32497754	40.32497754
48.61705890	48.61712850	48.61712853
	56.14607806	56.14607858
	61.97589462	61.97589249
	75.80685655	75.80712481
		79.11555079

table 5

1) Outer radius 1.000; inner radius .400; distance between centres .100.

2) Δr and $\Delta\theta$ as before

3) Matrix has order 420 and bandwidth 13.

4) Symmetry ratio was .63 - programmed to discontinue Lanczos iterations if S.R. $\geqslant$0.60 after 280 iterations.

5) Between 3 and 12 Laguerre iterations were required.

6) Initial vectors as before.

280 its		
-.001922682$\underline{049}$		
-.0019231028$\underline{02}$		
6.246$\underline{849100}$	)	
6.246$\underline{870023}$	}	
21.79$\underline{988273}$	)	
21.80689$\underline{794}$	}	?
24.978062$\underline{31}$		
31.4476639$\underline{2}$		
31.47020909		
38.403101$\underline{96}$		
47.3019$\underline{1831}$		
56.1465897$\underline{1}$		

table 6

1) Outer radius 1.000; inner radius .400; distance between centres .075.

2) Δr and $\Delta \theta$ as before.

3) Matrix has order 422.

4) Initial vectors $[1, 0, 1, 0, ..]^T$ and $[1, 1, 0, 1, ..]^T$. After 422 Lanczos iterations S.R.= 0.40. Roots isolated in between 3 and 12 Laguerre iterations. Case A.

5) Initial vectors as in 4) above - after 281 Lanczos iterations S.R.=0.43 . Roots found in from 3 to 11 Laguerre iterations. Case B.

6) Initial vectors: Both $[1, 1, 1, ... \quad]^T/\sqrt{422}$. S.R. = 0.68 after 281 steps. 3 to 7 Laguerre iterations. Case C.

A	B	C
		-.8896394410
		-.1441007695
-.006915915808		
	-.003544375245	
		-.003364563359
-.002375539892		
-.001919030365		
3.6562 ± i2.6500		
6,246728087	6.245452303	6.238470911
6.246858643	6.27182903	
	22.42241228	
23.05309540	23.05311027	23.05302290
23.06809932		
24.97936477	24.97689533	24.97651266
25.31726614		
31.16025235	31.16021609	31.16022409
	36.37293601	36.37294637
	46.61012717	46.61013153
	56.14509585	
	60.72423441	60.72423806
		78.25357051
		85.58270201

table 7

1) Outer radius 1.000; inner radius 0.400; distance between centres 0.050.

2) Δr and $\Delta \theta$ as before.

3) Matrix has order 422 and bandwidth 15.

4) Initial vectors : $[1, 0, 1, 0, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$. S.R. = 0.33 after 422 Lanczos iterations. From 3 to 8 Laguerre iterations required. Case A.

5) Initial vectors: both $[1, 1, 1, 1, \ldots]^T/\sqrt{422}$. S.R. = 0.65 after 251 Lanczos steps. 3 to 8 Laguerre iterations - no complex arithmetic was required at all.

A	B
$-221.1259\underline{387}$	
$-134.25203\underline{82}$	
$-48.8412\underline{4239}$	
$-0.82200045\underline{103}$	
	$-0.00192260752\underline{4}$
	-0.001895980614
	6.246861847
$18.57\underline{846865}$	
	$24.33\underline{155190}$
	24.97806324
$27.12\underline{803354}$	
	$30.42029\underline{466}$
	$35.789519\underline{41}$
	$44.8063\underline{0944}$
	46.11340696
	$60.3540\underline{2158}$
	$77.963\underline{95540}$
	$90.8840730\underline{7}$
$98.133\underline{98455}$	
$210.0125 \pm i60.6692$	
320.2267819	
$346.3\underline{715680}$	
$409.091850\underline{0}$	

table 8

Comparing the above roots with each other and with those obtained elsewhere it is clear that the results A above are extremely suspect.

1) Outer radius 1.000; inner radius .400; distance between centres .025.

2) Δr and $\Delta\theta$ as before.

3) Matrix has order 422 and bandwidth 15.

4) Initial vectors: $[1, 0, 1, 0, \ldots]^T$ and $[1, 1, 0, 1, 0, \ldots]^T$. After 422 Lanczos steps S.R.=0.43. 5 to 13 Laguerre iterations. Case A.

5) Initial vectors: $[1, 1, 1, \ldots]^T/\sqrt{422}$ (both). After 281 Lanczos S.R.=0.65.3 to Laguerre iterations. No complex arithmetic.

A	B
-0.001923779861	
-0.001922781859	-0.001922607592
-0.001893491016	-0.001922205595
6.246844077)	6.247503885
6.246863468 }	
24.97803215	24.97805930
25.56383611	25.56383610
29.30047859	29.30047839
29.46077186	
35.20639742)	35.20637832
35.20887557 }	
45.74162651	45.76104139
	52.61059535
	60.04904599
	77.66304742
	95.85556466

table 9

1) Outer radius 1.000; inner radius .400; distance between centres 0.000 .
2) Δr and $\Delta\theta$ as before.
3) Matrix has order 390 and bandwidth 13.
4) Both initial vectors were $[1, 1, 1, 1, \ldots]^T/\sqrt{390}$. After 260 Lanczos steps S.R.= 0.70. 3 to 11 Laguerre iterations. The tridiagonal matrix required scaling. No complex arithmetic needed at all.
5) Severe scaling problems were encountered when initial vectors $[1, 0, 1, 0, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$ were used - so severe in fact that this attempt was abandoned.

$-0.001921729\underline{375}$
$-0.00192260755\underline{0}$
$6.251\underline{218201}$
16.79002914
$24.97806\underline{218}$
$26.21547\underline{746}$
26.24364166
34.94698020
59.77875325
$97.7740004\underline{5}$
$99.4010\underline{9867}$
$99.6773\underline{5847}$

table 10

Summary of results for 0.400 hole :-

0.200	0.175	0.150	0.125	0.100
6.24686539	6.2468	6.2468	6.246382769	6.246
		6.253645840		
17.75604	18.660	19.6259	20.6603259	21.80689
24.9780545	24.978	24.978	24.97757291	24.978062
30.711	31.02524	31.29	31.4559370	31.4476639
				31.47020909
	36.46306			
				38.403101
			40.5249	
44.12534	43.2784	42.02771		
			48.617	47.3019
	52.9945	50.52696381		
	56.1471988	56.142	56.14607853	56.1465897
			61.975	
65.924		65.062		
		71.98414771	75.80712481	
77.08921337			79.11355079	

table 11

	0.075	0.050	0.025	0.000
	6.238470911	6.246851847	6.247505885	
				6.251
				16.7900291
	23.0530	24.33	24.978	24.97806
	24.976	24.97806324	25.563836	26.21547
				26.24364166
	31.16022409	30.42029	29.30047839	
	36.8729	35.789519	35.20637832	34.94698020
		44.8063	45.7610	
	46.61	46.11340696		
			52.610	
	60.7242380			
		60.3540	60.0490	59.77875325
	78.253570	77.964	77.66304	
	85.58270201			
		90.8840731		
			95.85556	97.7740004
				99.4010
				99.6773

table 11

1) Only the positive roots are shown as these are the only ones which have meaning for the differential operator on the domain under consideration.

2) Only figures about which there is some measure of certainty are shown in the table.

1) Outer radius 1.000; inner radius 0.300; distance between centres 0.200.

2) Δr and Δθ as before.

3) Matrix has order 480 and bandwidth 19.

4) Initial vectors: $[1, 0, 1, 0, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$. After 320 Lanczos S.R. = 0.64. 3 - 7 Laguerre iterations. No complex arithmetic required.

-2.109110958
2.082030299
13.19071710
14.23008453
25.77265174
35.62908874
45.03086011
46.49074186
52.17745328
59.40013135
77.49626375
89.41909211

table 12

- - - - - - -

1) Outer radius 1.000; inner radius .300; distance between centres 0.150.

2) Δr and Δθ as before.

3) Order 484, bandwidth 19.

4) Initial vectors: i) both $[1, 1, 1, \ldots]^T / 484$ and ii) $[1, 0, 1, 0, 1, \ldots]^T$ and $[1, 1, 0, 1, \ldots]^T$. 30 minutes of CPU time ran out before a tridiagonal matrix with S.R. ≤ .60 or the full tridiagonal matrix could be found in both cases.

1) Outer radius 1.000; inner radius .300; distance between centres .100.

2) Δr and $\Delta\theta$ as before.

3)Order of matrix 485, bandwidth 17.

4) Initial vectors:$[1, 0, 1, 0, \ldots.]^T$ and$[1, 1, 0, 1, 0, \ldots]^T$.

S.R. = 0.63 after 323 Lanczos iterations. 3 - 7 Laguerre iterations required. No complex arithmetic required.

-1.454652066
2.082052111
13.19104055
16.81013117
24.40718988
30.78788149
42.33311589
46.70162282
57.06984128
62.68545895
81.45435486
95.62170301

table 13

1) Outer radius 1.000; inner radius .300; distance between centres .050.
2) Δr and $\Delta\theta$ as before.
3) Matrix has order 487 and bandwidth 17.
4) Initial vectors : as in previous case. After 324 Lanczos iterations S.R. = 0.60. 4 – 6 Laguerre iterations. Only the complex root required complex arithmetic.

$2.082029\underline{195}$
$2.631192\underline{969}$
$13.190\underline{17144}$
$13.903\underline{42691}$
$18.26\underline{658392}$
$22.91\underline{197276}$
29.90050740
$39.4304997\underline{1}$
41.91186902
$57.695920\underline{07}$
$58.59063\underline{285} \pm i11.12541\underline{546}$

table 14

– – – – – – –

1) Outer radius 1.000; inner radius 0.300; distance between centres 0.000.
2) Δr and $\Delta\theta$ as before.
3) Matrix has order 455 and bandwidth 13.
4) Initial vectors as in previous case. After 455 Lanczos the S.R. was found to be 0.14 . No meaningful eigenvalues could therefore be found.

5) At this stage the program, which had been stored on disk, was unfortunately inadvertently destroyed so that this case could not be run with initial vectors $[1, 1, 1, 1, \ldots]^T/\sqrt{455}$.

Summary of results for the 0.300 hole:-

0.200	0.100	0.050
2.082030299	2.0820	
		2.631193
13.1907171	13.19104	13.190
14.23		13.903
	16.8101312	
		18.27
25.772	24.41	22.91
35.629088	30.788	29.90050740
		39.4304997
45.03	42.33	41.91186902
46.491	46.701623	
52.17745		
59.4001314	57.070	57.695928
77.49	62.6854590	
89.41909211	81.45	
	95.62170301	

table 15

The comments after table 11 are relevant here too.

In order to monitor the "accuracy" of the tridiagonalization technique each $v_i^{*T}v_1$ and h_{i1}, i=2,..,k were computed. (Initially the $v_i^T v_1^*$'s were also calculated, but as these were found to follow the $v_i^{*T}v_1$'s in behaviour, their computation was not persevered with).

2.6 Some comments on the above results.

1) In none of the test cases was the Lanczos iteration terminated due to a γ falling below the preset threshold. This is not entirely surprising as one would not expect the derived tridiagonal matrix to be derogatory.

2) These results again clearly illustrate the adverse effect of a low symmetry ratio. A low ratio results in excessive cancellation error, which can frequently be avoided by restarting the cycle with a different set of initial vectors.

2.7 Practical experiences in isolating the smaller roots of the high order polynomials associated with this eigenvalue problem.

Scaling of both the original coefficient matrix and the resulting polynomial is essential in order to keep all computed numbers within the allowable range. This scaling was performed at two points; first, the coefficient matrix is scaled by a power of 16 to ensure that all the elements down the main diagonal, the super-diagonal and the sub-diagonal are less than or equal to one in absolute value. Further, if any of the intermediate values used in the computation of $p_n(\lambda)$, $p_n'(\lambda)$ or $p_n''(\lambda)$ lie outside the range $\pm 2x10^{-40}$ to $\pm 10^{55}$ a second scaling routine, which scales the elements of the tridiagonal matrix, is invoked - see the program for details.

As only the smallest roots were sought, zero was always used as the starting point for the Laguerre iteration.

Even though only real roots were sought it was necessary to regard all quantities in the Laguerre iteration procedure as complex - frequently the intermediate iterates of a real root (which might even have a small complex part itself) were complex:

<u>example</u>: inner radius .4; distance between centres .2 ; 415 Lanczos iterations; 4th root. The iterates are:-

x	+ i y
0.	0.
0.4133845×10^{-3}	$-0.2252965 \times 10^{-3}$
0.1162117×10^{-2}	$-0.1026660 \times 10^{-3}$
0.1513481×10^{-2}	$-0.1027277 \times 10^{-4}$
0.1525119×10^{-2}	$-0.1784107 \times 10^{-8}$
0.1525116×10^{-2}	$-0.1915534 \times 10^{-9}$

<u>table 16</u>

At this stage the iteration ceased because $|\Delta z/z| < 10^{-5}$. Complex arithmetic is not always required however – especially when the number of Lanczos steps performed is smaller than the order of the natrix :

<u>example</u>: inner radius .4; distance between roots .175; 279 Lanczos iterations; 3rd root. The iterates are:-

x	+ i y
0.	0.
0.4258399×10^{-3}	0.
0.1174607×10^{-2}	0.
0.1421907×10^{-2}	0.
0.1495462×10^{-2}	0.
0.1516998×10^{-2}	0.
0.1523243×10^{-2}	0.
0.1524897×10^{-2}	0.
0.1525113×10^{-2}	0.
0.1525115×10^{-2}	0.

<u>table 17</u>

At this stage the iterative procedure was stopped, again because $|\Delta z/z| < 10^{-5}$.

A large function value is not necessarily indicative of the current iterate being far from the required root – in fact neither does a small function value always indicate proximity to the root – as a simple scaling of the tridiagonal matrix is sufficient to drastically alter the value of the characteristic polynomial.

<u>Example</u>:inner radius .4; distance between centres .125; 420 Lanczos

iterations; 3rd root. A subscript n on a number a (i.e. a_n) means $a{\times}10^n$ in the following two tables.

root		function val.		deriv. val.	
x +	i y	fr +	i fi	fdr ÷	i fdi
$-.2323192438_{-3}$	$.2333059461_{-3}$	$.30_{23}$	0.	$.13_{30}$	0.
$.8670688199_{-3}$	$.1122952829_{-3}$	$-.27_{29}$	$-.24_{29}$	$.10_{33}$	$.31_{33}$
$.1320851941_{-2}$	$.3863212952_{-4}$	$.19_{28}$	$-.79_{27}$	$-.72_{31}$	$.11_{31}$
$.1464933518_{-2}$	$.1154854038_{-4}$	$.12_{27}$	$-.54_{26}$	$-.14_{31}$	$.36_{30}$
$.1507481796_{-2}$	$.3386186392_{-5}$	$.84_{25}$	$-.35_{25}$	$-.30_{30}$	$.67_{29}$
$.1519944485_{-2}$	$.9913165141_{-6}$	$.68_{24}$	$-.28_{24}$	$-.81_{29}$	$.16_{29}$
$.1523592801_{-2}$	$.2901942536_{-6}$	$.57_{23}$	$-.23_{23}$	$-.23_{29}$	$.45_{28}$
$.1524660789_{-2}$	$.8494632912_{-7}$	$.49_{22}$	$-.20_{22}$	$-.67_{28}$	$.13_{28}$
$.1524973358_{-2}$	$.2484600450_{-7}$	$.42_{21}$	$-.17_{21}$	$-.20_{28}$	$.34_{27}$
$.1525064589_{-2}$	$.7199723506_{-8}$	$.36_{20}$	$-.14_{20}$	$-.57_{27}$	$.11_{27}$
$.1525090393_{-2}$	$.1874170990_{-8}$	$.30_{19}$	$-.12_{19}$	$-.17_{27}$	$.32_{26}$

table 18

At this point the iteration was stopped because $|\Delta z/z| < 10^{-5}$. It is interesting to compare the above with the convergence to the same root after the application of 280 Lanczos steps.

root		function val.		deriv. val.	
x +	i y	fr +	i fi	fdr +	i fdi
$-.1643811_{-2}$	$.1690914_{-2}$	$-.94_{-14}$	0.	$-.32_{-7}$	0.
$.8200854_{-3}$	$.8474231_{-3}$	$.19_{-6}$	$-.58_{-5}$	$-.56_{-2}$	$.14_{-1}$
$.1584186_{-2}$	$.7761511_{-4}$	$-.46_{-8}$	$.50_{-8}$	$.16_{-4}$	$.34_{-5}$
$.1524907_{-2}$	$.3272814_{-7}$	$.16_{-9}$	$.18_{-9}$	$.24_{-5}$	$-.28_{-6}$
$.1524996_{-2}$	$.7624689_{-16}$	$-.23_{-12}$	$.83_{-13}$	$.26_{-5}$	$-.11_{-9}$
$.1524996_{-2}$	$.2009192_{-26}$	$.12_{-19}$	$.19_{-21}$	$.26_{-5}$	$-.26_{-18}$

table 19

The iteration ceased here because

$$|\text{function value}| \leq 10^{-5} * |\text{root}| * |\text{derivative of function}| .$$

The apparent difference in the two above approximations to the same root is caused by the different scaling factors used in the two tridiagonal matrices. The final root of table 19 needs to be multiplied by only 1.0000619 in order to make it equal to that of table 18 (to 7S) and yet the function and derivative values differ vastly. Note too that in table 18 double precision was used and in table 19 single precision - hence the differing number of significant figures.

Determining the multiplicities of the eigenvalues without resorting to computing the corresponding eigenvectors is not an easy task. No conclusion in this regard may be made by considering the absolute values of the function and 1st derivative in isolation, as these quantities are sensitive to scaling. Reasonable conclusions may however be drawn by comparing the absolute values of these functions - if these were of the same small order of magnitude we concluded that multiple (or close) eigenvalues were present, otherwise not.

The root capturing criteria has to be chosen very carefully so as not to miss roots - we sometimes did (see, for example, the table on page 41). In this regard see also the paper by Peters and Wilkinson [27] .

FINAL COMMENTS ON PART ONE.

We have here described a modified form of the well-known Lanczos minimized iteration technique for reducing an arbitrary matrix with real roots to tridiagonal form. Various modified algorithms are given - all theoretically equivalent, but of course computationally different. We have indicated the most superior of these algorithms, pointing out that even with this version failure may occur but can be recognized before commencing the isolation of the roots of the tridiagonal form by monitoring the value of the Symmetry Ratio. A low ratio (approximately less than 0.6) goes hand in hand with large cancellation error, while when the ratio is high some of the extreme roots may be isolated by utilizing fewer than n applications of the theorem. Some indications of the stability have been given.

This technique was used to solve numerically the differential eigenvalue problem $\nabla^2 u - \lambda u=0$ on the domain of figure 1 of chapter 1. We used this technique primarily because considerably less storage than is customarily used by tridiagonalization techniques is required, thus obviating the need for a vast amount of fast storage or of continual paging or of rolling in and out of slower core. Two hole sizes were used, each of these was placed at various distances from the origin and the smallest eigenvalues of each case were found.

PART 2

CHAPTER 1 : INTRODUCTION TO TAU METHODS.

1.1 Introduction : Lanczos, in 1938 [17], introduced his tau method for the solution of the linear differential equation with polynomial coefficients and right hand side, say

$$D\ y(x) = f(x) \ . \tag{1}$$

He further expanded this method in 1957 [20]. Rather than truncate an infinite power series solution to this differential equation the Lanczos procedure perturbs the differential equation and finds the exact polynomial solution to this perturbed equation.

1.2 The tau method : We will illustrate the method by means of the simple differential equation $y'(x) + y(x) = 0$, $y(0) = 1$, which defines $y = e^{-x}$. Insert the formal power series approximation $y^*(x) = a_0 + a_1 x + \dots + a_n x^n + \dots$ to $y(x)$ into the differential equation and obtain the system of linear algebraic equations

$$j\ a_j + a_{j-1} = 0 \qquad j=1,2,\dots \ , \tag{2}$$

which is then solved in terms of a_o. The initial condition may be satisfied by adjusting a_o. This formal expansion may be tested for convergence.

The solution to this differential equation is an infinite power series. No polynomial solution can be obtained unless the exact solution is a polynomial. A polynomial solution of order n may be obtained by truncating the series defined by (2), this is however equivalent to solving only the first n equations in (2) with a perturbation term of the form τx^n on the right hand side of the differential equation, so that in the (n+1)-th equation

$$(n + 1)\ a_{n+1} + a_n = \tau \ ,$$

$a_{n+1} = 0$ and the cancellation of the coefficients a_j, $j \geqslant n$ propagates downwards instead of upwards, and the solution is preserved.

This solution is a partial sum of the Taylor series for $y(x)$ around $x = 0$ and therefore its accuracy deteriorates as we depart from the point of expansion. Lanczos hereupon proposed a perturbation term which distributes the error more evenly over the interval, J, on which the solution is required. If this interval is $[-1, 1]$ then it is natural to replace the original zero right hand side of the equation by its best algebraic polynomial approximation of degree n, that is, by the Chebyshev polynomial $T_n(x)$.

1.3 The Canonical polynomials : In 1952 [19], and more extensively in 1957 [20] , Lanczos introduced a sequence $Q = \{Q_m\}$, $m \in N_o$ (where N_o is the set of non negative integers), of canonical polynomials associated with the differential operator D, which he defined by means of the functional relation

$$D\, Q_m(x) = x^m, \quad m \in N_o.$$

If the given differential equation is perturbed by $H_n(x) = \tau_r(x)\rho_{n-r}(x)$, where $\rho_{n-r}(x) = \sum_{m=0}^{n-r} c_m^{(n-r)} x^m$, $\tau_r(x) = \sum_{m=0}^{r} \tau_m x^m$, n and r positive integers, and if $f(x) = \sum_{m=0}^{k} f_m x^m$, k an integer, then because of the linearity of D the solution to the perturbed equation,

$$D\, y_n(x) = f(x) + H_n(x),$$

is simply $y_n(x) = \sum_{m=0}^{n-r} c_m^{(n-r)} \sum_{i=0}^{r} \tau_i\, Q_{m+i}(x) + \sum_{m=0}^{k} f_m\, Q_m(x)$.

In particular then : assume that $D\, y(x) = 0$ is a proposed problem with initial conditions $y^{(j)}(\alpha) = y_\alpha^{(j)}$, $j=0,..,\nu-1$; J being the interval on which the solution is being sought and α a point of J. For simplicity assume further that D is a first order operator. If the canonical polynomials are known for all non-negative $n \in N$, then the solution to the perturbed problem

$$D\, y_n^*(x) = \tau\, T_n(x) = \tau \sum_{k=0}^{n} c_k^{(n)} x^k$$

is simply $y_n^*(x) = \tau \sum_{k=0}^{n} c_k^{(n)} Q_k(x)$. The parameter τ is chosen so that the initial condition $y(\alpha) = y_\alpha$ is matched. Therefore

$$y_n^*(x) = \frac{y_\alpha \sum_{k=0}^{n} c_k^{(n)} Q_k(x)}{\sum_{k=0}^{n} c_k^{(n)} Q_k(\alpha)} \quad . \tag{3}$$

There are several advantages to expressing the approximate solution in terms of canonical polynomial. First, the whole of the computation need not be repeated if an approximation of higher degree is required. Second, they do not depend on the initial or boundary conditions, or on the interval over which the solution is sought. Further, canonical polynomials may be used to solve eigenvalue problems where the parameter may enter either linearly or non-linearly.

1.4 Construction of Canonical polynomials : Lanczos' technique for constructing the canonical polynomials $Q_i(x)$ associated with D is to solve a system of linear equations, like (2), for $0 \leq j \leq i$, with a 1 on the right hand side of the i-th equation. This procedure need not be trivial as the system may be over-determined and for some subset of the index i the canonical polynomials may be multiple or even be undefined. All these possibilities have to be taken into account if the tau method is to be automized. In the next paragraph we give a short description of a more satisfactory technique for their construction.

This recursive technique is due to Ortiz [24]. Again consider the equation

$$D\, y(x) = f(x) \quad . \tag{4}$$

As $Q_n(x)$ is by definition a polynomial and D is a linear operator which maps polynomials into polynomials it is reasonable to start by considering the effect of D on the monomial x^n. This is the polynomial

$$D\, x^n = \sum_{r=0}^{m} a_r^{(n)}\, x^r \tag{5}$$

of degree $m \geqslant n$. Then

$$\frac{1}{a_m^{(n)}}\, D\, x^n = x^m + \frac{1}{a_m^{(n)}} \sum_{r=0}^{m-1} a_r^{(n)}\, x^r \quad .$$

Assuming that all the $Q_r(x)$, $r < m$, are known at this point we may write

$$\frac{1}{a_m^{(n)}}\, D\left[x^n - \sum_{r=0}^{m-1} a_r^{(m)}\, Q_r(x)\right] = x^m \quad , \tag{6}$$

because of the linearity of D. Therefore

$$Q_m(x) = \frac{1}{a_m^{(n)}} \left[x^n - \sum_{r=0}^{m-1} a_r^{(n)}\, Q_r(x)\right] \quad . \tag{7}$$

For the particular case $y'(x) + y(x) = 0$ we find that as $m=n$, $a_m^{(n)}=1$, $a_{m-1}^{(n)}=n$ and $a_r^{(n)}=0$ for $0 \leq r \leq m-1$ and $m \geqslant 2$, therefore

$$Q_n(x) = x^n - n\, Q_{n-1}(x),$$

which gives recursively $Q_0(x)=1$, $Q_1(x)=x-1$, $Q_2(x)=x^2-2x+2$, $Q_3(x)=x^3-3x^2+6x-6$ etc.

This technique is not however entirely without its difficulties. First, m need not be equal to n, in general it will be greater (e.g. if $D\,y(x) = y'(x) + y(x)$), causing a "gap" between the exponent of x^n and the leading one of Dx^n. Second, $a_m^{(n)}$ could be zero (e.g. $D\,y(x) = x\,y'(x) - y(x)$). Both of these situations give rise to undefined canonical polynomials, $Q_v(x)$ $v \in S$ say. These undefined canonical polynomials affect the possibility of generating all the canonical polynomials by means of (7), as not all the $Q_r(x)$, $0 \leq r \leq m-1$ are necessarily defined. This in turn affects the possibility of obtaining a solution at all to the perturbed problem $D\,y_n^*(x) = \tau T_n(x)$ as there may be no canonical polynomials available to generate the powers x^v, $v \in S$, in the expression of $T_n(x)$.

Another problem is that of multiple canonical polynomials, which arise in examples such as $D\,y(x) = x^2\,y''(x) + 2(x-1)\,y'(x) - 2\,y(x)$ – here $Q_0(x)$ is either $-\frac{1}{2}$ or $\frac{1}{2}$. $-\frac{x}{2}$

In order to circumvent these difficulties Ortiz has introduced the following modified definition for $Q_n(x)$:

$$D\,Q_n(x) = x^n + R_n(x) \quad , \tag{8}$$

where $R_n(x)$ is a polynomial generated by x^v , $v \in S$. This "Residual Polynomial" $R_n(x)$ belongs to the subspace $\mathbb{R}$ generated by the powers of x which are "unattainable" by the operator D acting on polynomials. Then, although x^v, $v \in S$, cannot be generated with the $Q_n(x)$'s, their residual polynomials $R_n(x)$, which belong to $\mathbb{R}$, will take care of that segment of the perturbation polynomial.

Far more detail concerning undefined and multiple canonical polynomials may be found in [24].

1.5 Eigenvalue problems : Fox and Parker [18] have discussed the application of the original formulation of the tau method to the eigenvalue problems of linear differential equations. In the next paragraphs we point out how the recursive technique of Ortiz may be used for these problems.

Here the differential operator depends on a parameter λ , hence so do the canonical polynomials. Because of the fact that the algebraic kernel of D_λ depends on the spectrum and may be empty for some eigenvalues and not others this extension is not entirely trivial. The advantages of using this approach are that exact polynomial solutions satisfying the boundary conditions are immediately detected; the basis in which the eigensolutions are represented is generated recursively; the order of the λ-determinant is independent of the degree of the desired approximation; as the degree of the approximation increases the lower eigenvalues are obtained with rapidly increasing accuracy and the higher order eigenvalues give a wide range of the spectrum and also that if an approximation of higher degree or at different boundary points is required the previous computational effort is not entirely wasted.

Again we illustrate via an example. Consider

$$D_\lambda y(x) = x(3x^2 - 1)y''(x) - 2y'(x) - \lambda x\, y(x) = 0 \qquad (9)$$

with the boundary conditions $y(\pm 1)=0$.

Applying D_λ to x^n we get

$$D_\lambda x^n = [3n(n-1) - \lambda]\, x^{n+1} - n(n+1)x^{n-1}$$

and immediately

$$Q_{n+1}(x,\lambda) = \frac{1}{3n(n-1)-\lambda}\left[x^n + n(n+1)\; Q_{n-1}(x,\lambda)\right] \text{for } n \; 1.$$

The set of indices of undefined canonical polynomials is $S = \{0\}$. In order to satisfy the three conditions, viz. two boundary conditions and one undefined canonical polynomial, a three term perturbation, of the form

$$H_n(x) = \tau_0\, T_n(x) + \tau_1\, T_{n-1}(x) + \tau_2\, T_{n-2}(x),$$

is used. Therefore

$$y_n^*(x,\lambda) = \sum_{i=0}^{2} \tau_i \sum_{k=0}^{n-1} c_k^{(n-i)}\; Q_k(x,\lambda) = \tau_0 A(x,\lambda) + \tau_1 B(x,\lambda) + \tau_2 C(x,\lambda).$$

The approximate solution has then to satisfy the three conditions:-

$$\tau_0 A(-1,\lambda) + \tau_1 B(-1,\lambda) + \tau_2 C(-1,\lambda) = 0$$

$$\tau_0 A(+1,\lambda) + \tau_1 B(+1,\lambda) + \tau_2 C(+1,\lambda) = 0$$

$$\tau_0\, \alpha \qquad + \tau_1\, \beta \qquad + \tau_2\, \gamma \qquad = 0,$$

where α, β, γ are the sum of residuals in the first, second and third terms respectively. In order to get a non-trivial solution the λ-determinant must vanish:

$$\begin{vmatrix} A(-1,\lambda) & B(-1,\lambda) & C(-1,\lambda) \\ A(+1,\lambda) & B(+1,\lambda) & C(+1,\lambda) \\ \alpha & \beta & \gamma \end{vmatrix} = 0\,.$$

The roots of this equation give the eigenvalues of (9).

CHAPTER 2 : SOME LANCZOS TAU - METHOD OF LINES TECHNIQUES.

2.1 Introduction : A few methods have recently been proposed for approximating the solution to parabolic partial differential equations using Chebyshev polynomials. Elliott [7] and Wragg [37] use semi-discretization techniques; Fox and Parker [10] , Knibb and Scraton [14] , Dew and Scraton [6] and Knibb [13] assume solutions of the form $u(x,t) = \sum_{r=1}^{N} a_r(t)\, T_r(x)$ or $u(x,t) = \sum_{r=1}^{N} a_r(t)\, x^r$, N finite or infinite, to the differential equation and Fox and Parker [10] also use a prior integration technique coupled with the assumption that the solution has the first form above.

Insofar as the solution to elliptic partial differential equations is concerned, Mason [21] has suggested a separation of the variables type solution, viz. $u(x,y) = \sum_{r,s=0}^{N} a_{rs}\, T_r(x)\, T_s(y)$. He also, rather tentatively perhaps, suggests a collocation method for solving these problems.

In this chapter we investigate some semi-discretization approaches to solving an elliptic partial differential equation and provide error analyses to these. Also, because of the similarity between Wragg's technique and those of this chapter we later, in another chapter, give an error analysis of his method. The error equations derived are differential-difference equations.

2.2 Techniques : Here we will attempt an approximate solution to Laplace's equation

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 , \tag{1a}$$

defined on the domain $x_a \leq x \leq x_b$, $y_a \leq y \leq y_b$, having the boundary conditions

$$\begin{aligned} u(x_a,y) &= f(y), \; u(x_b,y) = g(y), \\ u(x,y_a) &= h(x), \; u(x,y_b) = k(x). \end{aligned} \tag{1b}$$

Denote the interior of the rectangle by Ω, the boundary by S and let $\bar{\Omega} = \Omega + S$.

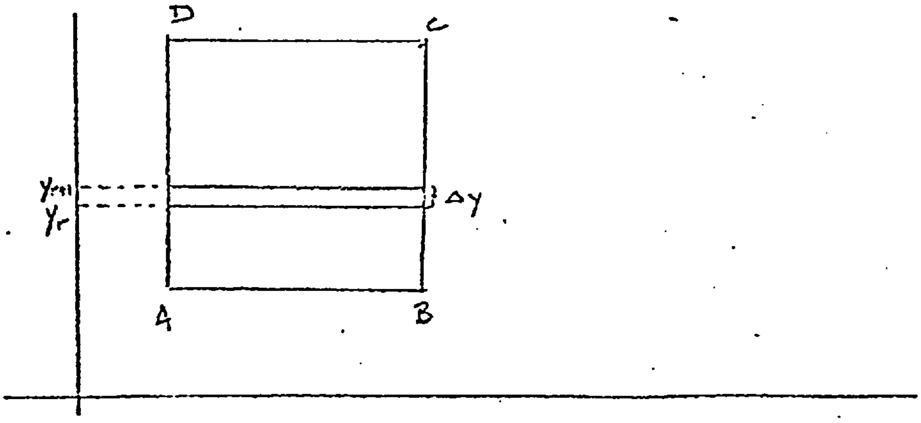

figure 1

2.3 Our first (unsuccessful) attempt at solving this problem using the tau method was via a straightforward semi-discretization approach. The domain Ω of figure 1 was divided into N strips of equal width, $\Delta y = (y_b - y_a)/N$, in the y-direction. Let $y_r = y_a + r.\Delta y$ and $u_r = u(x,y_r)$. We discretized (1) at the point (x,y_r) in the following four ways :

$$\text{I:}\quad \frac{d^2u_r}{dx^2} + \frac{u_r - 2\,u_{r-1} + u_{r-2}}{\Delta y^2} + 0(\Delta y) = 0$$

$(\Delta y)^2$

$$\text{II:}\quad \frac{d^2u_r}{dx^2} + \frac{2\,u_r - 5\,u_{r-1} + 4\,u_{r-2} - u_{r-3}}{\Delta y^2} + 0(\Delta y^2) = 0$$

$$\text{III:}\quad \frac{d^2u_r}{dx^2} + \frac{u_{r+1} - 2\,u_r + u_{r-1}}{\Delta y^2} + 0(\Delta y^2) = 0$$

$$\text{IV:}\quad \frac{d^2}{dx^2}\left(u_{r+1} + u_{r-1}\right) + \frac{u_{r+1} - 2\,u_r + u_{r-1}}{y^2} + 0(\Delta y^2) = 0.$$

In the sequel these discretizations will be denoted by I, II, III and IV respectively. I and III are the usual backward and central difference approximations, II (see Collatz [5] p539) is a more exact backward difference approximation, while IV is an improved central difference approximation. In each of the above cases a polynomial approximation $u_r = \sum_{m=0}^{n} a_m^{(r)} x^m$ (2) was assumed for $u(x,y_r)$. In order to obtain (2) as a solution to I – IV, perturb each of these by $(\tau_1^{(r+q)} + \tau_2^{(r+q)}\, x)\, T_n^*(x)$, q=0 for I, II; q=1 for III, IV. Hereafter we let $T_n^*(x) = \sum_{m=0}^{n} c_m^{(n)} x^m$ where $c_0^{(n)} = (-1)^n$;

n or (n+1) ?

T^*_{n-1}

reduced to [0,1]

$$c_m^{(n)} = 2^{2m-1}\left[2\binom{n+m}{n-m} - \binom{n+m-1}{n-m}\right](-1)^{n+m},\quad m=1,2,3,\ldots .$$

n or n+1 ?

Thereafter, for I, II and IV, equate coefficients, use the first two boundary conditions of (1b) (which become $\sum_{m=0}^{n} a_m^{(r)} x_a^m = f(y_r)$ and $\sum_{m=0}^{n} a_m^{(r)} x_b^m = g(y_r)$ respectively) and easily obtain a system of linear algebraic equations of the form

n or n+1

$$A\,\underset{\sim}{a}^{(r+q)} = \underset{\sim}{k} \qquad (3)$$

for the required coefficients $a_0^{(r+q)},\ldots,a_{n+1}^{(r+q)}$. Note that the vector $\underset{\sim}{k}$ is a function of the solution on lines prior to the (q+r)-th. Explicitly, the matrices of (3) are :-

I

$$A = \left[\begin{array}{ccccccccc}
1 & x_a & x_a^2 & x_a^3 & \cdots\cdots & \cdots\cdots & \cdots\cdots & 0 & 0 \\
1 & x_b & x_b^2 & x_b^3 & \cdots\cdots & \cdots\cdots & \cdots\cdots & 0 & 0 \\
1 & 0 & 2\Delta y^2 & 0 & \cdots\cdots & & & -c_o^n\Delta y^2 & 0 \\
0 & 1 & 0 & 6\Delta y^2 & \cdots\cdots & & & -c_1^n\Delta y^2 & -c_o^n\Delta y^2 \\
 & & \cdots\cdots & \cdots\cdots & \cdots & & & & \\
0 & 0 & 0 & 0\cdots\cdots 1 & & 0 & \cancel{\Delta y^2} & -c_{n-1}^n\Delta y^2 & -c_{n-2}^n\Delta y^2 \\
0 & 0 & 0 & 0\cdots\cdots 0 & & 1 & 0 & -c_n^n\Delta y^2 & -c_{n-1}^n\Delta y^2 \\
0 & 0 & 0 & 0\cdots\cdots 0 & & 0 & 1 & 0 & -c_n^n\Delta y^2
\end{array}\right]$$

$n(n-1)\Delta y^2$

$$\underset{\sim}{k} = \left[f(y_r)\ ,\ g(y_r)\ ,\ 2a_o^{(r-1)}-a_o^{(r-2)},\ldots,2a_{n+1}^{(r-1)}-a_{n+1}^{(r-1)}\right]^T.$$

/ a_n

II

$$A = \left[\begin{array}{cccccccccc}
1 & x_a & x_a^2 & x_a^3\cdots\cdots & & & & & 0 & 0 \\
1 & x_b & x_b^2 & x_b^3\cdots\cdots & & & & & 0 & 0 \\
2 & 0 & 2\Delta y^2 & 0\ \cdots\cdots & & & & & -c_o^{(n)}\Delta y^2 & 0 \\
0 & 2 & 0 & 6\Delta y^2\cdots\cdots & & & & & -c_1^{(n)}\Delta y^2 & -c_o^{(n)}\Delta y^2 \\
 & & \cdots\cdots & \cdots\cdots & & & & & & \\
 & & & & & 2 & 0 & (n+1)n\Delta y^2 & -c_{n-1}^{(n)}\Delta y^2 & -c_{n-2}^{(n)}\Delta y^2 \\
 & & 0 & & & & 2 & 0 & -c_n^{(n)}\Delta y^2 & -c_{n-1}^{(n)}\Delta y^2 \\
 & & & & & & & 2 & 0 & -c_n^{(n)}\Delta y^2
\end{array}\right]$$

$$\underset{\sim}{k} = \begin{bmatrix} f(y_r) \\ g(y_r) \\ 5a_o^{(r-1)}-4a_o^{(r-2)}+a_o^{(r-3)} \\ 5a_1^{(r-1)}-4a_1^{(r-2)}+a_1^{(r-3)} \\ \\ 5a_{n-1}^{(r-1)}-4a_{n-1}^{(r-2)}+a_{n-1}^{(r-3)} \\ 5a_n^{(r-1)}-4a_n^{(r-2)}+a_n^{(r-3)} \\ 5a_{n+1}^{(r-1)}-4a_{n+1}^{(r-2)}+a_{n+1}^{(r-3)} \end{bmatrix}$$

IV

$$A = \left[\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2\Delta & 0 & 0 & 0 & -2\Delta c_o & 0 \\ 0 & 1 & 0 & 6\Delta\ldots\ldots & 0 & 0 & -2\Delta c_1 & -2\Delta c_o \\ & & \cdots & \cdots & & & & \\ & & & & 0 & (n-1)n\Delta & -2\Delta c_{n-1} & -2\Delta c_{n-2} \\ & 0 & & & 1 & 0 & -2c_n\Delta & -2c_{n-1}\Delta \\ & & & & 0 & 1 & 0 & -2\ c_n\Delta \end{array}\right]$$

$$\Delta = \Delta y^2/2$$

$$\underset{\sim}{k} = \begin{bmatrix} f(y_{r+1}) \\ g(y_{r+1}) \\ -2\Delta a_2^{(r-1)} -.\, a_o^{(r-1)} + 2a_o^{(r)} \\ -6\Delta a_3^{(r-1)} - a_1^{(r-1)} + 2a_1^{(r)} \\ -12\Delta a_4^{(r-1)} - a_2^{(r-1)} + 2a_2^{(r)} \\ \cdots\cdots\cdots\cdots\cdots \\ -n(n+1)\Delta a_{n+1}^{(r-1)} - a_{n-1}^{(r-1)} + 2a_{n-1}^{(r)} \\ -a_n^{(r-1)} + 2\ a_n^{(r)} \\ - a_{n+1}^{(r-1)} + 2a_{n+1}^{(r)} \end{bmatrix}$$

III, on the other hand, leads to an almost explicit expression of u_{r+1} in terms of u_r and u_{r-1}.

2.4 Error analyses : The errors incurred in perturbing I – IV by $(\tau_1^{(r+q)} + \tau_2^{(r+q)} x)\, T_n^*(x)$ may be analysed by defining $z_r = \tilde{u}_r - u_r$ and forming the difference between the perturbed and unperturbed forms, giving rise, in each case, to a system of difference-differential equations of the type

$$D\, z_r + E\, z_r = -(\tau_1^{(r+q)} + \tau_2^{(r+q)} x)\, T_n^*(x). \qquad (4)$$

For the methods I – III D and E represent the differential and difference operators respectively, while in the case of IV $D\, z_r = \frac{1}{2} \frac{d^2}{dx^2} \left[z_{r+1} + z_{r-1} \right]$ and E is again the difference operator associated with this case. The difference-differential equations generated by I, II and IV are of retarded type and can therefore be solved by the process of continuation. A few applications of this process enables one to guess at a solution, the correctness of which is easily checked by induction.

In each of the cases leading to a retarded equation the solution is of the form

$$z_p = \sum_{i=0}^{p} \beta_{ip}\, x^i \cos\alpha x + \sum_{i=0}^{p} \gamma_{ip}\, x^i \sin\alpha x -$$

$$- \sum_{i=1}^{p+1} \delta_{ip}\, \omega^{2(i-1)}\, \mathbb{D}^i\, (\tau_1^{(p)} + \tau_2^{(p)} x)\, T_n^*(x) \quad , \qquad (5)$$

where $\mathbb{D}$ represents, in each case, a different differential operator and where we have assumed that $z_{-i}=0$ for i positive. ω ?

The case I, in particular, starting from (5), where

now $\alpha = 1/\Delta y$,

and $\mathbb{D} = 1/(\frac{d^2}{dx^2} + \alpha^2)$,

leads straightforwardly to

$$\begin{aligned} z_{p+1} = \Bigg[& \sum_{i=0}^{p} \alpha^2 (2\beta_{ip} - \beta_{i,p-1}) \sum_{j=0}^{[\frac{i}{2}]} \frac{(-)^j i!}{(i+1-2j)!} \frac{x^{i+1-2j}}{(2\alpha)^{2j+1}} + \\ & + \sum_{i=0}^{p} \alpha^2 (2\gamma_{ip} - \gamma_{i,p-1}) \sum_{j=0}^{[\frac{i-1}{2}]} \frac{(-)^j i!}{(i-2j)!} \frac{x^{i-2j}}{(2\alpha)^{2j+2}} \Bigg] \sin(\alpha x) + \\ & + \Bigg[\sum_{i=0}^{p} \alpha^2 (2\beta_{ip} - \beta_{i,p-1}) \sum_{j=0}^{[\frac{i-1}{2}]} \frac{(-)^j i!}{(i-2j)!} \frac{x^{i-2j}}{(2\alpha)^{2j+2}} - \\ & - \sum_{i=0}^{p} \alpha^2 (2\gamma_{ip} - \gamma_{i,p-1}) \sum_{j=0}^{[\frac{i}{2}]} \frac{(-)^j i!}{(i+1-2j)!} \frac{x^{i+1-2j}}{(2\alpha)^{2j+1}} \Bigg] \cos(\alpha x) + \\ & + \sum_{i=2}^{p+1} \alpha^{2(i-1)} (2\delta_{i-1,p} - \delta_{i-1,p-1}) D^i P_{n+1} + D P_{n+1} + \\ & + \beta_{o,p+1} \cos(\alpha x) + \gamma_{o,p+1} \sin(\alpha x) \end{aligned} \tag{6}$$

$$\begin{aligned} = \sum_{i=0}^{p+1} \beta_{i,p+1} x^i \cos(\alpha x) + \sum_{i=0}^{p+2} \gamma_{i,p+1} x^i \sin(\alpha x) + \\ + \sum_{i=1}^{p+2} \alpha^{2(i-1)} \delta_{i,p+1} D^i P_{n+1}. \end{aligned}$$

The last expression after a suitable ordering of the terms of (6). The constants $\beta_{o,p+1}$ and $\gamma_{o,p+1}$ are determined by the boundary conditions $z_{p+1}(0) = z_{p+1}(1) = 0$. Obviously, a rather involved recurrence relationship may be established, linking $\beta_{i,p+1}$ and $\gamma_{i,p+1}$ to the previously computed values of these constants.

Similarly,

$$z_r = \sum_{k=0}^{r} \beta_{rk} x^k \cos\sqrt{2}\, x + \sum_{k=0}^{r} \gamma_{rk} x^k \sin\sqrt{2}x x + \sum_{k=1}^{r+1} \delta_{kr} D^k P_{n+1}$$

for II. The coefficients again being obtained recursively from those on the previous lines and the "zeroeth" ones coming from the boundary conditions. This time $D = 1/(\frac{d^2}{dx^2} + 2\alpha^2)$.

Much the same can be said for IV.

We later actually compute the numerical values of some of these errors.

The solution to the error equation III is obtained as follows using Euler-Laplace transforms :-

Let $\tilde{u}_r$ be the solution to III and u_r the solution to its perturbed form. Also define $z_r = \tilde{u}_r - u_r$.

Then $$\frac{d^2 z_r}{dx^2} + \frac{z_{r+1} - 2z_r + z_{r-1}}{\Delta y^2} = -(\tau_1 + \tau_2 x)\, T_n^*(x) \; . \qquad (7)$$

Now define $$Z_r(s) = \int_a^b e^{-sx} z_r(x)\, dx \qquad (8)$$

and $$W(s) = \int_a^b e^{-sx} w(x)\, dx, \qquad (9)$$

where $w(x) = -(\tau_1' + \tau_2' x)\, T_n^*(x)$

and $\tau_1' = \Delta y^2 \tau_1$, $\tau_2' = \Delta y^2 \tau_2$.

(7) may be written as (where $c = \Delta y^2$)

$$c \frac{d^2 z_r}{dx^2} + z_{r+1}(x) - 2z_r(x) + z_{r-1}(x) = -(\tau_1' + \tau_2' x) T_n^*(x) \quad . \qquad (10)$$

Taking Euler-Laplace transforms (see Bellman and Cooke [39] and Pinney [40]) on both sides of (10) and using

$$s^2 Z_r(s) = e^{-sa}\left[s\, z_r(a) + z_r'(a)\right] - e^{-sb}\left[s\, z_r(b) + z_r'(b)\right] + \\ + \int_a^b e^{-sx} z_r''(x)\, dx$$

we easily obtain

$$c\, s^2 . Z_r(s) + Z_{r+1}(s) - 2\, Z_r(s) + Z_{r-1}(s) = W(s) + \\ + c\, e^{-sa}\left[s\, z_r(a) + z_r'(a)\right] - c\, e^{-sb}\left[s\, z_r(b) + z_r'(b)\right]$$

i.e. $$Z_{r+1}(s) - 2\left(1 - \frac{c\, s^2}{2}\right) Z_r(s) + Z_{r-1}(s) = W(s) + \\ + c\, e^{-sa}\left[s\, z_r(a) + z_r'(a)\right] - c\, e^{-sb}\left[s\, z_r(b) + z_r'(b)\right] \qquad (11)$$

This difference equation has, as solution,

$$\begin{aligned} Z_r(s) = {} & Z_{\nu-1}(s)\, U_{[r]+1}\left(1-\frac{cs^2}{2}\right) - Z_{\nu-2}(s)\, U_{[r]}\left(1-\frac{cs^2}{2}\right) + \\ & + \sum_{k=0}^{[r]} \Big[c\, e^{-sa} \left\{ s\, z_{r-1-k}(a) + z'_{r-1-k}(a) \right\} - \\ & - c\, e^{-sb} \left\{ s\, z_{r-1-k}(b) + z'_{r-1-k}(b) \right\} + W(s) \Big]\, U_k\left(1-\frac{cs^2}{2}\right) \end{aligned} \qquad (12)$$

where $\nu = [r] - r$,

$[r]$ = largest integer $\leq r$,

$$U_r(z) = \frac{\sin(r+1)\arcsin z}{\sin \arccos z}$$

= Chebyshev Polynomial of the second kind.

It is easily checked that (12) is a solution to (11) by substitution and the subsequent use of the following properties of Chebyshev polynomials of the second kind (Lanczos [22]) :

$$\left.\begin{aligned} &(a)\quad U_r(x) - 2\,x\,U_{r-1}(x) + U_{r-2}(x) = 0 \\ &(b)\quad U_0(x) = 1 \\ &(c)\quad U_1(x) = 2x. \end{aligned}\right| \qquad (13)$$

Substituting we get

$$\begin{aligned} & Z_{r-[r]-1}(s)\, U_{[r]+2}\left(1-\frac{cs^2}{2}\right) - Z_{r-[r]-2}(s)\, U_{[r]+1}\left(1-\frac{cs^2}{2}\right) + \\ & + \sum_{k=0}^{[r]+1} \Big[c\, e^{-sa} \left\{ s\, z_{r-k}(a) + z'_{r-k}(a) \right\} - c\, e^{-sb} \left\{ s\, z_{r-k}(b) + z'_{r-k}(b) \right\} + \\ & + W(s) \Big]\, U_k\left(1-\frac{cs^2}{2}\right) - 2\left(1-\frac{cs^2}{2}\right) Z_{r-[r]-1}(s)\, U_{[r]+1}\left(1-\frac{cs^2}{2}\right) + \\ & + 2\left(1-\frac{cs^2}{2}\right) Z_{r-[r]-2}(s)\, U_{[r]}\left(1-\frac{cs^2}{2}\right) - 2\left(1-\frac{cs^2}{2}\right) \sum_{k=0}^{[r]} \Big[c\, e^{-sa} \left\{ s\, z_{r-k-1}(a) + \right. \\ & \left. + z'_{r-k-1}(a) \right\} - c\, e^{-sb} \left\{ s\, z_{r-1-k}(b) + z'_{r-1-k}(b) \right\} + W(s) \Big]\, U_k\left(1-\frac{cs^2}{2}\right) + \\ & + Z_{r-[r]-1}(s)\, U_{[r]}\left(1-\frac{cs^2}{2}\right) - Z_{r-[r]-2}(s)\, U_{[r]-1}\left(1-\frac{cs^2}{2}\right) + \\ & + \sum_{k=0}^{[r]-1} \Big[c\, e^{-sa} \left\{ s\, z_{r-2-k}(a) + z'_{r-2-k}(a) \right\} - c\, e^{-sb} \left\{ s\, z_{r-2-k}(b) + z'_{r-2-k}(b) \right\} + \\ & + W(s) \Big]\, U_k\left(1-\frac{cs^2}{2}\right) \\ & = c\, e^{-sa} \left[s\, z_r(a) + z'_r(a) \right] - c\, e^{-sb} \left[s\, z_r(b) + z'_r(b) \right] + W(s) \end{aligned}$$

(after some manipulation).

It is easily verified that

$$s^{\mu} Z_r(s) = e^{-sa}\sum_{k=0}^{\mu-1} z_r^{(k)}(a)\, s^{\mu-k-1} - e^{-sb}\sum_{k=0}^{\mu-1} z_r^{(k)}(b)\, s^{\mu-k-1} + \int_a^b e^{-sx}\, z_r^{(\mu)}(x)\, dx,$$

whence $P(s)\, Z_r(s) = \int_a^b e^{-sx}\, P(\frac{d}{dx})\, z_r(x)\, dx + e^{-sa}$(polynomial in s) +

$+\ e^{-sb}$(polynomial in s),

where P(s) is an arbitrary polynomial in s. Inserting this into (12) gives

$$Z_r(s) = \int_a^b e^{-sx}\Big[U_{[r]+1}(1-\frac{c}{2}\frac{d^2}{dx^2})\, z_{\nu-1}(x) - U_{[r]}(1-\frac{c}{2}\frac{d^2}{dx^2})\, z_{\nu-2}(x) + \sum_{k=0}^{[r]} U_k(1-\frac{c}{2}\frac{d^2}{dx^2})\, w(x)\Big]dx + e^{-sa}\, P_a(s) + e^{-sb}\, P_b(s) \qquad (14)$$

where $P_a(s)$ and $P_b(s)$ are polynomials in s.

Applying the <u>theorem</u>: If g(x) is integrable over (a,b) and is of bounded variation in some neighbouhood of x, then for G(s) defined by $G(s) = \int_a^b e^{-sx}\, g(x)\, dx$ and any constant c

$$\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty} G(s)\, e^{sx}\, ds \begin{array}{ll} = 0 & x < a \\ = \frac{1}{2}\, g(a+0) & x = a \\ = \frac{1}{2}\,[g(x+0) + g(x-0)] & a < x < b \\ = \frac{1}{2}\, g(b-0) & x = b \\ = 0 & x > b\ , \end{array}$$

we can invert (14) for $a \leqslant x \leqslant b$, $r \geqslant 0$, giving

$$z_r(x) = U_{[r]+1}(1 - \frac{c}{2}\frac{d^2}{dx^2})\, z_{\nu-1}(x) - U_{[r]}(1 - \frac{c}{2}\frac{d^2}{dx^2})\, z_{\nu-2}(x) + \sum_{k=0}^{[r]} U_k(1 - \frac{c}{2}\frac{d^2}{dx^2})\, w(x) + \frac{1}{2\pi i}\int_{x_0-i\infty}^{x_0+i\infty} e^{-sa}\, P_a(s)\, e^{sx}\, ds + \frac{1}{2\pi i}\int_{x_0-i\infty}^{x_0+i\infty} e^{-sb}\, P_b(s)\, e^{sx}\, ds \quad . \qquad (15)$$

The two integrals on the right may have their contours closed by adding the right hand and left hand semi-circles at infinity respectively. Since the integrands are analytic within these contours, the integrals vanish. Therefore, for $x \in [a,b]$, $r \geqslant 0$

$$z_r(x) = U_{[r]+1}(1 - \frac{c}{2}\frac{d^2}{dx^2})\; z_{\nu-1}(x) - U_{[r]}(1 - \frac{c}{2}\frac{d^2}{dx^2})\; z_{\nu-2}(x) +$$

$$+ \quad U_k(1 - \frac{c}{2}\frac{d^2}{dx^2})\; \varpi(x) \; .$$

If r takes on integral values only, then $\nu = r - [r] = 0$ and

$$z_{r}(x) = U_{r+1}(1 - \frac{c}{2}\frac{d^2}{dx^2})\; z_{-1}(x) - U_r(1 - \frac{c}{2}\frac{d^2}{dx^2})\; z_{-2}(x) +$$

$$+ \sum_{k=0}^{r} U_k(1 - \frac{c}{2}\frac{d^2}{dx^2})\; \varpi(x) \qquad . \qquad (16)$$

Assuming that $z_{-1} = z_{-2} = 0$ and using the definitions of $\varpi(x)$ and c:

$$z_r(x) = \sum_{k=0}^{r} U_k(1 - \frac{y^2}{2}\frac{d^2}{dx^2}\;)\; \Delta\, y^2\; (\tau_1 + \tau_2 x)\; T_n^*(x) \quad . \qquad (17)$$

Explicit realisations may be had for each r by utilising the expansions

$$\begin{aligned}
U_o(x) &= 1 \\
U_1(x) &= 2x \\
U_2(x) &= 4x^2 - 1 \\
U_3(x) &= 8x^3 - 4x \\
U_4(x) &= 16x^4 - 12x^2 + 1 \\
U_5(x) &= 32x^5 - 32x^3 + 6x \\
U_6(x) &= 64x^6 - 80x^4 + 24x^2 - 1
\end{aligned} \qquad (18)$$

....

Expanding (17) explicitly it is seen then that

$$z_o(x) = \Delta y^2\; (\tau_1 + \tau_2\, x)\; T_n^*(x)$$

$$z_1(x) = (1 - \frac{y^2}{2}\frac{d^2}{dx^2}\;)\,\Delta y^2\; (\tau_1 + \tau_2\, x)\; T_n^*(x) \qquad (19)$$

....

$$z_6(x) = (6 - \frac{21}{2}\, y^2\, \frac{d^2}{dx^2} + \frac{35}{4}\, y^4 \frac{d^4}{dx^4} - \frac{35}{8}\, y^6\, \frac{d^6}{dx^6} + \frac{21}{16}\, y^8 \frac{d^8}{dx^8} - \frac{7}{32}\, y^{10} \frac{d^{10}}{dx^{10}} +$$

$$+ \frac{1}{64}\, y^{12} \frac{d^{12}}{dx^{12}}\;)\,\Delta y^2\; (\tau_1 + \tau_2\, x)\; T_n^*(x)$$

....

It follows immediately from these explicit expressions that one should endeavour to use a small Δy coupled with a low n. This is so since $T_n^*(x)$ is a polynomial of degree n and hence the higher its order the more non-zero terms occur in $z_i(x)$ - in general these non-zero terms do not cancel each other out, but rather combine to swell the magnitude of the error.

2.5 Numerical results : The impracticability of solving Laplace's equation by the techniques I - IV described above is vividly illustrated by computing numerical values for some of the previous error expressions. Tables 1 and 2 show the magnitudes of these errors (an entry $\hat{n}$ here means an error having magnitude $10^{\hat{n}}$) for the cases I and II with boundary conditions f(y)=h(x)=k(x)=0 and g(y)=1, with a Chebyshev perturbation of degree 19 and steplength Δy = 1/8 in both cases. IV produces similar errors to these, while the large errors obtained from III are easily seen by referring to (19).

y	x								
	0	.125	.250	.375	.500	.625	.750	.875	1.00
.125	0.	0.	0.	0.	0.	0.	0.	0.	0.
.250	0.	0.	0.	0.	0.	0.	0.	0.	0.
.375	19	27	28	27	28	28	28	28	29
.500	3	37	37	38	38	38	38	38	38
.625	9	9	9	9	8	9	9	9	7
.750	13	42	42	41	42	43	42	43	43
.875	18	17	17	17	16	17	18	18	17
1.000	22	26	26	26	27	27	27	27	28

I ; n=19; Δy=1/8

table 1

Table 1 ?

These rather large errors are easily confirmed by actually computing the relevant approximate solutions, we show those associated with the above error tables in tables 3 and 4. No discernable improvement was obtained by decreasing the step size. Similar results are obtained, too, with perturbations of different degree.

Some comments on the computation of the approximate solution are necessary. In each of the above techniques a knowledge of the coefficients a_i on lines prior to the r-th is necessary in order to compute the $a_i^{(r)}$. Following a technique described by Fox [9]p58-63, values were selected for these coefficients on the first lines and

y					x				
	.000	.125	.250	.375	.5C0	.625	.750	.875	1.00
.125	0.	0.	0.	0.	0.	0.	0.	0.	0.
.250	0.	0.	0.	0.	0.	0.	0.	0.	0.
.375	5	22	22	22	22	22	22	21	22
.500	13	23	23	23	23	23	23	24	22
.625	20	24	23	24	24	24	24	24	23
.750	27	27	28	27	27	27	27	28	27
.875	34	34	35	34	34	34	34	35	35
1.000	41	41	42	41	41	41	41	41	42

II : n = 19 : Δy = 1/8

table 1

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.000	.0	.0	.0	.0	.0	.0	.0	.0	.0
.125	13	13	13	13	13	13	13	13	13
.250	.0	−1	00	−1	00	00	00	00	11
.375	19	19	19	19	19	19	19	19	19
.500	.0	2	3	2	3	3	3	3	4
.625	5	5	5	5	4	5	5	4	5
.750	19	19	19	19	19	19	19	19	19
.875	1	7	8	7	8	8	8	8	8
1.000	9	9	9	9	−7	9	10	9	10

I

table 2

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.000	.0	.0	.0	.0	.0	.0	.0	.0	.0
.125	36	36	36	36	36	36	36	36	36
.250	.0	−3	−3	−3	−15	−2	−3	−2	−2
.375	18	18	18	18	18	18	18	18	18
.500	.0	2	3	3	3	4	5	6	7
.625	7	7	7	7	7	7	6	8	10
.750	8	8	8	8	8	8	9	11	12
.875	−1	9	10	10	10	10	11	13	14
1.000	12	12	13	13	−4	12	14	15	16

II

table 3

hence several solutions were constructed – their number being equal to the number of sets of coefficients required to compute the solution on the r-th line. The constructed solutions on the final line were then combined in order to satisfy the boundary condition there – the solution on other lines was then obtained by combining the previously computed solutions there in the same way.

2.6 Comment : Mason, in solving a similar problem using an approximate solution of the form $u(x,y) = \sum_{r,s=0}^{n} a_{rs}T_r(x)T_s(y)$, obtained satisfactory results. The reason for this is that he simultaneously applied all the boundary conditions, while we have initially applied three of these conditions and have attempted to satisfy the fourth at a later stage – obviously without any success. In the next chapter we give a modified version of this technique which works remarkably well.

2.7 The perturbed forms of the equations I , II and IV may be solved using the "method of selected points" or the prior integration method (Fox and Parker [10]). These techniques of solution do not alter the given error analyses.

(a) Consider first the method of selected points (collocation):

(i) Assuming the solution of the perturbed form of I to be $u_r = \sum_{m=0}^{n} a_m^{(r)} x^m$. Substitute this into the perturbed equation and then satisfy the equation at the zeros of $T_n^*(x)$, i.e. at $x_k = \{1 + \cos(2k-1)\pi/2n\}/2$, k=1,2,..,n. As before the boundary conditions on the r-th line are $u_r(x_a) = f(y_r)$ and $u_r(x_b)=g(y_r)$. The coefficients $a_m^{(r)}$ are obtained by solving

$$\begin{bmatrix} 1 & x_a & x_a^2 & x^3 & x_a^4 & \cdots\cdots & x_a^{n+1} \\ 1 & x_b & x_b^2 & x_b^3 & x_b^4 & & x_b^{n+1} \\ 1 & x_1 & (2\,y^2+x_1^2) & (6\,y^2+x_1^2)x_1 & (12\,y^2+x_1^2)x_1^2 & \ldots((n+1)n\,y^2+x_1^2)x_1^{n-1} & \\ & & & \cdots\cdots\cdots & & & \\ 1 & x_n & (2\,y^2+x_n^2) & (6\,y^2+x_n^2)x_n & (12\,y^2+x_n^2)x_n^2 & \ldots((n+1)n\,y^2+x_n^2)x_n^{n-1} & \end{bmatrix}$$

$$\begin{bmatrix} a_o^{(r)} \\ a_1^{(r)} \\ a_2^{(r)} \\ \cdots \\ a_{n+1}^{(r)} \end{bmatrix} = \begin{bmatrix} f(y_r) \\ g(y_r) \\ 2a_m^{(r-1)}-a_m^{(r-2)} \quad x_1^m \\ \cdots\cdots \\ 2a_m^{(r-1)}-a_m^{(r-2)} \quad x_n^m \end{bmatrix} \qquad (20)$$

Typical of the coefficient matrices which arise is that for $\Delta y=1/8$ and $n=8$; correct to 2S this matrix is

$$\begin{bmatrix} 1.0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 \\ 1.0 & .99 & 1.1 & 1.2 & 1.3 & 1.4 & 1.6 & 1.8 & 2.0 & 2.2 \\ 1.0 & .92 & .93 & .94 & .97 & 1.0 & 1.1 & 1.1 & 1.2 & 1.2 \\ 1.0 & .78 & .70 & .62 & .56 & .51 & .46 & .42 & .38 & .35 \\ 1.0 & .60 & .45 & .33 & .24 & .18 & .13 & .94_{-1} & .67_{-1} & .48_{-1} \\ 1.0 & .40 & .26 & .14 & .77_{-1} & .41_{-1} & .21_{-1} & .11_{-1} & .55_{-2} & .27_{-2} \\ 1.0 & .22 & .14 & .53_{-1} & .18_{-1} & .57_{-2} & .17_{-2} & .50_{-3} & .14_{-3} & .39_{-4} \\ 1.0 & .84_{-1} & 1.0 & .16_{-1} & .23_{-2} & .28_{-3} & .33_{-4} & .37_{-5} & .41_{-6} & .43_{-7} \\ 1.0 & .96_{-2} & .94_{-1} & .18_{-2} & .29_{-4} & .42_{-6} & .56_{-8} & .72_{-10} & .88_{-12} & .11_{-13} \end{bmatrix}$$

From the wide range of coefficient sizes it is immediately apparent that with this choice of n and Δy, the system (20) is ill-conditioned. Evaluating the determinant confirms this. The following table is interesting:

n	Δy	determinant
6	.50	$-.2167_{-1}$
8	.25	$.2604_{-10}$
12	.25	$-.1185_{-25}$
4	.125	$-.8724_{-5}$
8	.125	$.1475_{-15}$
8	.0625	$.4158_{-19}$

table 4

This should not be totally surprising when the values of the x_k's are remembered. Hence a solution by this means will produce contaminated results.

(ii) II leads to the matrix equation $A\underset{\sim}{a} = \underset{\sim}{b}$. The matrix A being that of (i) above, except that terms $(s\Delta y^2+x_i^2)$ there are replaced by $(s\Delta y^2+2x_i^2)$ here; the vector $\underset{\sim}{a}$ is the previous unknown vector

and $$\underset{\sim}{b} = \begin{bmatrix} f(y_r) \\ g(y_r) \\ \sum_{m=0}^{n+1}\left[5a_m^{(r-1)} - 4a_m^{(r-2)} + a_m^{(r-3)}\right] x_1^m \\ \cdots\cdots\cdots \\ \sum_{m=0}^{n+1}\left[5a_m^{(r-1)} - 4a_m^{(r-2)} + a_m^{(r-3)}\right] x_n^m \end{bmatrix} .$$

As this coefficient matrix is similar to that of (i), the conclusions of that section apply here too.

(iii) The coefficient matrix arising from the application of a collocation method to II is just that of (ii) above, while the right hand side vector is

$$\begin{bmatrix} f(y_r) \\ g(y_r) \\ \sum_{m=0}^{n+1}\left[4a_m^{(r-1)} - 2a_m^{(r-2)}\right] x_1^m - \Delta y^2 \sum_{m=2}^{n+1} m(m-1)a_m^{(r-2)}\, x_1^m \\ \cdots\cdots\cdots \\ \sum_{m=0}^{n+1}\left[4a_m^{(r-1)} - 2a_m^{(r-2)}\right] x_n^m - \Delta y^2 \sum_{m=2}^{n+1} m(m-1)a_m^{(r-2)}\, x_n^m \end{bmatrix} .$$

Again the conclusions of (i) apply.

(b) Each of the semi-discretized sets of the previous sections may, following Clenshaw [38], be integrated first. An infinite Chebyshev expansion may then be assumed for u_r and the coefficients of the Chebyshev expansion may be obtained from a backward iteration on the difference equations obtained by equating coefficients.

(i) As an example, consider the integrated form of I, namely

$$\Delta y^2 u_r + \iint (u_r - 2u_{r-1} + u_{r-2})\, dx\, dx = 0. \qquad (21)$$

Assume that $u_r(x) = \sum_{m=0}^{\infty}{}' a_m^{(r)} T_m^*(x)$, (22)

where the dash indicates, in the usual way, that half the first coefficient should be taken. Substitute (22) into (21) and use

$$\int T_0^*(x)\, dx = \tfrac{1}{2}(T_1^*(x) + T_0^*(x))$$

$$\int T_1^*(x)\, dx = (T_2^*(x) - T_0^*(x))/8$$

$$\int T_m^*(x)\, dx = \tfrac{1}{4}\left(\frac{T_{m+1}^*(x)}{m+1} - \frac{T_{m-1}^*(x)}{m-1} \right),\ m=2,3,\ldots$$

twice. This leads to the set of difference equations:-

$$\frac{\Delta y^2}{2} a_0^{(r)} + \left[\frac{3}{32} a_0^{(r)} - \frac{1}{16} a_1^{(r)} + \frac{1}{32} a_2^{(r)}\right] - 2\left[\frac{3}{32}a_0^{(r-1)} - \frac{1}{16}a_1^{(r-1)} + \frac{1}{32}a_2^{(r-1)}\right] +$$

$$+\left[\frac{3}{32}a_0^{(r-2)} - \frac{1}{16}a_1^{(r-2)} + \frac{1}{32}a_2^{(r-2)}\right] = 0 \qquad (23)$$

$$\Delta y^2 a_1^{(r)} + \left[\frac{1}{8} a_0^{(r)} - \frac{3}{32} a_1^{(r)} + \frac{1}{32} a_2^{(r)}\right] - 2\left[\frac{1}{8}a_0^{(r-1)} - \frac{3}{32}a_1^{(r-1)} + \frac{1}{32}a_2^{(r-1)}\right] +$$

$$+\left[\frac{1}{8} a_0^{(r-2)} - \frac{3}{32} a_1^{(r-2)} + \frac{1}{32} a_2^{(r-2)}\right] = 0 \qquad (24)$$

$$\Delta y^2 a_2^{(r)} + \left[\frac{1}{32} a_0^{(r)} - \frac{5}{96}a_2^{(r)} + \frac{1}{48}a_3^{(r)}\right] - 2\left[\frac{1}{32}a_0^{(r-1)} - \frac{5}{96}a_2^{(r-1)} + \frac{1}{48}a_3^{(r-1)}\right] +$$

$$+\left[\frac{1}{32}a_0^{(r-2)} - \frac{5}{96}a_2^{(r-2)} + \frac{1}{48}a_3^{(r-2)}\right] = 0 \qquad (25)$$

$$\Delta y^{2} a_m^{(r)} + \frac{1}{16}\left[\frac{a_{m-2}^{(r)} - a_{m-1}^{(r)}}{m-1} - \frac{a_m^{(r)} - a_{m+1}^{(r)}}{m+1}\right] - \frac{1}{8}\left[\frac{a_{m-2}^{(r-1)} - a_{m-1}^{(r-1)}}{m-1}\right.$$

$$\left. - \frac{a_m^{(r-1)} - a_{m+1}^{(r-1)}}{m+1}\right] + \frac{1}{16}\left[\frac{a_{m-2}^{(r-2)} - a_{m-1}^{(r-2)}}{m-1}\right.$$

$$\left. - \frac{a_m^{(r-2)} - a_{m+1}^{(r-1)}}{m+1}\right] = 0 \qquad . \qquad (26)$$

$$m = 3,4,\dots$$

Using the backward recurrence device (Fox and Parker [10] p 99) we take

$$a_{n-1}^{(r)} = 1 \ , \ a_n^{(r)} = a_{n+1}^{(r)} = \dots = 0$$

$$a_n^{(r)} = 1 \ , \ a_{n+1}^{(r)} = a_{n+2}^{(r)} = \dots = 0 \qquad (27)$$

$$a_{n+1}^{(r)} = 1 \ , \ a_{n+2}^{(r)} = a_{n+3}^{(r)} = \dots = 0$$

(in turn) for some large n to obtain three independent solutions. Let the two general solutions be called I_r and II_r and the particular solution III_r. The solution is given by the linear combination

$$A_1^{(r)}\,(I_r) + A_2^{(r)}\,(II_r) + III_r \qquad . \qquad (28)$$

The constants A_1 and A_2 are found from the as yet unused boundary conditions, namely

$$u_r(x_a) = f(y_r)$$

$$u_r(x_b) = g(y_r) \qquad . \qquad (29)$$

(ii) A similar procedure may be followed for solving II and IV .

2.8 Another approach to the solution of the given problem is via the Lanczos-Ortiz canonical polynomial theory.

Following Ortiz the canonical polynomials for the problem I are

$$Q_r(x) = \sum_{i=0}^{[r/2]} (-)^i \frac{r!}{(r-2i)!} \Delta y^{2i+2}\, x^{r-2i} \qquad . \qquad (30)$$

The solution to the perturbed differential equation must satisfy

$$\left(\frac{d^2u_r}{dx^2} + \frac{u_r}{y^2} \right) = \frac{1}{\Delta y^2} (2u_{r-1} - u_{r-2}) + (\tau_1^{(r)} + \tau_2^{(r)} x) \, T_n^*(x)$$

together with the boundary conditions

$$u_r(x_a) = f(y_r) \text{ and } u_r(x_b) = g(y_r).$$

Hence the solution is

$$u_r(x) = \frac{2}{\Delta y^2} \sum_{m=0}^{n+1} a_m^{(r-1)} Q_m(x) - \frac{1}{\Delta y^2} \sum_{m=0}^{n+1} a_m^{(r-2)} Q_m(x) +$$

$$+ \sum_{m=0}^{n} c_m^{(n)} \left(\tau_1^{(r)} Q_m(x) + \tau_2^{(r)} Q_{m+1}(x) \right) \qquad (31)$$

$\tau_1^{(r)}$ and $\tau_2^{(r)}$ are obtained by making the solution $u_r(x)$ satisfy the boundary conditions. Once the taus have been computed the solution is

$$u_r(x) = \left[\frac{2}{\Delta y^2} a_o^{(r-1)} - \frac{1}{\Delta y^2} a_o^{(r-2)} + c_o^{(n)} \tau_1^{(r)} \right] Q_o(x)$$

$$+ \sum_{m=1}^{n} \left[\frac{2}{\Delta y^2} a_m^{(r-1)} - \frac{1}{\Delta y^2} a_m^{(r-2)} + c_m^{(n)} \tau_1^{(r)} + c_{m-1}^{(n)} \tau_2^{(r)} \right] Q_m(x)$$

$$+ \left[\frac{2}{\Delta y^2} a_{n+1}^{(r-1)} - \frac{1}{\Delta y^2} a_{n+1}^{(r-2)} + c_n^{(n)} \tau_2^{(r)} \right] Q_{n+1}(x) \qquad (32)$$

CHAPTER 3 : MATRIX LINES-TAU METHOD

3.1 Introduction : The previous attempt at solving Laplace's equation by a combination of the lines and tau methods failed, as we have previously pointed out, because of the manner in which the boundary conditions were used. We here describe, what we have termed, a matrix lines-tau method in which we impose the boundary conditions even from the beginning of the procedure. The matrix part in the name comes from the vector canonical polynomials which we define.

In this chapter we describe the technique as applied to the solution of Laplace's equation on regions as in figure 1. An extension to Poisson's equation is also given. Some numerical results are given for both types of equation. We give an error analysis and an extension to the eigenvalue problem in chapters 4 and 5 respectively. It is shown also that fairly complex boundary conditions may be handled successfully.

3.2 Method applied to Laplace's equation : We consider now the equation of Laplace on the curvilinear trapezium of figure 1. The domain boundary consists of the segments AB and CD of

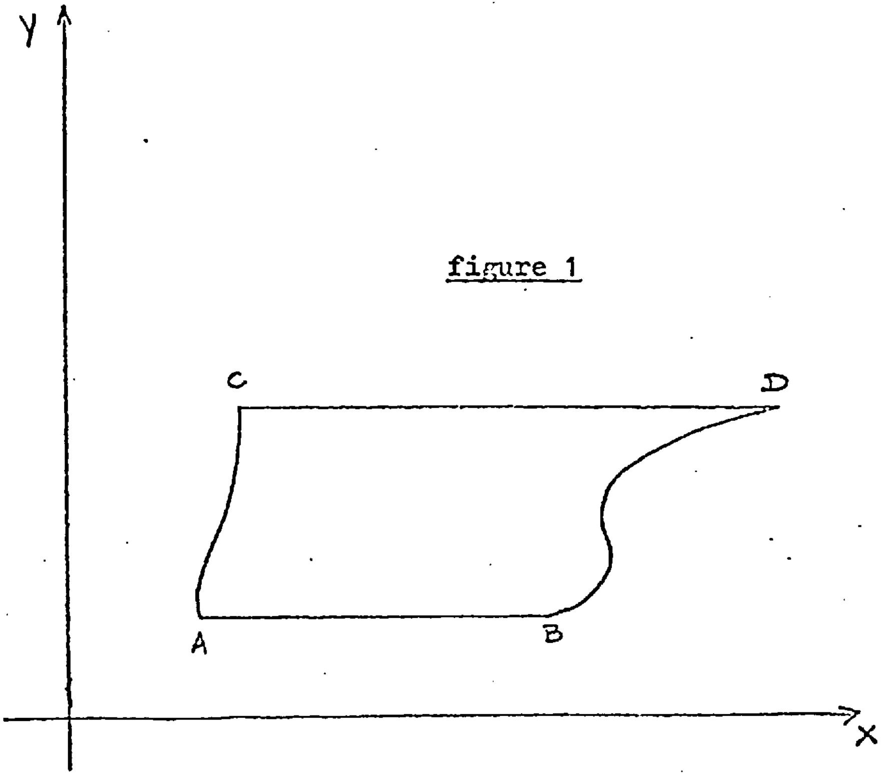

figure 1

straight lines parallel to the OX axis and of arcs AC and BD which each intersect any straight line parallel to OX in at most one point. Consider Laplace's equation

$$\left.\begin{array}{l} \nabla^2 u = 0 \\ \text{with boundary condition } u=0 \text{ on AB and CD ,} \\ \qquad u = f(x,y) \text{ on AC} \\ \qquad u = g(x,y) \text{ on BD.} \end{array}\right| \qquad (1)$$

Again equally spaced lines are drawn parallel to OX (interval between them being h). Denote these lines by $y=y_0$, $y=y_1$, $\dots$,$y=y_{n+1}$, where y_0 and y_{n+1} coincide with AB and CD respectively. Introduce the notation

$$u_k(x) = u(x,y_k)\ ,\ f_k(x) = f(x,y_k)\ ,\ g_k(x) = g(x,y_k) \qquad (2)$$

$$k=0,1,\dots,n+1.$$

Let the arc AC cut the lines $y=y_i$ $i=0(1)n+1$ at the points $(\bar{x}_i,y_i)$ and let the arc BD cut the lines $y=y_i$ at $(\bar{\bar{x}}_i,y_i)$.

Define $\bar{X} = \left[\bar{x}_1, \bar{x}_2,\dots,\bar{x}_n\right]^T$ and $\bar{\bar{X}} = \left[\bar{\bar{x}}_1, \bar{\bar{x}}_2,\dots,\bar{\bar{x}}_n\right]^T$. (3)

Mikhlin [22] , quoting Faddeeva and Slobedyansky, shows that the problem (1) may be approximated by the system of ordinary differential equations

$$\frac{5}{6}u''_k(x) + \frac{1}{12}\left[u''_{k+1} + u''_{k-1}\right] + \frac{1}{h^2}\left[u_{k+1} - 2u_k + u_{k-1}\right] + \qquad (4)$$

$$+ O(h^4) = 0 \qquad k=1,\dots,n\ .$$

Along the boundaries AB and CD $u_0(x) = u_{n+1}(x) = u''_0(x) = u''_{n+1}(x) = 0.$ (5)

This may be checked by considering the variables separable solution.

Ignoring the error term in (4), combining these equations and utilising (5) leads to the equation

$$A'\ U'' + \frac{M}{h^2}\ U = 0, \qquad (6)$$

where $U(X) = \left[u_1(x),\ u_2(x),\dots,\ u_n(x)\right]^T$

$0 \quad = \left[0,\ 0,\dots,0\right]^T$

$A' = (a'_{ij})$, $a'_{ii}=5/6$, $a'_{i,i+1} = a'_{i+1,i} = 1/12$;

$M = (m_{ij})$, $m_{ii} = -2$, $m_{i,i+1} = m_{i+1,i} = 1$.

A' and M are both of order nxn. The boundary conditions associated with (6) are

$$U(\bar{X}) = F(\bar{X}) \qquad (7)$$

and $U(\bar{\bar{X}}) = G(\bar{\bar{X}})$, (8)

where $F(\bar{X}) = \left[f_1(\bar{x}_1),\dots,f_n(\bar{x}_n)\right]^T$

$G(\bar{\bar{X}}) = \left[g_1(\bar{\bar{x}}_1),\dots,g_n(\bar{\bar{x}}_n)\right]^T$.

Let $b = \max\limits_{0\le i\le n+1} (\bar{x}_i,\bar{\bar{x}}_i)$ and $a = \min\limits_{0\le i\le n+1} (\bar{x}_i,\bar{\bar{x}}_i)$.

By means of the linear transformation $\xi = \frac{1}{b-a}(x-a)$

it can be ensured that all of the lines $y_i(x)$ i=0(1)n+1 lie within $[0,1]$. Now let $A = \frac{1}{(b-a)^2} A'$. In what follows we will continue

to denote the boundary points, namely $\bar{\xi}_i = \frac{\bar{x}_i-a}{b-a}$ and $\bar{\bar{\xi}}_i = \frac{\bar{\bar{x}}_i-a}{b-a}$, i=0,..,n+1, by $\bar{x}_i$ and $\bar{\bar{x}}_i$. Also, we will still denote the independent variable in the transformed equation by x. The transformed equations are still of the form

$$A\,U'' + \frac{M}{h^2}\,U = 0.$$

Define the matrix differential operator D by $D = A\frac{d^2}{dx^2} + \frac{1}{h^2}M$. The field of definition of D is the set of all nx1 vectors with twice differentiable elements. Define the nx1 vectors

$$X^m = \left[x^m,\ x^m,\dots,x^m\right]^T \qquad m=0,1,\dots$$

and $TT^*_N(X) = \left[T^*_N(x),\ T^*_N(x),\dots,T^*_N(x)\right]^T$, $T^*_N(x)$ is the shifted Chebyshev polynomial of the first kind of degree N and $T^*_N(x) = \sum_{m=0}^{N} c_m^{(N)} x^m$. Thus $TT^*_N(X) = \sum_{m=0}^{N} c_m^{(N)} X^m$.

Let $\tau = \mathrm{diag}(\tau_1,\tau_2,\dots,\tau_n)$.

Following Lanczos [20] define vectors $Q_m(X) = \left[q_{m1}(x),\dots,q_{mn}(x)\right]^T$ such that $D\,Q_m(X) = X^m$. (9)

Now $D X^m = m(m-1) A X^{m-2} + \frac{M}{h^2} X^m \quad m=2,3,..$ (10)

hence $Q_m = h^2 M^{-1} \left[X^m - m(m-1) A Q_{m-2} \right]$ (11)

and, from this, it easily follows that

$$Q_m = h^2 M^{-1} \sum_{i=0}^{[m/2]} \frac{(-)^i m!}{(m-2i)!} h^{2i} S^i X^{m-2i} \tag{12}$$

where $S^i = (AM^{-1})^i$,i=1,2,.. and $S^o = I$ (the unit matrix).

Perturb (6) by $\tau' \, TT_N^*(X) + \tau'' \, TT_{N+1}^*(X)$ to give

$$D U = \tau' \, TT_N^*(X) + \tau'' \, TT_{N+1}^*(X). \tag{13}$$

The solution to this perturbed equation is obviously

$$\begin{aligned} U(X) &= \tau' \sum_{m=0}^{N} c_m^{(N)} Q_m(X) + \tau'' \sum_{m=0}^{N+1} c_m^{(N+1)} Q_m(X) \\ &= \tau' \sum_{m=0}^{N} c_m^{(N)} h^2 M^{-1} \sum_{i=0}^{[m/2]} \frac{(-)^i m!}{(m-2i)!} h^{2i} S^i X^{m-2i} + \\ &\quad + \tau'' \sum_{m=0}^{N+1} c_m^{(N+1)} h^2 M^{-1} \sum_{i=0}^{[m/2]} \frac{(-)^i m!}{(m-2i)!} h^{2i} S^i X^{m-2i} . \end{aligned} \tag{14}$$

In a computationally more convenient form this is

$$\begin{aligned} U(X) &= h^2 \tau' M^{-1} \sum_{i=0}^{[N/2]} (-)^i h^{2i} S^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} X^{m-2i} + \\ &\quad + h^2 \tau'' M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} S^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} X^{m-2i} \end{aligned} \tag{15}$$

Applying the boundary condition (7) to (15) yields

$$\begin{aligned} & h^2 \tau' M^{-1} \sum_{i=0}^{[N/2]} (-)^i h^{2i} S^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i} + \\ & + h^2 \tau'' M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} S^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i} = F(\bar{X}) . \end{aligned} \tag{16}$$

The condition (8) leads to a similar result. These two boundary conditions are of the form

$$\begin{aligned} \tau' K + \tau'' L &= F \\ \tau' P + \tau'' R &= G \end{aligned} \tag{17}$$

in which $K = [k_1, \dots, k_n]^T$

$$= h^2 M^{-1} \sum_{i=0}^{[N/2]} (-)^i h^{2i} S^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} X^{m-2i}$$

$\overline{X}$./

and similarly for $L = [l_1, \dots, l_n]^T$,

$$P = [p_1, \dots, p_n]^T ,$$
$$R = [r_1, \dots, r_n]^T .$$

Equations (17) are equivalent to the (2n)x(2n) system

$$\begin{bmatrix} k_1 & & & l_1 & & \\ & k_2 & & & l_2 & \\ & & \ddots \; k_n & & & \ddots \; l_n \\ p_1 & & & r_1 & & \\ & p_2 & & & r_2 & \\ & & \ddots \; p_n & & & \ddots \; r_n \end{bmatrix} \begin{bmatrix} \tau_1' \\ \tau_2' \\ \vdots \\ \tau_n' \\ \tau_1'' \\ \tau_2'' \\ \vdots \\ \tau_n'' \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \\ g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix} \qquad (18)$$

which may easily be solved for the τ_i' and τ_i'' .

3.3 As a special case consider the square region ABCD with $0 \leq x \leq 1$, $0 \leq y \leq 1$. We will solve the boundary value problems defined by the following sets of boundary conditions:

a) u=0 on AB, CD and AC, u=1 on BD;

b) u=0 on AB, CD and AC, $u=\sin(\pi y)$ on BD;

c) u=0 on AB, CD and AC, $u=\cos(\pi y)$ on BD.

The boundary conditions a) and c) are discontinuous, while b) is continucus. In each of these cases $\overline{X} = [0, 0, \dots, 0]^T$, $\overline{\overline{X}} = [1, 1, \dots, 1]^T$ and $F = [0, 0, \dots, 0]^T$, while for

a) $G = [1, 1, \dots, 1]^T$,

b) $G = [\sin(\pi h), \sin(2\pi h), \dots, \sin(n\pi h)]^T$,

c) $G = [\cos(\pi h), \cos(2\pi h), \dots, \cos(n\pi h)]^T$.

We will denote the above all by $G(\mathbb{I})$.

The boundary condition (16) now reduces to

$$\tau' M^{-1} \sum_{i=0}^{[N/2]} (-)^i h^{2i} (2i)!\, c_{2i}^{(N)} S^i \mathbb{I} + \tau'' M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} (2i)!\, c_{2i}^{(N+1)} S^i \mathbb{I} = 0 \qquad (19)$$

where $\mathbb{I} = [1, 1, \dots, 1]^T$ and $0 = [0, 0, \dots, 0]^T$.

The other boundary condition is

$$h^2\tau' M^{-1}\sum_{i=0}^{[\frac{N}{2}]} (-)^i\, h^{2i}\, S^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!}\, \Phi \; +$$

$$+h^2\tau'' M^{-1}\sum_{i=0}^{[(N+1)/2]} (-)^i\, h^{2i}\, S^i \sum_{m=2i}^{N} c_m^{(N+1)} \frac{m!}{(m-2i)!}\, \Phi = G(\Phi) \quad . \qquad (20)$$

Each of the problems defined by a) - c) and two others (defined later) were solved for N=7(1)13 and with h=0.25 and 0.125 . However only the results for the cases N=7, h=0.25 and h=0.125 and N=13, h=0.25 are reproduced. An exception is made in the case of a) where the result for N=7 h=0.0625 also appears.

3.4 Numerical results : The results tabulated in sub-sections (a)-(c) correspond to the problems (a) - (c) of the previous section.

Section (a)

We tabulate first the solution obtained by the usual separation of the variables technique for comparison :-

	x - values								
y	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.00000	0000	0000	0000	0000	0000	0000	0000	0000	0.0000
.03125	0000	0044	0094	0159	0249	0377	0565	084	0.1284
.06250	0000	0087	0187	0316	0495	0751	1124	1674	0.2484
.09375	0000	0129	0278	0470	0736	1118	1673	2490	0.3696
.12500	0000	0170	0366	0620	0971	1473	2206	3283	0.4872
.15625	0000	0209	0452	0764	1196	1815	2717	4044	0.6002
.18750	0000	0247	0532	0900	1410	2139	3202	4766	0.7074
.21875	0000	0282	0608	1028	1610	2442	3657	5442	0.8077
.25000	0000	0314	0677	1146	1794	2722	4076	6066	0.9003
.28125	0000	0343	0740	1253	1961	2976	4455	6631	0.9842
.31250	0000	0369	0796	1348	2110	3201	4792	7132	1.059
.34375	0000	0392	0845	1429	2238	3395	5083	7565	1.123
.37500	0000	0410	0845	1497	2344	3557	5325	7925	1.176
.40625	0000	0425	0916	1551	2428	3684	5516	8209	1.218
.43750	0000	0436	0939	1590	2488	3776	5653	8413	1.249
.46875	0000	0442	0953	1613	2525	3831	5736	8537	1.267
.50000	0000	0444	0958	1621	2537	3850	5764	8578	1.273
.53125	0000	0442	0953	1613	2525	3831	5736	8537	1.267
				....					

(symmetric about y=.50000)

table 1

h =.25 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	0000	0350	0754	1275	1995	3026	4529	6740	1.0000
.50	0000	0349	0753	1275	1995	3027	4530	6740	1.0000
.75	0000	0350	0754	1275	1995	3026	4529	6740	1.0000

τ'^{T}	$-.1923_{-3}$	$-.1360_{-3}$	$-.1923_{-3}$
τ''^{T}	$-.1706_{-4}$	$-.1206_{-4}$	$-.1706_{-4}$

table 2

As the solution is symmetric about y=0.5 we only show $\frac{1}{2}$ the computed values in the next tables.

h = .125 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	0000	0349	0753	1273	1992	3023	4526	6736	1.0000
.250	0000	0349	0752	1273	1993	3024	4527	6737	1.0000
.375	0000	0349	0752	1273	1993	3024	4527	6737	1.0000
.500	0000	0349	0752	1273	1993	3024	4527	6737	1.0000

τ'^{T}	$-.3562_{-3}$	$-.1928_{-3}$	$-.1476_{-3}$	$-.1364_{-3}$	$-.1476_{-3}$	$-.1928_{-3}$	$-.3562_{-3}$
τ''^{T}	$-.3163_{-4}$	$-.1712_{-4}$	$-.1310_{-4}$	$-.1210_{-4}$	$-.1310_{-4}$	$-.1712_{-4}$	$-.3163_{-4}$

table 3

h = .0625 N=7

On all the lines the approximate values (to 4D) are:

0000 0349 075(2 or 3) 1273 1992 3023 452(6 or 7) 6736 1.0000

τ'^{T}	$-.6988_{-3}$	$-.3563_{-3}$	$-.2454_{-3}$	$-.1929_{-3}$	$-.1640_{-3}$	$-.1476_{-3}$	$-.1391_{-3}$	$-.1364_{-3}$
τ''^{T}	$-.6205_{-4}$	$-.3163_{-4}$	$-.2179_{-4}$	$-.1712_{-4}$	$-.1456_{-4}$	$-.1310_{-4}$	$-.1234_{-4}$	$-.1211_{-4}$

table 4

h = .125 N= 13

To 4D these values are all:

0000 0349 0753 1275 1995 3026 4530 6739 1.0000

τ'^{T}	$-.3458_{-10}$	$-.2445_{-10}$	$-.3458_{-10}$
τ''^{T}	$-.1770_{-11}$	$-.1252_{-11}$	$-.1770_{-11}$

table 5

Section (b)

h = .25 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	0000	0247	0533	0901	1411	2140	3203	4765	7071
.50	0000	0349	0753	1275	1995	3027	4530	6740	1.000
.75	0000	0247	0533	0901	1411	2140	3203	4765	7071

τ'^{T}	$-.1360_{-3}$	$-.1361_{-3}$	$-.1360_{-3}$
τ''^{T}	$-.1206_{-4}$	$-.1206_{-4}$	$-.1206_{-4}$

table 6

As the solution is precisely symmetric about y=0.5 only half of the following tables are given.

h = .125 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	0000	0133	0288	0487	0762	1157	1732	2578	3827
.250	0000	0249	0532	0900	1409	2138	3201	4764	7071
.375	0000	0322	0695	1176	1841	2794	4183	6275	9239
.500	0000	0349	0752	1273	1993	3024	4527	6738	1.000

τ'^{T}	$-.1363_{-3}$	$-.1364_{-3}$	$-.1364_{-3}$	$-.1364_{-3}$
τ''^{T}	$-.1210_{-4}$	$-.1210_{-4}$	$-.1210_{-4}$	$-.1210_{-4}$

table 7

h = .25 N=13

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	0000	0247	0533	0901	1411	2140	3203	4765	7071
.50	0000	0349	0753	1275	1995	3026	4230	6739	1.000

τ'^{T}	$-.2445_{-10}$	$-.2445_{-10}$
τ''^{T}	$-.1252_{-11}$	$-.1252_{-11}$

table 8

Compare the results of this section with variables separable solution given overleaf.

y	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.00000	0000	0000	0000	0000	0000	0000	0000	0000	0000
.03125	0000	0034	0074	0125	0195	0296	0444	0660	0980
.06250	0000	0068	0147	0248	0389	0590	0883	1314	1951
.09375	0000	0101	0218	0369	0578	0878	1314	1956	2903
.12500	0000	0134	0288	0487	0763	1157	1732	2578	3827
.15625	0000	0164	0355	0600	0939	1425	2134	3176	4714
.18750	0000	01938	0418	0707	1107	1680	2515	3743	5556
.21875	0000	0221	0477	0808	1264	1918	2872	4274	6344
.25000	0000	0247	0532	0900	1409	2138	3201	4764	7071
.28125	0000	0270	0581	0984	1540	2337	3499	5208	7730
.31250	0000	0290	0625	1058	1657	2514	3764	5602	8315
.34375	0000	0308	0663	1123	1757	2667	3992	5942	8819
.37500	0000	0322	0695	1176	1841	2793	4182	6224	9239
.40625	0000	0334	0720	1218	1907	2893	4332	6447	9569
.43750	0000	0342	0738	1249	1954	2966	4440	6608	9808
.46875	0000	0347	0749	1267	1983	3009	4505	6705	9952
.50000	0000	0349	0752	1273	1993	3024	4527	6737	1.0000
				....					

(this table is symmetric about y=0.5000)

Section (c)

h = .25 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	0000	0247	0533	0901	1411	2140	3203	4765	7071
.50	0000	0000	0000	0000	0000	0000	0000	0000	0000
.75	0000	-0247	-0533	-0901	-1411	-2140	-3203	-4765	-7071

τ'^{T}	$-.1360_{-3}$	$-.2373_{-19}$	$.1360_{-3}$
τ''^{T}	$-.1206_{-4}$	$-.2103_{-20}$	$.1206_{-4}$

table 9

These results are anti-symmetric about y=0.5, hence hereafter we display only half of each table.

h = .125 N=7

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	0000	0323	0695	1176	1841	2793	4182	6224	9239
.250	0000	0247	0532	0900	1409	2138	3201	4764	7071
.375	0000	0134	0288	0487	0763	1157	1732	2578	3827
.500	00000	0000	0000	0000	0000	0000	0000	0000	0000

τ'^{T}	$-.3291_{-3}$	$-.1364_{-3}$	$-.5649_{-4}$	$-.2379_{-19}$
τ''^{T}	$-.2922_{-4}$	$-.1210_{-4}$	$-.5014_{-5}$	$-.2111_{-20}$

table 10

h = .25 N=13

	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	0000	0247	0533	0901	1411	2140	3203	4765	7071
.50	0000	0000	0000	0000	0000	0000	0000	0000	0000

τ'^{T}	$-.2445_{-10}$	$-.4264_{-26}$
τ''^{T}	$-.1252_{-11}$	$-.2183_{-27}$

table 11

For the purpose of comparison an exact solution appears in table 12.

y	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	0000	0019	0052	0118	0265	0600	1408	3589	1.094
.250	0000	0028	0073	0166	0366	0804	1704	3870	8488
.375	0000	0019	0051	0116	0253	0540	1115	2177	4150
.500	0000	0000	0000	0000	0000	0000	0000	0000	0000

table 12

Good agreement was also obtained with the following boundary conditions:-

d) $u = y(y-1)$ on BD

e) $u = y(y^2-1)$ "

f) $u = y^2(y^2-1)$ "

g) $u = y$ "

h) $u = y-\frac{1}{2}$ " .

3.5 In order to illustrate the use of the method on non-rectangular regions we considered two further problems. First, Laplace's equation on the region of figure 2 .

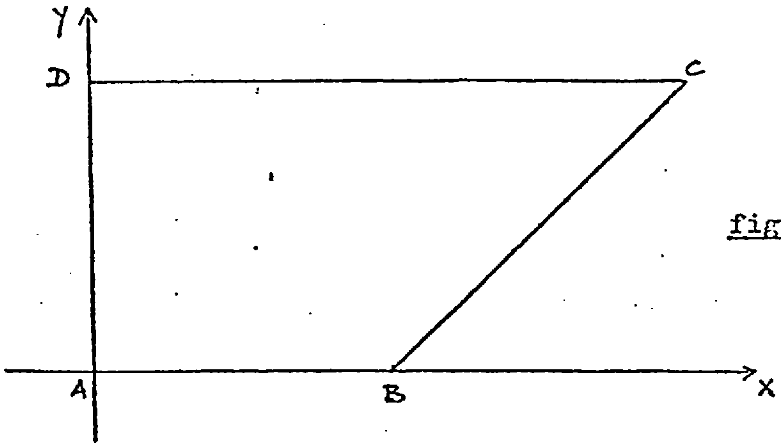

figure 2

The vertices A,B,C,D have coordinates (0,0), (1,0), (2,1), (0,1) respectively. The boundary conditions are u(x,y)=0 on AB, CD, AD and u(x,y)=sin(πy) on BC. The approximate solution was tabulated at the coordinates shown in table 13.

	x - values								
y	1	2	3	4	5	6	7	8	9
875	0	23437	46875	70312	93750	1.1719	1.4063	1.6406	1.8750
750	0	21875	43750	65625	87500	1.0938	1.3125	1.5313	1.7500
625	0	20312	40625	60937	81250	1.0156	1.2188	1.4219	1.6250
500	0	18750	37500	56250	75000	93750	1.1250	1.3125	1.5000
375	0	17187	34375	51562	68750	85937	1.0313	1.2031	1.3750
250	0	15625	31250	46875	62500	78125	93750	1.0938	1.2500
125	0	14062	28125	42187	56250	70312	84375	98437	1.1250

table 13

In the following tables the entry in row y=aaa and column M should be understood to be the approximate value of the solution at the point (x_M, aaa).

h=0.25 N=7

	1	2	3	4	5	6	7	8	9
750	0	0076108	016260	031433	058491	10790	20065	37574	70711
500	0	0091946	022094	043996	082969	15387	28566	53392	1.000
250	0	0060919	015450	031095	058962	10965	20337	37854	70711

τ'^{T}	-.018673	-.0018829	-.018954
τ''^{T}	-.003462	-.0034883	-.003514

table 14

h=0.25 N=8

	1	2	3	4	5	6	7	8	9
750	0	0061210	015484	030835	058177	10826	20139	37633	70711
500	0	0096041	022621	044136	082735	15378	28590	53351	1.000
250	0	0071385	016257	031414	058711	10907	20277	37803	70711

τ'^{T}	-.0034730	-.0034889	-.0035015
τ''^{T}	$-.58370_{-3}$	$-.58617_{-3}$	$-.58849_{-3}$

table 15

h=0.25 N=12

	1	2	3	4	5	6	7	8	9
750	0	006478	015882	031165	058459	10864	20205	37714	70711
500	0	009477	022515	044113	082712	15369	28582	53346	1.000
250	0	006744	015951	031215	058510	10871	20216	37728	70711

τ'^{T}	$-.15782_{-5}$	$-.15788_{-5}$	$-.15793_{-5}$
τ''^{T}	$-.18783_{-6}$	$-.18789_{-6}$	$-.18796_{-6}$

table 16

h=0.25 N=13

	1	2	3	4	5	6	7	8	9
750	0	006723	015932	031193	058478	10866	20207	37718	70711
500	0	009471	022511	044110	082711	15369	28582	53346	1.000
250	0	006676	013907	031190	058493	10869	20213	37725	70711

τ'^{T}	$-.18786_{-6}$	$-.18789_{-6}$	$-.18793_{-6}$
τ''^{T}	$-.20661_{-7}$	$-.20665_{-7}$	$-.20669_{-7}$

table 17

h=.125 N=7

	1	2	3	4	5	6	7	8	9
875	0	0042556	0086131	016511	030619	056665	10630	20095	38268
750	0	0072917	015610	030273	056601	10509	19688	37183	70711
625	0	0087605	019959	039324	074071	13791	25815	48685	92388
500	0	0088069	021229	042423	080371	14997	28053	52819	1.000
375	0	0077442	019423	039161	074468	13917	26016	48908	92388
250	0	0058391	014857	030001	057151	10692	19978	37504	70711
125	0	0032281	0080928	016260	030985	058023	10839	20325	38268

τ'^{T}	$-.1792_{-1}$	$-.1798_{-1}$	$-.1806_{-1}$	$-.1815_{-1}$	$-.1822_{-1}$	$-.1828_{-1}$	$-.1831_{-1}$
τ''^{T}	$-.3327_{-2}$	$-.3336_{-2}$	$-.3350_{-2}$	$-.3365_{-2}$	$-.3379_{-2}$	$-.3391_{-2}$	$-.3399_{-2}$

<u>table 18</u>

h=.125 N= 8

	1	2	3	4	5	6	7	8	9
875	0	0030155	0079274	015993	030421	056975	10680	20143	38268
750	0	0058693	014876	029720	056336	10547	19766	37255	70711
625	0	0081123	019775	039084	073830	13814	25879	48742	92388
500	0	0092093	021733	042554	080153	14988	28071	52834	1.000
375	0	0087950	020297	039487	074231	13875	25983	48875	92388
250	0	0068449	015621	030296	056901	10635	19915	37444	70711
125	0	0037107	0084631	016410	030812	057593	10786	20277	38268

τ'^{T}	$-.3342_{-2}$	$-.3348_{-2}$	$-.3356_{-2}$	$-.3365_{-2}$	$-.3372_{-2}$	$-.3378_{-2}$	$-.3382_{-2}$
τ''^{T}	$-.5622_{-3}$	$-.5631_{-3}$	$-.5644_{-3}$	$-.5657_{-3}$	$-.5671_{-3}$	$-.5682_{-3}$	$-.5689_{-3}$

<u>table 19</u>

h=.125 N=12

	1	2	3	4	5	6	7	8	9
875	0	0034412	0082521	016256	030649	057297	10735	20210	38268
750	0	0063755	015260	030047	056632	10588	19837	37347	70711
625	0	0083608	019961	039275	074010	13837	25922	48801	92388
500	0	0090852	021632	042530	080128	14980	28063	52827	1.000
375	0	0084231	020006	039308	074046	13842	25931	48811	92388
250	0	0064639	015325	030095	056683	10596	19849	37362	70711
125	0	0035038	0082976	016290	030680	057351	10743	20221	38268

τ'^T	$-.1526_{-5}$	$-.1526_{-5}$	$-.1527_{-5}$	$-.1527_{-5}$	$-.1527_{-5}$	$-.1528_{-5}$	$-.1528_{-5}$
τ''^T	$-.1818_{-6}$	$-.1818_{-6}$	$-.1818_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$

table 20

h=.125 N=13

	1	2	3	4	5	6	7	8	9
875	0	0034916	0082859	016276	030658	057313	10737	20213	38268
750	0	0064439	015306	030073	056651	10591	19840	37350	70711
625	0	0084051	019990	039291	074022	13838	25924	48203	92388
500	0	0090808	021628	042528	080127	14980	28062	52827	1.000
375	0	0083753	019974	039290	074033	13841	25928	48809	92388
250	0	0064016	015284	030071	056666	10594	19846	37358	70711
125	0	0034617	0082700	016274	030669	057338	10741	20219	38268

τ'^T	$-.1818_{-6}$	$-.1818_{-6}$	$-.1818_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$	$-.1819_{-6}$
τ''^T	$-.2001_{-7}$	$-.2001_{-7}$	$-.2002_{-7}$	$-.2002_{-7}$	$-.2002_{-7}$	$-.2002_{-7}$	$-.2002_{-7}$

table 21

Laplace's equation was also solved on the domain of figure 3 with boundary conditions:-

(i) u=0 on AB,AD,DC;

(ii) u=sin(πy) on BC.

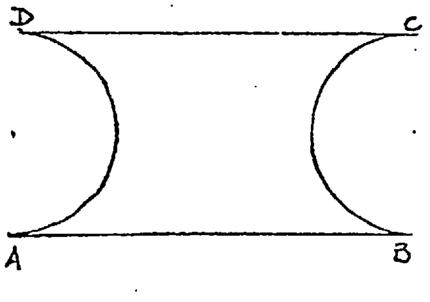

figure 3

The boundary lines AB and DC have equations y=0 and y=1 respectively, while the arcs AD and BC have equations $(y - 0.5)^2 + x^2 = 4$ and $(y - 0.5)^2 + (x - 6)^2 = 4$ respectively. The solution was tabulated at the points of table 22.

	1	2	3	4	5	6	7	8	9
875	1.9645	2.2234	2.4823	2.7411	3.0000	3.2589	3.5177	3.7766	4.0355
750	1.9843	2.2382	2.4922	2.4761	3.0000	3.2539	3.5078	3.7618	4.0157
625	1.9961	2.2471	2.4980	2.7490	3.0000	3.2510	3.5020	3.7529	4.0039
500	2.0000	2.2500	2.5000	2.7500	3.0000	3.2500	3.5000	3.7500	4.0000
375	1.9961	2.2471	2.4980	2.7490	3.0000	3.2510	3.5070	3.7529	4.0039
250	1.9843	2.2382	2.4922	2.7461	3.0000	3.2539	3.5078	3.7618	4.0157
125	1.9645	2.2234	2.4823	2.7411	3.0000	3.2584	3.5177	3.7766	4.0355

table 22

A value found in row y=.aaa and column n is to be taken as the approximate solution at $(x_n, .aaa)$ of figure 3 – for this section only of course.

h=.25 N=7

	1	2	3	4	5	6	7	8	9
.750	0	0021812	0058319	013494	029715	065498	14497	32020	70711
.500	0	0029526	0081938	019035	042046	092758	20522	45320	1.000
.250	0	0021812	0058319	013494	029715	065498	14497	32020	70711

τ'^{T}	$-.73162_{-2}$	$-.73273_{-2}$	$.73162_{-2}$
τ''^{T}	$-.14338_{-2}$	$-.14348_{-2}$	$-.14338_{-2}$

table 23

Because of the symmetry of the solution about y=0.5 we only show half the solution (and the taus) subsequently.

h=.25 N=8

	1	2	3	4	5	6	7	8	9
.750	0	0021861	0058225	013343	029715	065648	14498	32019	70711
.500	0	0031577	0082785	018901	042046	092892	20514	45299	1.000

$\tau,^{T}$	$-.14338_{-2}$	$-.14348_{-2}$
τ''^{T}	$-.25497_{-3}$	$-.25505_{-3}$

table 24

h=.25 N=12

	1	2	3	4	5	6	7	8	9
.750	0	0021937	0058477	013361	029702	065669	14503	32024	70711
.500	0	0031099	0082713	018896	042006	092871	20511	45289	1.000

$\frac{1}{2}\tau'^{T}$	$-.81822_{-6}$	$-.81823_{-6}$
$\frac{1}{2}\tau''^{T}$	$-.10330_{-6}$	$-.10331_{-6}$

table 25

h=.25 N=13

	1	2	3	4	5	6	7	8	9
.750	0	0021985	0058483	013362	029702	065669	14503	32024	70711
.500	0	0031802	0082703	018896	042006	092871	20511	45289	1.000

τ'^{T}	$-.10330_{-6}$	$-.10331_{-6}$
τ''^{T}	$-.12062_{-7}$	$-.12062_{-7}$

table 26

h=.125 N=7

	1	2	3	4	5	6	7	8	9
.875	0	0012157	0031303	0072457	015928	035146	078022	17274	38268
.750	0	0021521	0052555	013341	029441	065040	14428	31943	70711
.625	0	0027178	0074826	017398	038481	085065	18865	41760	92388
.500	0	0029047	0080832	018819	041658	092108	20425	45211	1.000

τ'^{T}	$-.72706_{-2}$	$-.72805_{-2}$	$-.72885_{-2}$	$-.72915_{-2}$
τ''^{T}	$-.14271_{-2}$	$-.14279_{-2}$	$-.14286_{-2}$	$-.14289_{-2}$

table 27

h=.125 N=8

	1	2	3	4	5	6	7	8	9
.875	0	0011430	0030942	0071288	015928	035263	078058	17282	38268
.750	0	0021537	0057459	013192	029441	065188	14429	31942	70711
.625	0	0028595	0075370	017257	038481	085206	18860	41746	92388
.500	0	0031137	0081699	018687	041658	092240	20416	45191	1.000

τ'^{T}	$-.14271_{-2}$	$-.14279_{-2}$	$-.14286_{-2}$	$-.14289_{-2}$
τ''^{T}	$-.25401_{-3}$	$-.25407_{-3}$	$-.25413_{-3}$	$-.25415_{-3}$

table 28

h=.125 N=12

	1	2	3	4	5	6	7	8	9
.875	0	0011714	0031278	0071490	015926	035290	078117	17289	38268
.750	0	0021656	0057708	013210	029428	065208	14434	31947	70711
.625	0	0028309	0075407	017260	038450	085199	18859	41741	92388
.500	0	0030647	0081623	018683	041619	092219	20413	45180	1.000

τ'^{T}	$-.81714_{-6}$	$-.81715_{-6}$	$-.81716_{-6}$	$-.81716_{-6}$
τ''^{T}	$-.10324_{-6}$	$-.10324_{-6}$	$-.10324_{-6}$	$-.10324_{-6}$

table 29

h=.125 N=13

	1	2	3	4	5	6	7	8	9
.875	0	0011728	0031235	0071494	015927	035290	078118	17290	38268
.750	0	0021666	0057713	013210	029429	065208	14434	31947	70711
.625	0	0028303	0075403	017260	038450	085199	18859	41741	92388
.500	0	0030633	0081615	018682	041618	092219	20413	45180	1.000

τ'^{T}	$-.10324_{-6}$	$-.10324_{-6}$	$-.10324_{-6}$	$-.10324_{-6}$
τ''^{T}	$-.12064_{-7}$	$-.12064_{-7}$	$-.12064_{-7}$	$-.12064_{-7}$

table 30

3.6 Comments and notes:

a) The approximate solution to a problem with continuous boundary conditions exactly matches the separation of the variables solution, while the others differ by a small amount. The boundary conditions always fit exactly.

b) The tau values are independent of the h values. It is apparent from the problem (b) that the accuracy depends however on h as well as the order of the perturbation, however even for crude h (namely h=.25) a result correct to 2S is attained with N=7. Here the accuracy is apparently a function of log($\tau^{\frac{1}{2}}$).
Also, the tau value is a function of the shape of the domain. For a particular problem it depends too , naturally, on the order of the perturbation.

c) Small taus alone do not indicate an accurate solution - small taus coupled with a small h do however. The reason for this is clear - increasing the order of the Chebyshev perturbation is equivalent in a full discretization approach to decreasing the x step size. In a more standard approach both step lengths ought to be small for accuracy. Compare, for example, tables 7 and 8 with the separation of variables solution on page 87: on the line y=0.25 the Chebyshev solution with h=0.125, N=7 compares more favourably with the "exact" solution than that with h=0.25, N=13 and yet in the latter case the taus are about 10^{-7} times those in the former.

d) In the final problem considered the approximate solution arrived at by using Chebyshev perturbations of odd and even degree apparently bracket the correct solution.

3.7 Extension of method.

Consider the Poisson equation

$$\nabla^2 u = \emptyset(x,y) \qquad (21)$$

4?

on the domain ABCD of figure 4 and boundary conditions

$$\begin{aligned} u &= f(x,y) \quad \text{on AC} \\ u &= g(x,y) \quad \text{on BD} \qquad (22) \\ u &= 0 \text{ on AB and CD.} \end{aligned}$$

Discretizing as before, or by means of one of the formulae of Collatz [5] (to obtain a different set of equations), we get the set

$$\frac{5}{6} u''_k(x) + \frac{1}{12}\left[u''_{k+1}(x) + u''_{k-1}(x)\right] + \frac{1}{h^2}\left[u_{k+1}(x) - 2u_k(x) + u_{k-1}(x)\right] +$$

$$+ 0(h^4) = \emptyset(x,y_k) \qquad ,k=1,2..,n. \qquad (23)$$

Again, along the boundaries AB and CD

$$u_o(x) = u_{n+1}(x) = u''_o(x) = u''_{n+1}(x) = 0 \qquad (24)$$

and so, ignoring the error term, the equations (23) may be written

$$A\ U'' + \frac{M}{h^2}\ U = \emptyset\emptyset \quad , \qquad (25)$$

in which, A, U, M have the same meaning as before and

$$\emptyset\emptyset = \left[\emptyset\ (x,y_1),\ \emptyset(x,y_2),\dots,\ \emptyset(x,y_n)\right]^T \ .$$

Again, constructing canonical polynomials and perturbing (25) by

$$\tau'\ TT^*_N + \tau''\ TT^*_{N+1} \qquad (26)$$

we have

$$\bar{U}(X) = \tau' \sum_{m=0}^{N} c_m^{(N)}\ Q_m(X) + \tau'' \sum_{m=0}^{N+1} c_m^{(N+1)}\ Q_m(X) + \emptyset\emptyset(Q_m) \qquad (27)$$

as the approximate solution to (25). We have tacitly assumed here that the elements, $\emptyset(x,y_i)$, of $\emptyset\emptyset$ are polynomials in X - or may be closely approximated by polynomials in X. $\emptyset\emptyset(Q_m)$ means that X^m is to be replaced by Q_m. The steps required to evaluate τ' and τ'' using the boundary conditions along AC and AD in (27) are obvious - hence the solution.

3.8 Numerical results : We computed approximate solutions to the Poisson equation $\nabla^2 u = x^2 - 1$, using the technique described above, on three different regions of the type shown in figure 1. In each case the boundary conditions are u=0 on the boundaries AB, AC and CD and u=1 on BD. AB and CD are the lines y=0 and y=1 respectively. The results, in each case, are tabulated for N=12, y=1/4, 1/8 and 1/16 at the tabulated coordinates. We show the results for only one value of N because the computed approximation is that to which the approximations converge and only at the tabulated points so as not to overwhelm with a mass of numerical data.

(a) AC : x=0

BD : x=1.

Coordinates of tabulated approximations :-

y						x				
		1	2	3	4	5	6	7	8	9
1	.250	.000	.125	.250	.375	.500	.625	.750	.875	1.00
2	.500					as above				
3	.750					as above				

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1259	1630	2107	2763	3699	5060	7057	1.0000
2	0000	1563	1923	2385	3019	3923	5237	7163	1.0000
3					symmetric				

All the taus lie in the range 10^{-8} to 10^{-10}.

y = .250 table 31

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1255	1622	2095	2748	3682	5044	7046	1.0000
2	0000	1557	1912	2369	3000	3902	5217	7149	1.0000
3					symmetric				

The taus lie in the range above.

y = .125 table 32

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1254	1620	2092	2744	3679	5041	7044	1.0000
2	0000	1556	1909	2365	2995	3897	5212	7146	1.0000
3					symmetric				

Taus as above.

y = 0.0625 table 33

(b) AC : $x=0$

BD : $x=1+y$.

Coordinates of tabulated approximations :-

y					x				
	1	2	3	4	5	6	7	8	9
1 .25	0000	15625	31250	46875	62500	78125	93750	1.0938	1.2500
2 .50	0000	18750	37500	56250	75000	93750	1.1250	1.3125	1.5000
3 .75	0000	21875	43750	65625	87500	1.0938	1.3125	1.5313	1.7500

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1029	1151	1351	1705	2350	3548	5787	1.0000
2	0000	1340	1460	1655	1998	2622	3778	5938	1.0000
3	0000	1028	1152	1352	1706	2352	3548	5787	1.0000

The taus lie within the range 10^{-4} to 10^{-6}.

$y = 0.25$ table 34

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1022	1136	1327	1668	2300	3486	5730	1.0000
2	0000	1331	1443	1627	1957	2568	3712	5878	1.0000
3	0000	1021	1136	1326	1668	2300	3485	5728	1.0000

The taus are as above.

$y = 0.125$ table 35

y					x				
	1	2	3	4	5	6	7	8	9
1	0000	1020	1133	1321	1660	2289	3472	5716	1.0000
2	0000	1329	1438	1620	1948	2555	3697	5863	1.0000
3	0000	1019	1132	1320	1659	2288	3470	5714	1.0000

The taus are as above.

$y = 0.0625$ table 36

(c) AC : $x = \sqrt{\frac{1}{4} - (y - \frac{1}{2})^2}$

BD : $x = 2 - \sqrt{\frac{1}{4} - (y - \frac{1}{2})^2}$

Coordinates of tabulated approximations :-

y					x				
	1	2	3	4	5	6	7	8	9
1 .25	43301	57476	71651	85825	1.0000	1.1417	1.2835	1.4252	1.5670
2 .50	50000	62500	75000	87500	1.0000	1.1250	1.2500	1.3750	1.5000
3 .75					symmetric				

y					x				
	1	2	3	4	5	6	7	8	9
1	0891	1182	1513	1944	2553	3451	4804	6860	1.0000
2	1189	1478	1803	2223	2814	3682	4987	6972	1.0000
3					symmetric				

All taus between 10^{-4} and 10^{-6}.

h = 0.25 table 37

y					x				
	1	2	3	4	5	6	7	8	9
1	0926	1185	1487	1890	2473	3350	4693	6770	1.0000
2	1235	1487	1780	2170	2734	3581	4877	6882	1.0000
3					symmetric				

Taus as above

h = 0.125 table 38

y					x				
	1	2	3	4	5	6	7	8	9
1	0935	1185	1476	1874	2449	3318	4656	6739	1.0000
2	1246	1488	1772	2154	2709	3548	4841	6851	1.0000
3					symmetric				

Taus as above

h = 0.0625 table 39

3.9 Boundary conditions more complex than those encountered in the last few sections may be handled by discretizing Laplace's equation by means of the simpler central difference approximation. For example, consider the conditions

$$u(x,y) = p(x) \text{ along } AB$$

$$\text{and} \quad u(x,y) = q(x) \text{ along } CD \text{ of figure 4 .}$$

4?

Introducing equally spaced mesh lines $y=y_0$, $y=y_1, \ldots$, $y=y_{n+1}$, as before, and discretizing in the y-direction leads to

$$I \frac{d^2U}{dx^2} + \frac{MU}{h^2} = R(X) , \qquad (28)$$

where, as before, $U(X) = \left[u_1(x), u_2(x), \ldots, u_n(x)\right]^T$;

$M = (m_{ij})$, $m_{ii}=-2$, $m_{i,i+1}=m_{i+1,i}=1$,$(n \times n)$;

$X^m = \left[x^m, x^m, \ldots, x^m\right]^T$, $(n \times 1)$;

I the unit matrix;

$R(X) = \frac{-1}{h^2}\left[p(x), 0, 0, \ldots, 0, 0, q(x)\right]^T$, $(n \times 1)$.

If p(x) and q(x) are polynomials, or if they may be accurately represented by polynomials, we may write

$$p(x) = \sum_{i=0}^{n_p} p_i x^i = \sum_{i=0}^{\bar{n}} p_i x^i ,$$

$$q(x) = \sum_{i=0}^{n_q} q_i x^i = \sum_{i=0}^{\bar{n}} q_i x^i ,$$

where $\bar{n} = \max(n_p, n_q)$. Then $R(X) = \frac{-1}{h^2} \sum_{i=0}^{\bar{n}} C_i X^i$

where $$C_i = \underset{\longleftarrow\, n\text{x}(\bar{n}+1)}{\begin{bmatrix} p_o & p_1 & p_2 \cdots & p_n \\ 0 & 0 & 0 & 0 \\ & \cdots\cdots & \cdots & \\ 0 & 0 & 0 & 0 \\ q_o & q_1 & q_2 \cdots & q_n \end{bmatrix}} \Big[\overbrace{\underset{\sim}{0}, \underset{\sim}{0}, \underset{\sim}{0}, \ldots\ldots, \underset{\sim}{0}, \underset{(\bar{n}+1)\text{xn}}{\underset{\sim}{0}}}^{\longleftarrow i \text{ zero vectors} \longrightarrow}, e_{i+1}, \overbrace{\underset{\sim}{0}, \underset{\sim}{0}, \ldots, \underset{\sim}{0}}^{\substack{\bar{n}-i-1 \\ \leftarrow \text{zero vects} \rightarrow}}\Big]^T ,$$

is an n x n matrix, $\underset{\sim}{0}$ is the (n+1)xn zero vector and e_j is the j-th $(\bar{n}+1)$x1 unit vector.

The Lanczos canonical polynomials associated with (28) are

$$Q_m(X) = h^2 M^{-1} \sum_{i=0}^{[\frac{m}{2}]} (-)^i \frac{m!}{(m-2i)!} h^{2i} M^{-i} X^{m-2i}. \qquad (29)$$

Perturb (28) by $\tau' \, TT^*_N(X) + \tau'' \, TT^*_{N+1}(X)$, τ' , τ'' , TT^*_N and TT^*_{N+1} have been previously defined , to give

$$I \frac{d^2\bar{U}}{dx^2} + \frac{M}{h^2} \bar{U} = R(X) + \tau' \, TT^*_N(X) + \tau'' \, TT^*_{N+1}(X) , \qquad (30)$$

the solution to which is

$$\bar{U}(X) = \frac{-1}{h^2} \sum_{i=0}^{\bar{n}} C_i \, Q_i + \tau' \sum_{m=0}^{N} c_m^{(N)} \, Q_m + \tau'' \sum_{m=0}^{N} c_m^{(N+1)} \, Q_m . \qquad (31)$$

Substituting for $Q_m(X)$ (from (29)) in (30) and rearranging terms the solution becomes

$$\bar{U}(X) = h^2 \tau' M^{-1} \sum_{i=0}^{[\frac{N}{2}]} (-)^i h^{2i} M^{-i} \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} X^{m-2i} +$$

$$+ h^2 \tau'' M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} M^{-i} \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} X^{m-2i} + R(Q), \qquad (32).$$

where $R(Q)$ is to be interpreted as Q_i substituted for X^i in $R(X)$. Requiring that the solution (31) satisfy the as yet unsatisfied boundary conditions,

$$u(x,y) = f(x,y) \text{ on } AC$$

$$\text{and} \qquad u(x,y) = g(x,y) \text{ on } BD,$$

leads to the equations

$$\begin{aligned} \tau' K + \tau'' L &= F(\bar{X}) - R\, Q(\bar{X}) \\ \tau' V + \tau'' W &= F(\bar{X}) - R\, Q(\bar{X}) \end{aligned} , \qquad (33)$$

where $K = [k_1, k_2, \ldots, k_n]^T$

$$= h^2 M^{-1} \sum_{i=0}^{[\frac{N}{2}]} (-)^i h^{2i} M^{-i} \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i}$$

$$L = [l_1, l_2, \ldots, l_n]^T$$

$$= h^2 M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} M^{-i} \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} X^{m-2i} ,$$

$$
\begin{aligned}
V &= [v_1,\ v_2, \ldots,\ v_n]^T \\
&= h^2\ M^{-1} \sum_{i=0}^{[N/2]} (-)^i\ h^{2i}\ M^{-i} \sum_{m=2i}^{N+1} c_m^{(N)} \frac{m!}{(m-2i)!}\ X^{m-2i},
\end{aligned}
$$

$$
\begin{aligned}
W &= [w_1,\ w_2, \ldots,\ w_n]^T \\
&= h^2\ M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i\ h^{2i}\ M^{-i} \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!}\ X^{m-2i}.
\end{aligned}
$$

These equations are equivalent to:-

$$
\begin{bmatrix}
k_1 & & & l_1 & & \\
 & k_2 \ddots & & & l_2 \ddots & \\
 & & k_n & & & l_n \\
v_1 & & & w_1 & & \\
 & v_2 \ddots & & & w_2 \ddots & \\
 & & v_n & & & w_n
\end{bmatrix}
\begin{bmatrix}
\tau_1' \\ \tau_2' \\ \vdots \\ \tau_n' \\ \tau_1'' \\ \tau_2'' \\ \vdots \\ \tau_n''
\end{bmatrix}
=
\begin{bmatrix}
F - R[Q(X)] \\
G - R[Q(X)]
\end{bmatrix} . \qquad (34)
$$

Solve (34) for $\tau_1', \ldots, \tau_n', \tau_1'', \ldots, \tau_n''$ and hence obtain the solution.

The computationally most efficient way of computing R(Q) is had by writing

$$
-R(Q) = \sum_{m=0}^{\bar{n}} C_m\ M^{-1} \sum_{i=0}^{[m/2]} {}_{m,i}\ M^{-i}\ X^{m-2i}\ ,\quad \alpha_{m,i} = \frac{(-)^i\ m!\ h^{2i}}{(m-2i)!}
$$

$$
\begin{aligned}
&= \alpha_{0,0}\ C_0\ M^{-1}\ I \\
&\qquad\qquad\qquad + \alpha_{1,0} C_1\ M^{-1}\ X \\
&+ \alpha_{2,1}\ C_2\ M^{-2}\ I \qquad\qquad + \alpha_{2,0}\ C_2\ M^{-1}\ X^2 \\
&+ \ldots\ldots\ldots \qquad\qquad\qquad\qquad\qquad (35\ )
\end{aligned}
$$

In those cases where $\bar{X} = [0,\ 0,\ \ldots\ ,\ 0]^T$, the vectors K and L reduce to

$$
\begin{aligned}
K &= h^2\ M^{-1} \sum_{i=0}^{[N/2]} (-)^i\ h^{2i}\ c_{2i}^{(N)}\ (2i)!\ M^{-i}\ I\ , \\
L &= h^2\ M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i\ h^{2i}\ c_{2i}^{(N+1)}\ (2i)!\ M^{-i}\ I \qquad .(36\ )
\end{aligned}
$$

and $R[Q(0)] = \bar{R} = [\bar{r}_1,\ \bar{r}_2, \ldots,\ \bar{r}_n]^T$ is obtained from (35).

With $\bar{\bar{X}} = [1,\ 1, \ldots,\ 1]^T$, V and W become

$$V = h^2 M^{-1} \sum_{i=0}^{[N/2]} (-)^i h^{2i} M^{-i} \left\{ \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \right\} \Xi \quad ,$$

$$W = h^2 M^{-1} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} M^{-i} \left\{ \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \right\} \Xi \qquad (37)$$

and $R\left[Q(\Xi)\right] = \bar{\bar{R}} = \left[\bar{\bar{r}}_1, \bar{\bar{r}}_2, \ldots, \bar{\bar{r}}_n\right]^T$ is had from (35)

The solution (34) may be expressed in several ways once the taus have been established, the two most useful perhaps being:-

a) (34) as it stands taken together with (35) – this form minimizes the number of times M^{-i} has to be computed;

b) a rearrangement of (34) into vector polynomial form, viz.

$$U(X) = h^2 \tau' M^{-1} \sum_{i=1}^{N} \left[\sum_{m=0}^{[(N-i)/2]} (-)^m h^{2m} c_{i+2m}^{(N)} \frac{(i+2m)!}{i!} M^{-m} \right] X^i +$$

$$+ h^2 \tau'' M^{-1} \sum_{i=1}^{N+1} \left[\sum_{m=0}^{[(N+1-i)/2]} (-)^m h^{2m} c_{i+2m}^{(N+1)} \frac{(i+2m)!}{i!} M^{-m} \right] X^i -$$

$$- \sum_{i=0}^{\bar{n}} \left[\sum^{[(\bar{n}-i)/2]} (-)^m h^{2m} \frac{(i+2m)!}{i!} C_{2m} M^{-m-1} \right] X^i . \qquad (38)$$

<u>3.10</u> Laplace's equation was solved on the unit square $0 \leq x \leq 1$, $0 \leq y \leq 1$ with the following boundary conditions :-

a) $u=0$ on $x=0$, $y=0$, $y=1$;
   $u=1$ on $x=1$:

b) $u=0$ on $x=0$, $y=0$, $y=1$;
   $u=\sin(\pi y)$ on $x=1$:

c) $u=y(y-1)$ on $x=0$;
   $u=\sin(\pi y)$ on $x=1$;
   $u=x(x-1)$ on $y=0$;
   $u=x^2(x^2-1)$ on $y=1$:

d) $u=\cos(\pi y)$ on $x=0$;
   $u=\sin(\pi y)$ on $x=1$
   $u=-x$ on $y=0$;
   $u=x$ on $y=1$.

The conditions (b) and (c) are continuous, while (a) and (d) are not. Approximate numerical solutions computed for (c) and (d) are given in the next section.

3.11 Results : The obtained approximations are tabulated on the y_i-mesh lines for h=0.25, h=0.125 and h=0.0625 at the x-values x=0.000(0.125)1.000 and for N=7 and N=13. The relevant taus are also tabulated.

Problem (c) :

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.250	-18750	01822	19250	33455	44572	52977	59338	64712	70711
.500	-25000	-13333	-03635	05525	15504	27782	44177	67118	1.000
.750	-18750	-04488	88162	22092	35571	48818	60670	69060	70711

τ_1	$-.661_{-4}$	$-.148_{-3}$	$.101_{-3}$
τ_2	$-.928_{-5}$	$-.764_{-5}$	$.231_{-4}$

h=0.25 N=7

table 40

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	-10938	17352	39210	51867	54744	49371	39621	32311	38268
.250	-18750	-10161	03142	03393	10449	19117	30733	47090	70711
.375	-23438	-12564	-03626	04754	13868	25119	40243	61570	92388
.500	-25000	-13352	-03758	05258	15085	27237	43589	66659	1.000
.625	-23438	-12564	-36256	04754	13868	25119	40243	61570	92388
.750	-18750	-10161	-31415	03393	10449	19117	30733	47090	70711
.875	-10938	13487	37603	60372	78421	87268	83002	64736	38268

τ_1	$-.7_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.3_{-2}$
τ_2	$-.1_{-3}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.3_{-3}$

h=0.125 N=7

table 41

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.0625	-05859	51221	91324	1.077	99354	70972	33587	06340	19509
.1250	-10938	-06031	-02067	01573	05457	10192	16515	25410	38268
.1875	-15234	-08322	-02710	02478	08053	14881	24026	36915	55557
.2500	-18750	-10163	-03161	03348	10377	19023	30631	47009	70711
.3125	-21484	-11573	-03463	04107	12316	22442	36063	55300	83147
.3750	-23438	-12568	-03653	04694	13772	24994	40107	61464	92388
.4375	-24609	-13160	-03755	05065	14675	26569	42600	65261	98078
.5000	-25000	-13356	-03788	05192	14981	27102	43442	66544	1.000
			• • •	• • •	• • •	• • •	•		
.9375	-05859	30940	70037	1.054	1.262	1.214	87118	37197	19509

τ_1	$-.4_{-2}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	..	$-.3_{-1}$
τ_2	$-.8_{-3}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	$-.9_{-5}$	..	$-.3_{-2}$

h=0.0625 N=7

table 42

The dots indicate symmetry.

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.250	-18750	01822	19250	33455	44574	52980	59341	64715	70711
.500	-25000	-13328	-03632	05526	15502	27777	44170	67111	1.000
.750	-18750	-04489	88143	22089	35567	48812	60663	69053	70711

τ_1	$-.1_{-10}$	$-.2_{-10}$	$.2_{-10}$
τ_2	$-.8_{-12}$	$-.7_{-12}$	$.2_{-11}$

h=0.250 N=13

table 43

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	−10938	17353	39216	51878	54763	49399	39658	32348	38268
.250	−18750	−10163	−03143	03392	10450	19120	30737	47093	70711
.375	−23438	−12562	−03624	04755	13867	25117	40240	61568	92388
.500	−25000	−13349	−03755	05259	15084	27234	43584	66654	1.000
				• • • • • • • •					
.875	−10938	13436	37569	69370	78461	87353	83125	64867	38268

τ_1 $-.1_{-9}$	$-.3_{-10}$	$-.3_{-10}$	$-.3_{-10}$ ... $-.6_{-9}$	
τ_2 $-.1_{-10}$	$-.9_{-12}$	$-.9_{-12}$	$-.9_{-12}$ ... $-.3_{-10}$	

h=0.125   N=13

table 44

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.0625	−05859	51216	91332	1.077	99406	71053	33696	06454	19509
.1250	−10938	−06034	−02070	01572	05456	10195	16519	25415	38268
.1875	−15234	−08325	−02713	02478	08054	14884	24031	36920	55557
.2500	−18750	−10165	−03163	03347	10378	19025	30634	47012	70711
.3125	−21484	−11573	−03464	04106	12316	22443	36064	55301	83147
.3750	−23438	−12566	−03652	04694	13772	24993	40105	61462	92388
.4375	−24609	−13157	−03753	05066	14674	26567	42596	65257	98079
.5000	−25000	−13353	−03785	05193	14980	27099	43437	66539	1.000
					• • • • • • • •				
.9375	−05859	30704	69884	1.054	1.264	1.218	87695	37826	19509

τ_1 $-.7_{-9}$	$-.3_{-10}$	$-.3_{-10}$	$-.3_{-10}$	$-.3_{-10}$	$-.3_{-10}$	$-.3_{-10}$ .. $-.5_{-8}$
τ_2 $-.8_{-10}$	$-.9_{-12}$	$-.9_{-12}$	$-.9_{-12}$	$-.9_{-12}$	$-.9_{-12}$	$-.9_{-12}$ .. $-.3_{-9}$

h=0.0625   N=13

table 45

Problem (d)

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	70711	33067	04028	−17003	−29437	−31410	−19505	11751	70711
.50	0	03677	07903	13302	20673	31108	46153	68037	1.000
.75	−70711	−27860	07153	35814	58665	75389	84756	84449	70711

τ_1 $-.3_{-3}$	$-.1_{-3}$	$.9_{-4}$
τ_2 $-.4_{-4}$	$-.1_{-4}$	$.3_{-4}$

h=.25 N=7   table 46

y	x .000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	92388	09986	-59348	-1.147	-1.531	-1.688	-1.527	-90742	38268
.250	70711	50254	37548	30633	28440	30633	37548	50254	70711
.375	38268	29112	24445	23546	26278	33061	44942	63751	92388
.500	0	03534	07615	12872	20113	30455	45493	67542	1.000
.625	-38268	-22581	-10372	00238	10885	23210	39113	61046	92388
.750	-70711	-45252	-26775	-12429	0	12429	26775	45252	70711
.875	-92388	-07276	65180	1.246	1.685	19209	1.875	1.424	38268

τ_1 $-.2_{-2}$	$.5_{-19}$	$-.8_{-4}$	$-.1_{-3}$	$-.2_{-3}$	$-.3_{-3}$	$.2_{-2}$
τ_2 $-.2_{-3}$	$-.2_{-4}$	$-.2_{-4}$	$-.1_{-4}$	$-.7_{-5}$	$-.5_{-20}$	$.2_{-3}$

h=0.125 N=7

table 47

y	x .000	.125	.250	.375	.500	.625	.750	.875	1.00
.0625	98078	-59121	-1.984	-3.141	-3.970	-4.326	-3.994	-2.649	1.951
.1250	92388	63613	44752	32863	26092	23383	24315	29032	38268
.1875	83147	57992	41872	32273	27700	27440	31452	40362	55557
.2500	70711	50142	37382	30443	28244	30443	37382	50142	70711
.3125	55557	40363	31454	27442	27703	32277	41877	57997	83147
.3750	38268	29032	24316	23386	26097	32871	44763	63623	92388
.4375	19509	16584	16242	18430	23487	32201	45928	66805	98078
.5000	0	03498	07544	12765	19974	30293	45328	67418	1.000
.5625	-19509	-09721	-01444	06610	15694	27220	42984	65440	98078
.6250	-38268	-22566	-10375	00202	10810	23100	38988	60945	92388
.6875	-55557	-34542	-18906	-06214	05510	18093	33492	54107	83147
.7500	-70711	-45189	-26710	-12390	0	12390	26710	45189	70711
.8125	-83147	-54099	-33486	-18089	-05510	06211	18900	34535	55557
.8750	-92388	-60930	-38975	-23093	-01081	-00203	10366	22555	38268
.9375	-98078	60489	2.013	3.191	4.047	4.444	4.171	2.911	1.951

τ_1	$-.9_{-2}$	$.2_{-3}$	$.7_{-4}$	$.5_{-19}$	$-.4_{-4}$	$-.8_{-4}$	$-.1_{-3}$	$-.1_{-3}$
	$-.2_{-3}$	$-.2_{-3}$	$-.2_{-3}$	$-.3_{-3}$	$-.3_{-3}$	$-.5_{-3}$	$.8_{-2}$	
τ_2	$-.9_{-3}$	$-.4_{-4}$	$-.3_{-4}$	$-.2_{-4}$	$-.2_{-4}$	$-.2_{-4}$	$-.1_{-4}$	$-.1_{-4}$
	$-.1_{-4}$	$-.7_{-5}$	$-.4_{-5}$	$-.7_{-20}$	$.6_{-5}$	$.2_{-4}$	$.9_{-3}$	

h=0.0625 N=7

table 48

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.25	70711	33066	04029	-16999	-29428	-31396	-19488	11768	70711
.50	0	03680	07905	13301	20670	31103	46146	68031	1.000
.75	-70711	27862	07150	35810	58660	75382	84748	84442	70711

τ_1	$-.5_{-10}$	$-.2_{-10}$	$.2_{-10}$
τ_2	$-.4_{-11}$	$-.9_{-12}$	$.3_{-11}$

h=0.25 N=13

table 49

y					x				
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.125	92388	09964	-59360	-1.147	-1.531	-1.687	-15263	-90663	38268
.250	70711	50256	37551	30635	28442	30635	37551	50256	70711
.375	38268	29112	24444	23546	26277	33060	44940	63750	92388
.500	0	03536	07617	12872	20112	30452	45488	67538	1.000
.625	-38268	-22579	-10371	00238	10884	23208	39111	61044	92388
.750	-70711	-45256	-26779	-12431	0	12431	26779	45256	70711
.875	-92388	-07258	65190	12455	1.684	1.920	1.874	1.424	38268

τ_1	$-.3_{-13}$	$.4_{-25}$	$-.1_{-10}$	$-.2_{-10}$	$-.3_{-10}$	$-.5_{-10}$	$-.3_{-9}$
τ_2	$-.2_{-10}$	$-.2_{-11}$	$-.2_{-11}$	$-.1_{-11}$	$-.7_{-12}$	0	$.2_{-10}$

h=0.125 N=13

table 50

y	x								
	.000	.125	.250	.375	.500	.625	.750	.875	1.00
.0625	98078	-59195	-1.984	-3.141	-3.969	-43251	-39923	-26462	19509
.1250	92383	63622	44761	32870	26096	23385	24316	29032	38268
.1875	83147	57997	41877	32278	27703	27442	31454	40363	55557
.2500	70711	50144	37384	30445	28246	30445	37384	50144	70711
.3125	55557	40363	31454	27442	27703	32278	41877	57997	83147
.3750	38268	29032	24316	23385	26096	32870	44761	63622	92388
.4375	19509	16585	16243	18430	23486	32199	45925	66801	98078
.5000	0	03500	07545	12765	19973	30290	45324	67414	1.000
.5625	-19509	-09719	-01442	06611	15693	27218	42981	65436	98078
.6250	-38268	-22564	-10374	00202	10809	23099	38986	60943	92383
.6875	-55557	-34543	-18907	-06214	05511	18093	33493	54108	83147
.7500	-70711	-45194	-26713	-01239	0	01239	26713	45194	70711
.8125	-83147	-54108	-33493	-18093	-05510	06214	18907	34543	55557
.8750	-92388	-60943	-38986	-23099	-01081	-00202	10374	22564	38268
.9375	-98078	60561	2.014	3.191	4.047	4.443	4.169	2.909	19509

τ_1 $-.2_{-8}$ $.3_{-10}$ $.1_{-10}$ $.4_{-25}$ $-.8_{-11}$ $-.1_{-10}$ $-.2_{-10}$ $-.2_{-10}$
$-.3_{-10}$ $-.3_{-10}$ $-.4_{-10}$ $-.5_{-10}$ $-.6_{-10}$ $-.8_{-10}$ $-.2_{-8}$

τ_2 $-.1_{-9}$ $-.4_{-11}$ $-.3_{-11}$ $-.2_{-11}$ $-.2_{-11}$ $-.2_{-11}$ $-.1_{-11}$ $-.1_{-11}$
$-.1_{-11}$ $-.7_{-12}$ $-.4_{-12}$ $-.1_{-27}$ $.6_{-12}$ $.2_{-11}$ $.9_{-10}$

h=0.0625 N=13

table 51

The computed approximate solutions to problems (a) and (b) were also satisfactory.

CHAPTER 4 : EXTENSIONS OF METHOD.

4.1 Introduction : In this chapter we show how to solve eigenvalue problems and also more general elliptic equations using the techniques of the previous chapter.

4.2 Eigenvalue problem : The eigenvalue problem $\nabla^2 u - \lambda u = 0$, with $u=0$ on the boundary , will be solved here on the region of figure 3.1. Using the notation introduced there, this equation may be discretized to give

$$A\,U'' + \frac{M}{h^2}\,U - \lambda\,U = 0 \qquad (1)$$

together with the boundary conditions

$$U(\bar{X}) = 0$$
$$U(\bar{\bar{X}}) = 0.$$

The matrices A and M and vectors U and U" were defined previously. The canonical polynomials are easily seen (by induction) to be

$$Q_m = h^2\, S^{-1} \sum_{i=0}^{[\frac{m}{2}]} (-)^i \frac{m!\; h^{2i}}{(m-2i)!} (A\, S^{-1})^i\, X^{m-2i} \qquad (2)$$

where $S = \left[M - \lambda h^2\, I\right]$. (3)

Perturbing (1) suitably (as before) we now have to solve

$$A\,U'' + \frac{M}{h^2}\,U - \lambda\,U = \tau'\; TT^*_N(X) + \tau''\; TT^*_{N+1}(X) \quad . \qquad (4)$$

The solution to which is

$$U(X) = \tau' \sum_{m=0}^{N} c_m^{(N)}\; Q_m(X) + \tau'' \sum_{m=0}^{N} c_m^{(N+1)}\; Q_m(X) \;,$$

where, as before, the $c_m^{(N)}$ and $c_m^{(N+1)}$ are the coefficients of the N-th and (N+1)-th Chebyshev polynomials of the first kind respectively.

$$\text{So } U(X) = h^2 \tau' \sum_{m=0}^{N} c_m^{(N)}\; S^{-1} \sum_{i=0}^{[\frac{m}{2}]} (-)^i \frac{m!}{(m-2i)!}\; h^{2i}\; (A\, S^{-1})^i\; X^{m-2i} +$$

$$+ h^2 \tau'' \sum_{m=0}^{N+1} c_m^{(N+1)}\; S^{-1} \sum_{i=0}^{[\frac{m}{2}]} (-)^i \frac{m!}{(m-2i)!}\; h^{2i}\; (A\, S^{-1})^i\; X^{m-2i} \;.$$

Rearranging this

$$U(X) = h^2 \tau' \, S^{-1} \sum_{i=0}^{[\frac{N}{2}]} (-)^i \, h^{2i} \, (A\,S^{-1})^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \, X^{m-2i} \; +$$

$$+ \, h^2 \, \tau'' \, S^{-1} \sum_{i=0}^{[N+1/2]} (-)^i \, h^{2i} \, (A\,S^{-1})^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \, X^{m-2i} . \qquad (5\,)$$

Denote S^{-1} by P. The n x n matrix $P = (p_{ij})$ may be constructed by the following algorithm:-

1) Let $d_o=0$, $d_1=1$ and construct

$$d_r(\lambda) = -(2 + \lambda h^2)\, d_{r-1}(\lambda) - d_{r-2}(\lambda) \quad \text{for } r=2(1)n \,.$$

2) $\bar{p}_{1j} = (-)^{j+1}\, d_{j-1}$, $j=1(1)n$.

3) Define $\bar{p}_{oj} = 0$, $1 \leq j \leq n$ and construct

$$\bar{p}_{ij} = \delta_{i-1,j}\, d_n + (2 + \lambda h^2)\, \bar{p}_{i-1,j} - \bar{p}_{i-2,j} \,,$$

$$i=2(1)n, \; j=1(1)n.$$

4) $P = (p_{ij}) = \frac{1}{d_n} \; (\bar{p}_{ij}) = \frac{1}{d_n} \bar{P}$.

The amount of computation is minimized if use is made of the fact that P is symmetric about both diagonals.

In terms of P then , the solution is

$$U(X) = h^2 \tau' \, P \sum_{i=0}^{[\frac{N}{2}]} (-)^i \, h^{2i} \, (AP)^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \, X^{m-2i} \; +$$

$$+ \, h^2 \, \tau'' \, P \sum_{i=0}^{[(N+1)/2]} (-)^i \, h^{2i} \, (AP)^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \, X^{m-2i} \; . \qquad (6\,)$$

The, as yet unsatisfied, boundary conditions require that

$$\tau' \, P \sum_{i=0}^{[\frac{N}{2}]} (-)^i \, h^{2i} \, (AP)^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \, \bar{X}^{m-2i} \; +$$

$$\tau'' \, P \sum_{i=0}^{[(N+1)/2]} (-)^i \, h^{2i} \, (AP)^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \, \bar{X}^{m-2i} = 0 \,,$$

and a similar condition at $\bar{\bar{X}}$. The λ terms in the denominator of the boundary condition may be cancelled by multiplying by

$d_n(\lambda)^{[(N+1)/2]+1}$ to give

$$\tau' \bar{P} \sum_{i=0}^{[\frac{N}{2}]} (-)^i h^{2i} d_n^{[(N+1)/2]-i} (A\bar{P})^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i} +$$

$$+\tau'' \bar{P} \sum_{i=0}^{[(N+1)/2]} (-)^i h^{2i} d_n^{[(n+2)/2]-i} (A\bar{P})^i \sum_{m=2i}^{N+1} c_m^{(N+1)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i} = 0$$

and a similar condition at $\bar{\bar{X}}$. These two boundary conditions are of the form

$$\tau' K(\lambda) + \tau'' L(\lambda) = 0$$
$$\tau' R(\lambda) + \tau'' V(\lambda) = 0,$$

where $K(\lambda) = [k_1,\ldots,k_n]^T$

$$= \bar{P} \sum_{i=0}^{[\frac{N}{2}]} (-)^i h^{2i} d_n^{[(N+1)/2]-i} (A\bar{P})^i \sum_{m=2i}^{N} c_m^{(N)} \frac{m!}{(m-2i)!} \bar{X}^{m-2i},$$

and similarly for $L(\lambda) = [l_1,\ldots,l_n]^T$,

$R(\lambda) = [r_1,\ldots,r_n]^T$,

$V(\lambda) = [v_1,\ldots,v_n]^T$.

Equivalently then

$$\begin{bmatrix} k_1 & & & l_1 & & \\ & k_2 & & & l_2 & \\ & & \ddots\; k_n & & & \ddots\; l_n \\ r_1 & & & v_1 & & \\ & r_2 & & & v_2 & \\ & & \ddots\; r_n & & & \ddots\; v_n \end{bmatrix} \begin{bmatrix} \tau'_1 \\ \tau'_2 \\ \vdots \\ \tau'_n \\ \tau''_1 \\ \tau''_2 \\ \vdots \\ \tau''_n \end{bmatrix} = 0 . \qquad (7)$$

Equating the determinant of the coefficient matrix to zero, the eigenvalues are found by isolating the roots of this determinantal equation. Having obtained the eigenvalues, the taus may be obtained from (7) and the eigenfunctions from (6).

Because of the inefficiency of this method compared with other techniques for solving this eigenvalue problem we did not perform any numerical computations.

?

4.3 The eigenvalue problem of the above section may also be solved using a central difference approach rather than the more elaborate approximation employed there. In fact the analysis and technique remain largely unaltered, the one major difference is that the matrix A has to be replaced by the unit matrix.

4.4 General elliptic boundary value problem : This problem, viz.

$$L[u] = a\,u_{xx} + c\,u_{yy} + d\,u_x + e\,u_y + f\,u = g \qquad (8a)$$

on a region such as that of figure 3.1 subject to the conditions

$$\left.\begin{aligned} u(x,y) + \alpha\, u_y(x,y) &= u_{ab} \quad \text{on AB,}\\ u(x,y) + \beta\, u_y(x,y) &= u_{cd} \quad \text{on CD,}\\ u(x,y) + \gamma\, u_x(x,y) &= u_{bc} \quad \text{on BC,}\\ u(x,y) + \delta\, u_x(x,y) &= u_{cd} \quad \text{on CD } - \end{aligned}\right\} \qquad (8b)$$

the functions a, b, c, d, e, f, g being polynomial-type functions in both x and y – may be solved by the method of the previous sections, subject to certain conditions being satisfied by c, e and f. Introducing, as before, n+1 equally spaced mesh lines (a distance h apart) in the y-direction and discretizing the differential operator of (8) by the usual central difference operator yields

$$\left[\frac{c_k}{h^2} + \frac{e_k}{2h}\right]u_{k+1} + \left[\frac{-2c_k}{h^2} + f_k\right]u_k + \left[\frac{c_k}{h^2} - \frac{e_k}{2h}\right]u_{k-1} +$$

$$+d_k\,u_k' + a_k\,u_k'' = g_k + O(h^2)\ ,\ k=0,1,\ldots,n+1\ .$$

The notation a_k, c_k,..., u_k indicates that these functions are to be evaluated at (x,y_k) . The dashes mean differentiation with respect to x. Introducing imaginary lines y_{-1} and y_{n+2}, discretizing the first two boundary conditions of (8b) and using the previously defined U, U', U" we now have

$$A\,U'' + B\,U' + C\,U = D\ ;$$

or $$\left[A\frac{d^2}{dx^2} + B\frac{d}{dx} + C\right]U = D; \qquad (9)$$

where $A = \text{diag}(a_o, a_1, \ldots, a_n, a_{n+1})$;

$B = \text{diag}(d_o, d_1, \ldots, d_n, d_{n+1})$;

$C = (c_{ij})$,

and $c_{11} = \frac{2c_o}{h}\left(\frac{1}{\alpha} - \frac{1}{h}\right) + f_o - \frac{e_o}{\alpha}$, $\quad c_{ii} = \frac{-2c_{i-1}}{h^2} + \frac{e_{i-1}}{2h} \quad 2 \leq i \leq n+1$,

$$c_{n+2,n+2} = \frac{-2c_{n+1}}{h}\left(\frac{1}{\beta} + \frac{1}{h}\right) + f_{n+1} - \frac{e_{n+1}}{\beta} ,$$

$$c_{i+1,i} = \frac{c_i}{h^2} - \frac{e_i}{2h} \quad i=1,\ldots,n \qquad c_{n+2,n+1} = \frac{2c_{n+1}}{h^2}$$

$$c_{i,i+1} = \frac{c_{i-1}}{h^2} + \frac{e_{i-1}}{2h} \quad i=2,\ldots,n+1 \quad , c_{12} = \frac{2c_o}{h^2} \quad ;$$

$$D = \begin{bmatrix} g_o + \frac{2}{}\left(\frac{c_o}{h} - \frac{e_o}{2}\right) \\ g_1 \\ \vdots \\ g_n \\ g_{n+1} - \frac{2}{}\left(\frac{c_{n+1}}{h} + \frac{e_{n+1}}{2}\right) \end{bmatrix} ; \quad U = \begin{bmatrix} u_o \\ u_1 \\ \cdot \\ \cdot \\ \cdot \\ u_n \\ u_{n+1} \end{bmatrix} .$$

Define the matrix differential operator DD by

$$DD = A\frac{d^2}{dx^2} + B\frac{d}{dx} + C ,$$

which has as its domain of definition the set of n-dimensional vectors with twice differentiable elements. Following Ortiz [24] the Lanczos canonical polynomials

$$Q_m = C^{-1}\left[X^m - m\,B\,Q_{m-1} - m(m-1)\,A\,Q_{m-2}\right], m=0,1,2,\ldots.$$

are easily obtained. This recurrence relationship is only valid if C is non-singular - an ill-conditioned C could result in the computed approximate solution being inaccurate. An explicit relationship for

Q_m, namely

$$Q_m(X) = C^{-1} \sum_{i=0}^{m} \gamma_{m,i}\, G_{m,i}(S,T)\, X^i \tag{10}$$

where $\gamma_{m,i} = \frac{m!}{(m-i)!}$, $S = A\,C^{-1}$, $T = B\,C^{-1}$ and $G_{m,i}(S,T)$ is a matrix polynomial in S and T, is obtainable from the recurrence relationship.

Now write each element of $D = (\delta_i)$ as a polynomial, where

$$\delta_i = \sum_{k=0}^{n_i} p_{ik}\, x^k = \sum_{k=0}^{\bar{n}} p_{ik}\, x^k$$

and $n = \max_{1 \leq i \leq n+2} (n_i)$, $p_{ik} = 0$ if $k > n_i$.

Then $\Pi = (p_{ik})$ is an (n+2)x(n+1) matrix. As before, define

$$\Pi_i = \Pi.[\underset{\sim}{0}, \underset{\sim}{0}, \ldots, \underset{\sim}{0}, e_{i+1}, \underset{\sim}{0}, \ldots, \underset{\sim}{0}] \text{ , then}$$

$D = \sum_{k=0}^{\bar{n}} \Pi_k\, X^k$ is a vector polynomial of degree n.

The solution to the differential equation (9) perturbed by

$$\tau'\, TT^*_N(X) + \tau''\, TT^*_{N+1}(X)$$

is then

$$\begin{aligned} U(X) &= \sum_{i=0}^{\bar{n}} \Pi_i\, Q_i + \tau' \sum_{m=0}^{N} c_m^{(N)}\, Q_m + \tau'' \sum_{m=0}^{N+1} c_m^{(N+1)}\, Q_m \\ &= C^{-1} \sum_{i=0}^{\bar{n}} \left\{ \sum_{k=i}^{\bar{n}} \gamma_{k,i}\, \Pi_k\, G_{k,i}(S,T) \right\} X^i + \tau'\, C^{-1} \sum_{i=0}^{N} \left\{ \sum_{k=i}^{N} c_k^{(N)} \gamma_{k,i} G(S,T) \right\} X^i \\ &+ \tau''\, C^{-1} \sum_{i=0}^{N+1} \left\{ \sum_{k=i}^{N+1} c_k^{(N+1)} \gamma_{k,i}\, G_{k,i}(S,T) \right\} X^i . \end{aligned}$$

A simple application of the two remaining boundary conditions of (8) leads to an equation of the form

$$\begin{bmatrix} k_o & & & l_o & & \\ & \ddots & & & \ddots & \\ & & k_{n+1} & & & l_{n+1} \\ p_o & & & q_o & & \\ & \ddots & & & \ddots & \\ & & p_{n+1} & & & q_{n+1} \end{bmatrix} \begin{bmatrix} \tau'_o \\ \vdots \\ \tau'_{n+1} \\ \tau''_o \\ \vdots \\ \tau''_{n+1} \end{bmatrix} = \begin{bmatrix} r_o \\ \vdots \\ r_{n+1} \\ s_o \\ \vdots \\ s_{n+1} \end{bmatrix}$$

again, which is easily solved for the taus, and the solution follows.

Once more, we do not give any numerical results.

CHAPTER 5 : AN ERROR ANALYSIS.

5.1 Introduction : We give here an error analysis for the central difference semi-discretization of equation (3.28). That was the case where more complex boundary conditions could be handled.

5.2 Analysis : The error $E(X) = \overline{U}(X) - U(X)$ incurred by perturbing equation (3.28) into the form (3.30) satisfies

$$\frac{d^2E}{dx^2} + \frac{M}{h^2} E = -\tau' \, TT^*_N(X) - \tau'' \, TT^*_{N+1}(X) \qquad (1)$$

with $E(0) = E(1) = 0$.

Note that the additional $0(h^2)$ error has been ignored in this equation. An approximate solution to (1) may be obtained via a Picard type procedure - i.e. by solving (1) recursively in the form

$$\frac{d^2E_{r+1}}{dx^2} = -\frac{M}{h^2} E_r - \tau' \, TT^*_N - \tau'' \, TT^*_{N+1} \, . \qquad (2)$$

Starting from $E_o=0$ and using the well-known identity

$$\iint T^*_k \, dx \, dx = \frac{1}{16}\left[\frac{1}{(k+2)(k+1)} T^*_{k+2} - \frac{1}{k^2-1} T^*_k + \frac{1}{(k-2)(k-1)} T^*_{k-2}\right] \; ; \qquad (3)$$

we have

$$E_1 = \frac{-1}{16}\tau' \left\{\frac{1}{(N+2)(N+1)} TT^*_{N+2} - \frac{2}{N^2-1} TT^*_N + \frac{1}{(N-2)(N-1)} TT^*_{N-2}\right\} -$$

$$\frac{-1}{16}\tau'' \left\{\frac{1}{(N+3)(N+2)} TT^*_{N+3} - \frac{2}{N(N+2)} TT^*_{N+1} + \frac{1}{(N-1)N} TT^*_{N-1}\right\} +$$

$$+ A_o^{(1)} \mathbb{I} + A_1^{(1)} X \qquad (4a)$$

$$= -\tau' \, t_1(X) - \tau'' \, t_2(X) + A_o^{(1)} \mathbb{I} + A_1^{(1)} X \text{ say.} \qquad (4b)$$

$A_o^{(1)}$ and $A_1^{(1)}$ are constant diagonal matrices which are easily evaluated from the boundary conditions as follows:-

$$A_o^{(1)} \mathbb{I} = \tau' \, t_1(0) + \tau'' \, t_2(0)$$

and so

$$A_o^{(1)} = \frac{\frac{3}{4}(-)^N}{(N+2)(N+1)(N-1)(N-2)} \tau' + \frac{\frac{3}{4}(-)^{N+1}}{(N+3)(N+2)(N)(N-1)} \tau'' \, . \qquad (5)$$

Also $A_1^{(1)}\,\Xi = \tau'\,t_1(\Xi) + \tau''\,t_2(\Xi) - A_0^{(1)}\,\Xi$ and hence

$$A_1^{(1)} = \frac{\frac{3}{4}\,(1-(-)^N)}{(N+2)(N+1)(N-1)(N-2)}\,\tau' + \frac{\frac{3}{2}\,(1+(-)^N)}{(N+3)(N+2)(N)(N-1)}\,\tau'' \,. \tag{6}$$

The next iteration gives

$$E_2(X) = \frac{1}{196h^2}\,M\,\tau'\Big\{\frac{1}{(N+1)(N+2)(N+3)(N+4)}\,TT^*_{N+4}(X) -$$

$$- \frac{4}{(N-1)(N+1)(N+2)(N+3)}\,TT^*_{N+2}(X) + \frac{6}{(N-2)(N-1)(N+1)(N+2)}\,TT^*_N(X) -$$

$$- \frac{4}{(N-3)(N-2)(N-1)(N+1)}\,TT^*_{N-2}(X) + \frac{1}{(N-4)(N-3)(N-2)(N-1)}\,TT^*_{N-4}(X)\Big\} +$$

$$+ \frac{1}{196h^2}\,M\,\tau''\Big\{\frac{1}{(N+2)(N+3)(N+4)(N+5)}\,TT^*_{N+5}(X) -$$

$$- \frac{4}{N(N+2)(N+3)(N+4)}\,TT^*_{N+3}(X) + \frac{6}{(N-1)(N)(N+2)(N+3)}\,TT^*_{N+1}(X) -$$

$$- \frac{4}{(N-2)(N-1)(N)(N+2)}\,TT^*_{N-1}(X) + \frac{1}{(N-3)(N-2)(N-1)N}\,TT^*_{N-3}(X)\Big\} -$$

$$- \tau'\,t_1(X) - \tau''\,t_2(X) - \frac{M}{2h^2}\,A_0^{(1)}\,X^2 - \frac{M}{6h^2}\,A_1^{(1)}\,X^3 + A_0^{(2)}\,\Xi + A_1^{(2)}\,X$$

$$= M\,\tau'\,s_1(X) + M\,\tau''\,s_2(X) - \tau'\,t_1(X) - \tau''\,t_2(X) +$$

$$+ A_0^{(2)}\,\Xi + A_1^{(2)}\,X - \frac{M}{2h^2}A_0^{(1)}\,X^2 - \frac{M}{6h^3}\,A_1^{(1)}\,X^3 \quad \text{say.} \tag{7}$$

E_2 is required to satisfy the boundary conditions $E_2(0)=E_2(\Xi)=0$, from which it is easily established that

$$A_0^{(2)} = A_0^{(1)} + O(N^{-8}\,h^{-2})\,I$$

$$\text{and} \quad A_1^{(2)} = A_1^{(1)} + \frac{M}{6h^2}\,A_1^{(1)} + \frac{M}{2h^2}\,A_0^{(1)} + O(N^{-8}\,h^{-2})I \tag{8}$$

Almost equivalently then

$$E_2(X) = M\tau' s_1(X) + M\tau'' s_2(X) - \tau' t_1(X) - \tau'' t_2(X) + \\ + \sum_{i=0}^{3} A_i^{(2)} X^i \qquad (9a)$$

where now, $A_0^{(2)} = A_0^{(1)}$, $A_1^{(2)} = A_1^{(1)} + \frac{M}{6h^2} A_1^{(1)} + \frac{M}{2h^2} A_0^{(1)}$,

$$A_2^{(2)} = -\frac{M}{2h^2} A_0^{(1)} \ , \quad A_3^{(2)} = -\frac{M}{6h^2} A_1^{(1)} \ . \qquad (9b)$$

From the above it may be conjectured that

$$E_p(X) \doteq M\tau' s_1(X) + M\tau'' s_2(X) - \tau' t_1(X) - \tau'' t_2(X) + \\ + \sum_{i=0}^{2p-1} A_i^{(p)} X^i \qquad (10)$$

By (9) and (2)

$$E_{p+1}(X) = O(N^{-6} h^{-2})I + M\tau' s_1(X) + M\tau'' s_2(X) - \frac{M}{h^2} \sum_{i=2}^{2p+1} \frac{A_{i-2}^{(p)}}{i(i-1)} X^i + \\ + A_0^{(p+1)} I + A_1^{(p+1)} X - \tau' t_1(X) - \tau'' t_2(X).$$

Using the boundary conditions $E_{p+1}(0) = E_{p+1}(I) = 0$ it may be seen that

$$A_0^{(p+1)} = A_0^{(1)} + O(N^{-6} h^{-2})$$

$$A_1^{(p+1)} = A_1^{(1)} + \frac{M}{h^2} \sum_{i=2}^{2p+1} \frac{A_{i-2}^{(p)}}{i(i-1)} + O(N^{-6} h^{-2}) \ . \qquad (11)$$

Approximately, therefore

$$E_{p+1}(X) = M\tau' s_1(X) + M\tau'' s_2(X) - \tau' t_1(X) - \tau'' t_2(X) + \\ + \sum^{2p+1} A_i^{(p+1)} X^i \qquad (12)$$

where, in the definition (11) of $A_0^{(p+1)}$ and $A_1^{(p+1)}$ the error terms are now discarded, and

$$A_i^{(p+1)} = -\frac{M}{i(i-1)h^2} A_{i-2}^{(p)} \qquad , \ i=2,3,\dots,2p+1 \ .$$

(12) outlines an obvious algorithm for computing the E_p's.

The above analysis clearly shows the dangers of coupling a small h with a small M - in fact in this case the approximations made in the error analysis may not be acceptable.

Thus, for i=2,3,..,2p-1, $A_i^{(p)} = \frac{(-)^{[i/2]}}{i!h^{2[i/2]}} M^{[i/2]} A_{i-2[i/2]}^{(p-[i/2])}$,

from which, two cases arise:-

(i) i even (=2j say) then

$$A_{2j}^{(p)} = \frac{(-)^j}{(2j)!h^{2j}} M^j A_o^{(p-j)}$$

$$= \frac{(-)^j}{(2j)!h^{2j}} M^j A_o^{(1)} \text{ approximately - by 11.} \qquad (13)$$

(ii) i odd (=2j+1 say)

$$A_{2j+1}^{(p)} = \frac{(-)^j}{(2j+1)!h^{2j}} M^j A_1^{(p-j)}$$

$$= \frac{(-)^j}{(2j+1)!h^{2j}} M^j F(M, A_1^{(1)}, A_o^{(1)}) \text{ approximately.} \qquad (14)$$

Both $A_{2j}^{(p)}$ and $A_{2j+1}^{(p)}$ approach the null matrix as $j \to \infty$.

CHAPTER 6 : AN ERROR ANALYSIS OF THE METHOD OF WRAGG.

6.1 Introduction. In the paper [37] by Wragg he numerically solves the following particular Stefan problem,

$$u_t = u_{xx} \ , \quad 0 < x < x(t), \ t > 0 \ , \qquad (1)$$

$$u_x(0,t) = -1 \ , \quad t > 0 \quad , \qquad (2)$$

$$u(x(t),t) = 0 \ , \ x = x(t), \quad t > 0 \ , \qquad (3)$$

$$\frac{d\,x(t)}{dx} = - u_x(x(t),t) \ , \quad t > 0 \ , \qquad (4)$$

$$x(0) = 0 \ , \qquad (5)$$

using an extension of the Lanczos-tau algorithm.

$U_o(x)$, $U_1(x)$ are assumed to be approximations to $u(x,t_o)$, $u(x,t_1)$ and the points (x_o,t_o),(x_1,t_1) to lie on the moving boundary. If x_o and $U_o(x)$ are known then finite-difference representations of (1), (2), (3) and (4) yield the equations :-

$$\left.\begin{aligned} &\frac{d^2U_1(x)}{dx^2} - \frac{U_1(x)}{\Delta t} + \frac{U_o(x)}{\Delta t} = 0 \\ &\left[\frac{d}{dx} U_1(x)\right]_{x=o} = -1 \\ &\left[U_1(x)\right]_{x=x_1} = 0 \\ &\frac{x_1 - x_o}{\Delta t} = \left[\frac{dU_o(x)}{dx}\right]_{x=x_o} . \end{aligned}\right\} \qquad (6)$$

These equations are solved by Wragg using the Lanczos-tau method (Lanczos [19]) after the first equation of (6) has been perturbed by $(\tau' + \tau'' \frac{x}{x_1}) \, T^*_n(\frac{x}{x_1})$, where $T^*_n(x) = \sum_{m=o}^{n} c_m^{(n)} x^m$ is the n-th shifted Chebyshev polynomial of the first kind. Wragg compares the numerical results obtained in this way with those obtained by solving (1) - (5) using the Douglas-Gallie method. The following table has been extracted from his paper. In both cases $\Delta t = 0.1$.

	Wragg	D-G
x=0.4	0.4662	0.4659
x=0.8	1.0415	1.0408
x=1.2	1.7061	1.7051
x=1.6	2.4500	2.4488
x=2.0	3.2668	3.2654
x=2.4	4.1517	4.1502
x=2.8	5.1011	5.0996
x=3.2	6.1122	6.1107
x=3.6	7.1826	7.1812
x=4.0	8.3102	8.3090
time req'd	44.25s	72s

The results from these two methods are obviously in good agreement, with Wragg's method requiring considerably less computer time than the Douglas-Gallie method.

6.2 In this section we set out to analyse the errors introduced into the first equation of (6) by perturbing it by $(\tau' + \tau'' x)T_n^*(x)$. The first two equations of (6) are

$$\frac{d^2U_{i+1}}{dx^2} - \frac{U_{i+1}}{\Delta t} + \frac{U_i}{\Delta t} = 0 \qquad (7)$$

$$\left[\frac{dU_{i+1}}{dx}\right]_{x=0} = 0 \quad .$$

Let U_{i+1} be the exact solution to (7). Perturbing (7) by $(\tau' + \tau'' x)T_n^*(x)$ leads to

$$\frac{d^2U_{i+1}}{dx^2} - \frac{U_{i+1}}{\Delta t} + \frac{U_i}{\Delta t} = (\tau' + \tau'' x)\, T_n^*(x) \quad . \qquad (8)$$

Replace U in (7) by $\tilde{U}$, subtract (8) from it and let $z_i = \tilde{U}_i - U_i$. Then

$$\frac{d^2z_{i+1}}{dx^2} - \frac{z_{i+1}}{\Delta t} + \frac{z_i}{\Delta t} = -(\tau' + \tau'' x)T_n^*(x)$$

$$= -w(x) \text{ say.} \qquad (9)$$

The solution

$$z_{i+1}(x) = (1 - \Delta t \frac{d^2}{dx^2})^{-[i]-1} z_{i-[i]-1}(x) +$$

$$+ \Delta t \sum_{k=0}^{[i]} (1 - \Delta t \frac{d^2}{dx^2})^{-k-1} w(x)$$

is obtained by applying the Euler-Laplace transform to (9) , as in 5.14 and then inverting. Restricting i to the set of non-negative integers, the solution reduces to

$$z_{i+1}(x) = (1 - \Delta t \frac{d^2}{dx^2})^{-i-1} z_{-1}(x) + t \sum_{k=0}^{i} (1 - \Delta t \frac{d^2}{dx^2})^{-k-1} w(x).$$

If we assume that $z_{-1}(x) = 0$ it follows immediately that

$$z_i(x) \approx i.\Delta t.w(x) + \Delta t \sum_{k=1}^{\infty} \left\{ 1 + \sum_{j=2}^{i} \binom{-j}{k} \right\} \Delta t^k \frac{d^{2k}}{dx^{2k}} w(x) . \qquad (10)$$

<u>6.3</u> As a particular realisation of this, set $\tau' = \tau'' = 10^{-5}$ and $\Delta t = 0.01$, 0.04 and 0.10 (all these are typical values, taken from the paper by Wragg) . We then calculated the z_i's for $\Delta t \leq t \leq 1.00$ at the 9 points x=0.000(0.125)1.000 for $T^*_n(x)$ with n=3(1)8. Table 1 summarizes, very briefly, the many results computed - we have shown the error given by (10) at x = 0.500 and t = 1.

	Δt		
n	.01	.04	.10
3	$.606_{-4}$	$.620_{-4}$	$.636_{-4}$
4	$.840_{-2}$	$.912_{-2}$	$.107_{-1}$
5	$-.669_{-2}$	$-.729_{-2}$	$-.855_{-2}$
6	$-.103_{1}$	$-.121_{1}$	$-.164_{1}$
7	$.116_{1}$	$.137_{1}$	$.186_{1}$
8	$.190_{3}$	$.250_{3}$	$.410_{3}$

<u>table 1</u>

The conclusions to be drawn from this (and our many unpublished results) are:-

a) The errors increase with increasing n (!);

b) The error increases with increasing t, as tables 2 and 3 illustrate

c) The error (for a fixed n) is not dramatically improved by decreasing Δt.

	Δt		
t	.01	.04	.10
0.1	$.636_{-6}$	-	-
0.2	$.275_{-5}$	$.250_{-5}$	$.120_{-5}$
0.3	$.556_{-5}$	-	$.480_{-5}$
0.4	$.982_{-5}$	$.102_{-4}$	$.960_{-5}$
0.5	$.153_{-4}$	-	$.156_{-4}$
0.6	$.219_{-4}$	$.227_{-4}$	$.228_{-4}$
0.7	$.298_{-4}$	-	$.312_{-4}$
0.8	$.389_{-4}$	$.399_{-4}$	$.408_{-4}$
0.9	$.491_{-4}$	-	$.516_{-4}$
1.0	$.606_{-4}$	$.620_{-4}$	$.636_{-4}$

n=3 table 2

	Δt		
t	.01	.04	.10
0.1	$.545_{-2}$	-	-
0.2	.101	.289	$.125_{1}$
0.3	.628	-	$.440_{1}$
0.4	$.238_{1}$	$.440_{1}$	$.117_{2}$
0.5	$.681_{1}$	-	$.522_{2}$
0.6	$.162_{2}$	$.250_{2}$	$.522_{2}$
0.7	$.339_{2}$	-	$.953_{2}$
0.8	$.645_{2}$	$.904_{2}$	$.163_{3}$
0.9	$.114_{3}$	-	$.264_{3}$
1.0	$.190_{3}$	$.250_{3}$	$.409_{3}$

n=8 table 3

REFERENCES.

(1) Arnoldi W.E.,1951. The principle of minimized iterations in the solution of the matrix eigenvalue problem. Quart. Appl. Math., 9, 17-29.

(2) Berezin I.S. and Zhidkov N.P.,1965. Computing Methods vol II, (tr. by O.M. Blum). Pergammon.

(3) Bodewig E.,1946. Sur la methode de Laguerre pour l'approximation des racines de certaines equations et sur la critique d'Hermite. Nederl. Akad. Wetensch. Proc.,49, 911-921.

(4) Causey R.L. and Gregory R.T.,1961. On Lanczos' algorithm for tridiagonalizing matrices. SIAM Rev.,3, 322-328.

(5) Collatz L.,1966. The Numerical Treatment of Differential Equations. Springer Verlag.

(6) Dew P.M. and Scraton R.E.,1972. An improved method for the solution of the heat equation in Chebyshev series. J. Inst. Maths. Applics.,9,299-309.

(7) Elliott D.,1961. A method for the numerical solution of the one-dimensional heat equation using Chebyshev series. Camb. Phil. Soc. Proc.,57,823-832.

(8) Faddeev D.K. and Faddeeva V.N.,1963. Computational Methods of Linear Algebra (tr. by R.C. Williams). Freeman and Co.

(9) Fox L.(ed.),1962. Numerical Solution of Ordinary and Partial Differential Equations. Pergammon.

(10) Fox L. and Parker I.V.,1968. Chebyshev Polynomials in Numerical Analysis. Oxford University Press.

(11) Gregory R.T.,1958. Results using Lanczos' method for finding eigenvalues of arbitrary matrices. J.SIAM, 6,182-188.

(12) Kaniel S.,1968. Estimates for some computational techniques in linear algebra. Maths Comp.,20,369-378.

(13) Knibb D.,1973. The numerical solution of parabolic partial differential equations using the method of Lanczos. J.Inst. Maths. Applics. 11,181-190.

(14) Knibb D. and Scraton R.E.,1971. On the solution of parabolic partial differential equations in Chebyshev series. Comp. J., 14,428-432.

(16) Laguerre E.N. Oevres de Laguerre, Gauthier-Villars, Paris, vol I,87-103.

(17) Lanczos C.,1950. Trigonometric interpolation of empirical and analytic functions. J.Math. Phys.,17,123-199.

(18) Lanczos C.,1950. An iteration method for the solution of the eigenvalue problem of linear differential and integral problems. J. Res. NBS.,45,255-282.

(19) Lanczos C.,1952. Tables of Chebyshev polynomials. NBS Applied Math. Series No 9.

(20) Lanczos C.,1957. Applied Analysis. Pitman.

(21) Mason J.C.,1965. Some new approximations for the solution of differential equations. Oxford D.Phil. Thesis.

(22) Mikhlin S.G.,1964. Variational Methods in Mathematical Physics (tr. by T. Boddington). Pergammon.

(23) Mikhlin S.G. and Smolitsky K.L.,1967. Approximate Methods for the Solution of Differential and Integral Equations (tr. from Russian). Elsevier.

(24) Ortiz E.L.,1969. The tau method. SIAM J. Num. An., 6,480-492.

(25) Paige C.C.,1972. Computational variants of the Lanczos method for the eigenproblem. J. Inst. Maths. Applics.,10,373-381.

(26) Parlett B.,1964. Laguerre's method applied to the matrix eigenvalue problem. Maths Comp.,18,464-485.

(27) Peters G. and Wilkinson J.H.,1971. Practical problems arising in the solution of polynomial equations. J. Inst. Maths Applics., 8,16-35.

(28) Tewarson R.P.,1970. On the reduction of a sparse matrix to Hessenberg form. Int. J. Comp. Math.,2,283-295.

(29) Tewarson R.P.,1973. Sparse Matrices. Maths in Science and Engineering Series, Vol. 99. Academic.

(30) van der Corput J.G.,1964. Sur l'approximation de Laguerre des racins d'une equation qui a toutes ses racines rielles. Nederl. Akad. Wetensch. Proc.,49,922-929.

(31) Weinberg A.M. and Wigner E.P.,1958. The Physical Theory of Neutron Chain Reactors. Univ. of Chicago Press.

(32) Westlake J.R.,1968. A handbook of Numerical Matrix Inversion and Solution of Linear Equations. John Wiley and Sons.

(33) Wilkinson J.H.,1958. The calculation of eigenvectors by the method of Lanczos. Comp. J.,1,148-152.

(34) Wilkinson J.H.,1963. Rounding Error in Algebraic Processes. Her Majesty's Stationery Office.

(35) Wilkinson J.H.,1965. The Algebraic Eigenvalue Problem. Oxford University Press.

(36) Wilkinson J.H. and Reinsch C.,1971. Handbook for Automatic Computation; Vol. II. Linear Algebra. Springer Verlag.

(37) Wragg A.,1966. The use of Lanczos-tau methods in the solution of a Stefan problem. Comp.J.,9,106-109.

(38) Clenshaw C.W.,1957. The numerical solution of linear differential equations in Chebyshev series. Proc. Camb. Phil. Soc.,53,134-149.

(39) Bellman R. and Cooke K.L.,1963. Differential Difference Equations. Academic Press.

(40) Pinney E.,1958. Ordinary Difference-differential Equations. Univ. of California Press, Berkeley.

APPENDIX TO: NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS

by Colin John Wright

A thesis submitted to the Imperial College of Science and Technology of the University of London for the degree of Master of Philosophy.

August 1976.

We collect together, in this appendix, some of the programs used in the main body of this thesis - namely that used in Chapter two of Part one for the numerical determination of the eigenvalues of a certain differential operator defined there and that used for the solution of the Poisson equation in sections 3.7 and 3.8 of Part two.

I : Programs used in the isolation of the eigenvalues of the differential operator defined in Part 1: Chapter 2.

COLDIF produces the coefficient matrix in segmented form.

COL1 triangularizes the coefficient matrix, then produces its smallest (in modulus) 12 eigenvalues.

More detailed descriptions of the activities of various segments of these programs appear alongside and after them.

Program COLDIF :

```
      DIMENSION A(20,3),AU(20),FMT(3)
      DIMENSION BDR(65,2),ABDRY(2,64),RAD(65),NODE(65)
      INTEGER BAND
      COMMON A,AU,RAD,ABDRY,BDR,HSQ,THETA2,B,ROUT,RIN,H,PI,THET
      COMMON ABDR1,ABDR2,D1,ROUND,N,NODE,N2P1,N1,IRAD,K
      COMMON NMAX,N2,BAND
      WRITE(6,1001)
 1001 FORMAT(' WANTS OUTPUT FORMAT..MUST BE (1X,****)')
      READ(9,1002)(FMT(I),I=1,3)
 1002 FORMAT(3A4)
 1000 WRITE(6,666)
 666  FORMAT(' REQUIRES... NTHETA,IB,NH../..RIN,ROUT,B,BINC')
      READ(9,1)NTHETA,IB,NH
 1    FORMAT(3Y)
      IB=IB+1
      READ(9,4)RIN,ROUT,B,BINC
 4    FORMAT(4Y)
      ROUND=2.E-5
      F=1./RIN
      PI=4.*ATAN(1.)
      THETA=(2.*PI)/FLOAT(NTHETA)
      H=RIN/FLOAT(NH)
      HSQ=H*H
      THETA2=THETA*THETA
      N1=NTHETA/4+1
      N2=NTHETA/2
      N2P1=N2+1
      C1=-1./HSQ
      C2=2./HSQ
      CON=2./THETA2-0.25
      D1=-1./(THETA2*RIN**2)
      ABDR1=-F*(F-1./RIN)-2.*D1
      ABDR2=0.
      DO 1022 NB=1,IB
      WRITE(6,1023)ROUT,RIN,B
 1023 FORMAT(' OUTER CIRCLE RADIUS',F7.3,6X,' INNER CIRCLE RADI
      WRITE(6,1024)H,N2,THETA
 1024 FORMAT(' RADIUS STEP LENGTH',E11.4,6X,'ANGULAR STEP LENGT
      DO 3 I=1,N1
      AA=FLOAT(I-1)*THETA
      AA=COS(AA)
      DD=SQRT(B*B*(AA*AA-1.)+ROUT*ROUT)
      AA=B*AA
      R1=ABS(AA+DD)
      R2=ABS(AA-DD)
      NODE(I)=(R1-RIN+ROUND)/H+1.
      RAD(I)=R1
      IF(I.EQ.N1)GO TO 3
      K=N2+2-I
      RAD(K)=R2
      NODE(K)=(R2-RIN+ROUND)/H+1.
3     CONTINUE
      N=NCDE(1)
      NMAX=NODE(1)
      IMAX=1
      DO 30 J=2,N2P1
      IF(NMAX.GT.NODE(J))GO TO 30
      NMAX=NODE(J)
      IMAX=J
30    N=N+NODE(J)
      WRITE(6,600)(NODE(I),I=1,N2P1)
600   FORMAT(20I5)
      DO 99 J=1,NMAX
99    AU(J)=0.
      DO 220 I=1,NMAX
      DO 220 J=1,3
220   A(I,J)=0.
      DO 31 J=2,NMAX
      R=RIN+FLOAT(J-1)*H
      A(J,1)=C1
      A(J,2)=C2+CON/(R*R)
      A(J,3)=C1
31    AU(J)=-1./(THETA2*R*R)
      DO 301 J=1,2
      DO 300 I=1,N2P1
300   BDR(I,J)=0.
      DO 301 I=1,N2
301   ABDRY(J,I)=0.
      DO 102 IRAD=1,N2P1
      K=NODE(IRAD)
      CALL DEFINE(NP6)
      N=N+NP6
      NODE(IRAD)=NODE(IRAD)+NP6
      IF(NP6.LT.0)NODE(IRAD)=-NODE(IRAD)
102   CONTINUE
      IF(NODE(IMAX).LT.0)NMAX=NMAX-1
      BAND=2*NMAX+1
      FFF=ABS(A(2,1))
      DO 1019 I=2,NMAX
      DO 1019 J=1,3
      IF(FFF.LT.ABS(A(I,J)))FFF=ABS(A(I,J))
1019  CONTINUE
      FFF=ALOG(FFF)/ALOG(16.)
      IFFF=FFF+1
      FFF=16.**IFFF
      IF(FFF.LT.1.)FFF=1.
      D1=D1/FFF
      ABDR1=ABDR1/FFF
      ABDR2=ABDR2/FFF
      DO 1020 I=2,NMAX
      DO 1021 J=1,3
1021  A(I,J)=A(I,J)/FFF
1020  AU(I)=AU(I)/FFF
      DO 1025 IRAD=1,N2P1
      DO 1025 J=1,2
      IF(IRAD.EQ.N2P1)GO TO 1025
      ABDRY(J,IRAD)=ABURY(J,IRAD)/FFF
1025  BDR(IRAD,J)=BDR(IRAD,J)/FFF
      WRITE(6,110)N
110   FORMAT(' MATRIX IS OF ORDER',I4)
```

Coefficients
scaled if
necessary.

```
      WRITE(6,113)
113   FORMAT(/)
      WRITE(6,108)ABDR1,ABDR2,D1,FFF
      WRITE(6,113)
      DO 112 IRAD=1,N2P1
112   WRITE(6,108)(BDR(IRAD,J),J=1,2)
      WRITE(6,113)
      DO 111 I=2,NMAX
111   WRITE(6,108)(A(I,J),J=1,3)
      WRITE(6,113)
      WRITE(6,108)(AU(J),J=1,NMAX)
      WRITE(6,113)
      DO 1070 I=1,N2
1070  WRITE(6,108)(ABDRY(J,I),J=1,2)
108   FORMAT(13E10.3)
      WRITE(9,202)N,N2P1,NMAX,N2,BAND
      WRITE(9,200)(NODE(I),I=1,N2P1)
      WRITE(9,FMT)ABDR1,ABDR2,D1,FFF
      WRITE(9,FMT)((BDR(IRAD,J),J=1,2),IRAD=1,N2P1)
      WRITE(9,FMT)((A(I,J),J=1,3),I=1,NMAX)
      WRITE(9,FMT)(AU(J),J=1,NMAX)
      WRITE(9,FMT)((ABDRY(J,I),J=1,2),I=1,N2)
200   FORMAT(1X,25I3)
202   FORMAT(1X,5I4)
1022  B=B+BINC
      STOP
      END
      SUBROUTINE DEFINE(NP6)
      DIMENSION A(20,3),AU(20)
      DIMENSION ABDRY(2,64),BDR(65,2),RAD(65),NODE(65)
      INTEGER BAND
      COMMON A,AU,RAD,ABDRY,BDR,HSQ,THETA2,B,ROUT,RIN,H,PI,THETA
      COMMON ABDR1,ABDR2,D1,ROUND,N,NODE,N2P1,N1,IRAD,K
      COMMON NMAX,N2,BAND
      R=RIN+FLOAT(K-1)*H
      NP6=0
      P5=(RAD(IRAD)-R)
      IF(ABS(P5).GT.ROUND)GO TO 32
      NP6=NP6-1
      GO TO 8
32    P1=1.
      P5=P5/H
      CON5=1.
      IF(IRAD.EQ.N2P1)GO TO 6
      IF(K.LE.NODE(IRAD+1))GO TO 6
      AA=ABS((B*B+R*R-ROUT*ROUT)/(2.*B*R))
      IF(AA.GT.1.)GO TO 30
      AA=ATAN(SQRT(1./AA-AA))
      GO TO 31
30    AA=0.
31    IF(IRAD.LE.N1)GO TO 5
      AA=PI-AA
5     P1=AA/THETA-FLOAT(IRAD-1)
      IF(P1.LE.(1.+ROUND))CON5=0.
6     P3=1.
```

```
      AAA=2./(R*R*THETA2*(P1+P3))
      BDR(IRAD,2)=2./(P5*HSQ)+(2./(THETA2*P1*P3)+0.25)/(R*R)
      BDR(IRAD,1)=2./(HSQ*(P5+1.))*(-1.)
      IF(IRAD.EQ.N2P1)GO TO 70
      ABDRY(1,IRAD)=-AAA*CON5/P1
70    IF(IRAD.EQ.1)GO TO 8
      ABDRY(2,IRAD-1)=-AAA/P3
8     RETURN
      END
```

The computational details of this program are clear if it is read in conjunction with the expressions (7), (11), (14) of Part 1: Chapter 2 and the following symbol table.

SYMBOL	MEANING
ROUT	radius of outer circle
RIN	radius of hole
B	distance between centres
BINC	increment in distance between centres
IB	number of times distance between centres is to be incremented
NTHETA	number of angular step-lengths in 2π , i.e. $\Delta\theta = \frac{2\pi}{\text{NTHETA}}$
NH	number of radius steps in hole, i.e. $\Delta r = \frac{\text{RIN}}{\text{NH}}$
THETA	$\Delta\theta$
THETA2	$\Delta\theta^2$
H	Δr
HSQ	Δr^2
RAD(I)	radius of (i+1)th ray
NODE(I)	number of nodes along (i+1)th ray
N	order of coefficient matrix
BAND	bandwidth of coefficient matrix
FFF	scaling factor

```
CCL1
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*4 P(20,500),BDR(65,2),A(12,3),AU(12),ABDRY(2,64)
     1,ABDR1,ABDR2,D1,FFF,RIN,DIST
      DIMENSION V1(500),V1STA(500),U1STA(500),ALPHA(500)
     1,GAM(500),W(500),WSTA(500),VJP1(500),VSTAJ1(500),VV(500)
     1,VVST(500),U1(500),NODE(65)
      INTEGER BAND
      INTEGER*2 SIGN(500)
      COMMON A,AU,ABDRY,BDR,ABDR1,ABDR2,D1,NODE,N2P1,NMAX,N2
      COMMON/STORE/P/DF2/ALPHA,GAM,FFF,SIGN
      COMMON/PROD/VJP1,U1/PRODA/VSTAJ1,U1STA
      EQUIVALENCE(M,N)
      READ(5,202)M,N2P1,NMAX,N2,BAND
      READ(5,200)(NODE(I),I=1,N2P1)
      READ(5,201)ABDR1,ABDR2,D1,FFF                        Data
      READ(5,201)((BDR(IRAD,J),J=1,2),IRAD=1,N2P1)         input
      READ(5,201)((A(I,J),J=1,3),I=1,NMAX)                 from
      READ(5,201)(AU(I),I=1,NMAX)                          COLDIF
      READ(5,201)((ABDRY(J,I),J=1,2),I=1,N2)
      READ(5,201)RIN,DIST
  200 FORMAT(1X,25I3)
  201 FORMAT(1X,8Z9)
  202 FORMAT(1X,5I4)
      CALL COLMAS(N,BAND)
      NB2P1=BAND/2+1
      ISN=1
      KIP=0
      WRITE(6,2019)RIN,DIST
 2019 FORMAT(' INNER RADIUS',F6.3,3X,'DISTANCE',F6.3)
      WRITE(6,2020)FFF
 2020 FORMAT(' SCALING FACTOR IS',E14.7)
      RATLM=.6D0
      DO 300 I=1,N
      IF(I/2*2.EQ.I)GO TO 301
      V1(I)=1.D0
      V1STA(I)=0.D0
      GO TO 300
  301 V1(I)=0.D0                                          Definition
      V1STA(I)=1.D0                                       of initial
  300 CONTINUE                                            vectors
      V1STA(1)=1.D0
      GO TO 1002
 1001 IF(KIP.LT.0)STOP
      DO 1003 I=1,N
      V1(I)=1.D0
 1003 V1STA(I)=1.D0
      KIP=-1
      WRITE(6,1004)KIP
 1004 FORMAT(' KIP=',I3)
 1002 ALPH=0.D0
      DO 31 I=1,N                                         Scaling
   31 ALPH=ALPH+V1(I)*V1STA(I)                            of v1
      S1=DABS(ALPH)/ALPH                                  & v1*.
      ALPH=DSQRT(DABS(ALPH))
      DO 32 I=1,N
      V1(I)=V1(I)/ALPH
   32 V1STA(I)=V1STA(I)/ALPH
      DO 33 I=1,N
      VV(I)=V1(I)
   33 VVST(I)=V1STA(I)
      DO 40 I=1,N                                         Construction
   40 VJP1(I)=V1(I)                                       of u1
      CALL MULT(N,BAND)                                   & u1*.
      DO 41 I=1,N
```

```
41    VSTAJ1(I)=V1STA(I)
      CALL AMULT(N,BAND)
      DO 13 J=1,N
      ALPH=0.D0
      DO 6 K=1,N
6     ALPH=ALPH+V1STA(K)*U1(K)
      ALPHA(J)=ALPH*S1
      IF(J.EQ.N)GO TO 13
      DO 7 I=1,N
      W(I)=U1(I)-ALPHA(J)*V1(I)
7     WSTA(I)=U1STA(I)-ALPHA(J)*V1STA(I)
      ALPH=0.D0
      DO 8 I=1,N
8     ALPH=ALPH+WSTA(I)*W(I)
      IF(J.EQ.1.AND.DABS(ALPH).LT.1.D-20)GO TO 1001
      IF(J.NE.1.AND.DABS(ALPH).LT.1.D-20)GO TO 131
      S2=DABS(ALPH)/ALPH
      GAM(J+1)=DSQRT(DABS(ALPH))
      DO 9 I=1,N
      VJP1(I)=W(I)/GAM(J+1)
9     VSTAJ1(I)=WSTA(I)/GAM(J+1)
      AA=0.D0
      AASTA=0.D0
      DO 70 I=1,N
70    AA=AA+VV(I)*VSTAJ1(I)
      BET=S1*S2*GAM(J+1)
      JP1=J+1
      IF(S1*S2.GT.0.D0)GO TO 72
      ISN=ISN+1
      SIGN(ISN)=J+1
72    IF(J.LT.(2*N)/3)GO TO 71
      RATIO=1.D0-DFLOAT(ISN-1)/DFLOAT(J)
      IF(RATIO.GT.RATLM)GO TO 131
71    CALL MULT(N,BAND)
      CALL AMULT(N,BAND)
      DO 710 I=1,N
710   AASTA=AASTA+VVST(I)*U1(I)
      WRITE(6,130)AA,AASTA
      DO 12 I=1,N
      U1(I)=U1(I)-BET*V1(I)
      U1STA(I)=U1STA(I)-BET*V1STA(I)
      V1(I)=VJP1(I)
12    V1STA(I)=VSTAJ1(I)
      S1=S2
13    CONTINUE
      GO TO 132
131   N=J
132   WRITE(6,134)RATIO
134   FORMAT(' SYMMETRY RATIO IS',F5.2)
      WRITE(6,135)N
135   FORMAT(I4,' LANCZOS ITERATIONS WERE PERFORMED')
      WRITE(6,130)(ALPHA(I),I=1,N)
      WRITE(6,130)(GAM(I),I=2,N)
130   FORMAT(1X,9D14.7)
37    FORMAT(2I3,D14.7)
      SIGN(1)=ISN
      WRITE(6,133)(SIGN(I),I=1,ISN)
133   FORMAT(30I4)
      CALL COLOF2(N)
      STOP
      END
```

Lanczos iteration procedure.

```
      SUBROUTINE COLMAS(M,BAND)
      DIMENSION NODE(65),BDR(65,2),A(12,3),AU(12),ABDRY(2,64)
     1,P(20,500)
      INTEGER ROW,BAND
      COMMON A,AU,ABDRY,BDR,ABDR1,ABDR2,D1,NODE,N2P1,NMAX,N2
      COMMON/STORE/P
      NB2=BAND/2+1
      NB3=NB2+1
      NB1=NB2-1
      WRITE(6,200)M,NMAX
200   FORMAT(2I4)
      MP1=M+1
      DO 100 I=1,BAND
      DO 100 J=1,MP1
100   P(I,J)=0.
      ROW=0
      DO 120 K=1,N2P1
      FAC1=1.
      FAC2=1.
      IF(K.EQ.N2)FAC1=2.
      IF(K.EQ.2)FAC2=2.
      N=IABS(NODE(K))
      N1=N
      IF(NODE(K).LT.0)N=N+1
      IF(K.GT.1)NB=IABS(NODE(K-1))
      DO 12 I=1,N
      IF(NODE(K).LT.0.AND.I.EQ.N)GO TO 12
      ROW=ROW+1
      IF(I.GT.1)GO TO 2
      P(NB2,ROW)=ABDR1
      P(NB3,ROW)=A(2,1)
      IF(K.EQ.N2P1)GO TO 1
      P(NB2+N1,ROW)=D1*FAC1
1     IF(K.EQ.1)GO TO 12
      P(NB2-NB,ROW)=D1*FAC2
      GO TO 12
2     IF(I.GT.2)GO TO 4
      IF(N1.EQ.N.AND.I.EQ.(N-1))GO TO 7
      P(NB2,ROW)=A(2,2)
      P(NB1,ROW)=ABDR2
      IF(NODE(K).LT.0.AND.I.EQ.N1)GO TO 20
      P(NB3,ROW)=A(3,1)
20    IF(K.EQ.N2P1)GO TO 3
      P(NB2+N1,ROW)=AU(2)*FAC1
3     IF(K.EQ.1)GO TO 12
      P(NB2-NB,ROW)=AU(2)*FAC2
      GO TO 12
4     IF(I.GE.(N-1))GO TO 7
      P(NB2,ROW)=A(I,2)
      P(NB1,ROW)=A(I-1,3)
      IF(I.EQ.2)P(4,ROW)=ABDR2
      P(NB3,ROW)=A(I+1,1)
      P(NB2+N1,ROW)=AU(I)*FAC1
6     IF(K.EQ.1)GO TO 12
      P(NB2-NB,ROW)=AU(I)*FAC2
      GO TO 12
7     IF(I.EQ.N)GO TO 10
      P(NB2,ROW)=A(I,2)
      P(NB1,ROW)=A(I-1,3)
      IF(I.EQ.2)P(4,ROW)=ABDR2
      IF(NODE(K).LT.0)GO TO 8
      P(NB3,ROW)=BDR(K,1)
8     IF(K.EQ.N2P1)GO TO 9
```

```
      P(NB2+N1,ROW)=AU(I)*FAC1
9     IF(K.EQ.1)GO TO 12
      P(NB2-NB,ROW)=AU(I)*FAC2
      GO TO 12
10    IF(NODE(K).LT.0)GO TO 12
      P(NB2,ROW)=BDR(K,2)
      P(NB1,ROW)=A(I-1,3)
      IF(K.EQ.N2P1)GO TO 11
      IF(N1.GT.IABS(NODE(K+1)))GO TO 11
      P(NB2+N1,ROW)=ABDRY(2,K)*FAC1
11    IF(K.EQ.1)GO TO 12
      IF(NODE(K-1).LT.0.OR.NB.GT.N1)GO TO 110
      P(NB2-NB,ROW)=ABDRY(1,K-1)*FAC2
      GO TO 12
110   P(NB2-NB,ROW)=AU(I)*FAC2
12    CONTINUE
120   CONTINUE
      RETURN
      END
```

COLMAS massages input data into banded matrix form.

```
      SUBROUTINE MULT(M,BAND)
      REAL*8 AP(500),Q(500)
      DIMENSION P(20,500)
      INTEGER BAND
      COMMON/STORE/P/PROD/Q,AP
      NB2P1=BAND/2+1
      DO 1 K=1,M
      KK=K+NB2P1
      AP(K)=0.
      DO 1 I=1,BAND
      L=KK-I
      IF(L.LE.0.OR.L.GT.M)GO TO 1
      IF(P(I,L).NE.0.)AP(K)=AP(K)+DBLE(P(I,L))*Q(L)
1     CONTINUE
      RETURN
      END
```

With the banded matrix denoted by P MULT forms AP = P * Q in double precision.

```
      SUBROUTINE AMULT(M,BAND)
      REAL*8 AP(500),Q(500)
      DIMENSION P(20,500)
      INTEGER BAND
      COMMON/STORE/P/PRODA/Q,AP
      NB2P1=BAND/2+1
      DO 1 K=1,M
      AP(K)=0.
      L=K-NB2P1
      IF(L.LT.0)L=0
      DO 1 J=1,BAND
      IF((K+J).LE.NB2P1.OR.(K+J).GT.(M+NB2P1))GO TO 1
      L=L+1
      IF(P(J,K).NE.0.)AP(K)=AP(K)+DBLE(P(J,K))*Q(L)
1     CONTINUE
      RETURN
      END
```

AMULT forms AP = $P^T * Q$.

```
      SUBROUTINE COLDF2(N)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 Z,P0,P1,P0D,P1D,P0DD,P1DD,AMZ,P2,P2D,P2DD,AD,AD
      COMPLEX*16 TC,S1,S2,S,W,SOL(15),G
      DIMENSION ALPHA(500),GAM(500)
      REAL*4 FFF
      INTEGER*2 SIGN(500)
      COMMON/DF2/ALPHA,GAM,FFF,SIGN/COM/S,SR,SI/SOLN/SOL
      CALL ERRSET(208,0,-1)
      NO=12
      IRITE=1                This is the Laguerre root
      FF=DBLE(FFF)
      EPS=2.D-4              finding algorithm.
      ACPT1=2.D-8
      ACPT2=1.D-5
      ACPT3=1.D-8
      DN=DFLOAT(N)
      WARG=DATAN(1.D0)
      DPI=4.D0*WARG
      ITER=30
      PQ=1.D55
      PS=2.D-40
      WRITE(6,9930)ACPT1,ACPT2,ACPT3
 9930 FORMAT(3D10.3)
      C=0.D0
      DO 992 I=2,N
 992  GAM(I)=GAM(I)*GAM(I)
      KK=SIGN(1)
      IF(KK.LE.1)GO TO 994
      DO 993 I=2,KK
      KL=SIGN(I)
 993  GAM(KL)=-GAM(KL)
 994  DO 800 NROOT=1,NO
      WRITE(6,6000)NROOT
 6000 FORMAT(' N=',I4)
      Z=DCMPLX(0.D0,0.D0)
      NSCAL=0
      GO TO 6005
 6002 CALL SCALE(Z,FF,NROOT,1,IR,NSCAL,N)
      GO TO 6003
 6004 CALL SCALE(Z,FF,NROOT,2,IR,NSCAL,N)
 6003 WRITE(6,55)P2,P2D,P2DD,IR
 55   FORMAT(' AT 6003',6D10.3,I5)
      IF(NSCAL.GT.5)GO TO 600
 6005 DIF1=1000.D0
      DO 7 NOIT=1,ITER
      P0=DCMPLX(1.D0,0.D0)
      P1=DCMPLX(ALPHA(1),0.D0)-Z
      P0D=DCMPLX(0.D0,0.D0)
      P1D=DCMPLX(-1.D0,0.D0)
      P0DD=DCMPLX(0.D0,0.D0)
      P1DD=DCMPLX(0.D0,0.D0)
      DO 3 IR=2,N
      AMZ=DCMPLX(ALPHA(IR),0.D0)-Z
      P2=AMZ*P1-GAM(IR)*P0
      P2D=AMZ*P1D-P1-GAM(IR)*P0D
      P2DD=AMZ*P1DD-2.D0*P1D-GAM(IR)*P0DD
      IF(CDABS(P2).LE.PS.OR.CDABS(P2D).LE.PS.OR.
     1CDABS(P2DD).LE.PS)GO TO 6004
      IF(CDABS(P2).GT.PQ.OR.CDABS(P2D).GT.PQ.OR.
     1CDABS(P2DD).GT.PQ)GO TO 6002
      P0=P1
      P0D=P1D
      P0DD=P1DD
      P1=P2
      P1D=P2D
3     P1DD=P2DD
      AP1=CDABS(P1)
      AD=DCMPLX(0.D0,0.D0)
      ADD=DCMPLX(0.D0,0.D0)
      IF(NROOT.EQ.1)GO TO 5
      NN=NROOT-1
      DO 4 I=1,NN
      G=Z-SOL(I)
      T=CDABS(G)
      IF(T.LE.8.D-15)WRITE(6,6001)I,NROOT
6001  FORMAT(' ROOT',I4,' AND AN ITERATE OF ROOT',I4,
     1' ARE PATHALOGICALLY CLOSE')
      IF(T.LT.2.D-20)G=DCMPLX(1.D3,0.D0)
      TC=1.D0/G
      ADD=ADD+TC
4     AD=AD+TC*TC
5     IF(AP1.LE.2.D-30)WRITE(6,54)AP1
54    FORMAT(' AP1 TOO SMALL..',D10.3)
      S1=P1D/P1
      S2=S1*S1-P1DD/P1-AD
      S1=S1-ADD
      W=(DN-1.D0)*(DN*S2-S1*S1)
      W=CDSQRT(W)
      S=DCONJG(S1)
      S=S*W
      CALL RLIM
      IF(DABS(SR).GT.2.D-6)GO TO 51
      WMOD=CDABS(W)
      SR=WMOD*DCOS(WARG)
      SI=WMOD*DSIN(WARG)
      GO TO 52
51    IF(SR.GT.0.D0)GO TO 53
      S=W
      CALL RLIM
      SR=-SR
      SI=-SI
52    W=DCMPLX(SR,SI)
53    W=DN/(S1+W)
      Z=Z-W
      AZ=CDABS(Z)
      AW=CDABS(W)
      C1=CDABS(P1D)
      IF(IRITE.EQ.1)WRITE(6,91)Z,P1,P1D,P1DD,W
91    FORMAT(2D18.10,8D10.3)
      IF(AP1.LE.(ACPT1*AZ*C1))GO TO 81
      IF(AW.GE.EPS)GO TO 6
      IF(AW.GT.DIF1)GO TO 84
      DIF1=AW
```

```
 6     IF(AZ.LE.8.D-5)GO TO 60
       WMOD=AW/AZ
       IF(WMOD.LT.ACPT2)GO TO 82
 60    IF(AW.LE.(ACPT3*C))GO TO 83
 7     CONTINUE
       KK=99
       GO TO 20
 81    KK=1
       GO TO 20
 82    KK=2
       GO TO 20
 83    KK=3
       GO TO 20
 84    KK=4
 20    IF(C.LT.CDABS(Z))C=CDABS(Z)
       SOL(NROOT)=Z
       WRITE(6,21)KK,Z,P1,P1D,P1DD,W,NOIT
 21    FORMAT(I4,2D14.7,/,4(2X,2D10.3),I4)
 800   CONTINUE
       GO TO 601
 600   NO=NROOT-1
       IF(NO.EQ.0)STOP
 601   DO 801 I=1,NO
 801   SOL(I)=SOL(I)*FF
       WRITE(6,802)NO
 802   FORMAT(///,2X,I3,' OF ROOTS ARE...')
       WRITE(6,803)(SOL(I),I=1,NO)
 803   FORMAT(3(4X,2D17.10))
       STOP
       END
```

```
       SUBROUTINE RLIM
       REAL*8 X,Y,FR,FI
       COMMON /COM/X,Y,FR,FI
       FR=X
       FI=Y
       RETURN
       END
```

RLIM extracts the real and imaginary parts of $z = x + i\ y$.

```
      SUBROUTINE SCALE(Z,FF,NROOT,KIP,J,NSCAL,N)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ALPHA(500),GAM(500)
      COMPLEX*16 SOL(15),Z
      REAL*4 FFF
      INTEGER*2 SIGN(500)
      COMMON/DF2/ALPHA,GAM,FFF,SIGN/SOLN/SOL
      NSCAL=NSCAL+1
      IF(NSCAL.GT.5)RETURN
      IF(J.LE.(N/2))PQ=30.D0/DFLOAT(J)
      IF(J.GT.(N/2))PQ=20.D0/DFLOAT(J)
      PQ=10.D0**PQ
      IF(KIP.EQ.2)GO TO 992
      F1=PQ
      GO TO 993
992   F1=1.D0/PQ
993   ALPHA(1)=ALPHA(1)/F1
      DO 995 I=2,N
      ALPHA(I)=ALPHA(I)/F1
995   GAM(I)=GAM(I)/F1**2
      WRITE(6,901)
901   FORMAT(///)
      WRITE(6,900)(ALPHA(I),I=1,N)
      WRITE(6,900)(GAM(I),I=2,N)
900   FORMAT(2X,12D10.3)
      FF=FF*F1
      IF(NROOT.EQ.1)GO TO 3
      NN=NROOT-1
      DO 1 I=1,NN
1     SOL(I)=SOL(I)/F1
      Z=Z/F1
3     WRITE(6,2)FF,F1
2     FORMAT(' SCALING FACTOR IS..',2D17.10)
      RETURN
      END
```

If, for some reason or other, the value of the determinant, or of the first or second derivative of the characteristic polynomial of the tridiagonal get out of range, then the tridiagonal matrix, the roots already found and the current estimate are scaled here.

The following symbol table is useful in the interpretation of the program COL1.

SYMBOL	SYMBOL	
	All the symbols listed in the symbol table of COLDIF have the same meaning here, except B which is called DISP here.	
V1	v_j	of the generalized Lanczos method of chapter 1.
V1STA	v_j^*	
VJP1	v_{j+1}	
VSTAJ1	v_{j+1}^*	
U1	u_{j+1}	
U1STA	u_{j+1}^*	
W	w_j	
WSTA	w_j^*	
ALPHA	vector containing α_j's	
GAM	vector containing γ_j's	
SIGN	vector containing signs of the β_j's	
VV	stores v_1	
VVSTA	stores v_1^*	
RATIO	symmetry ratio at that specific point	
AA	$v_j^{*T} v_1$	
AASTA	value of h_{1j}	
P	coefficient matrix in massaged form	

SUBROUTINE COLDF2

ACPT1	constants for the stopping criteria 1,2,3 of 2.3.
ACPT2	
ACPT3	
C	maximum value of the moduli of the roots already found
NROOT	number of root currently being sought
Z	present approximation to root

PO	p_{r-2}	$,r=2,3,\ldots,n$ - see 2.2
P1	p_{r-1}	
POD	p'_{r-2}	
P1D	p'_{r-1}	
PODD	p''_{r-2}	
P1DD	p''_{r-1}	
P2	p_r	
P2D	p'_r	
P2DD	p''_r	
PQ	upper threshold on p_r, p'_r, p''_r.	
PS	lower threshold on p_r, p'_r, p''_r.	
S1	S_1	of 2.3
S2	S_2	
W	defined in 2.3	
SOL	this vector holds the accepted approximations to the roots	

Program description: The routine MAIN of COL1 initially reads in the finite difference coefficients passed to it from COLDIF. Control passes almost immediately to the routine termed COLMAS, where the input data is massaged into banded matrix form.

This matrix has not been densely packed as might easily (?) be done in the case of an extremely large matrix (or a small computer system) - see Tewarson [29]for details of packing techniques. On returning to MAIN the initial vectors v_1 and v_1^* are defined so that $v_1^T v_1^* = S_1$. After having formed u_1 and u_1^* the execution of the Lanczos algorithm commences.

The Lanczos algorithm of chapter 1 is applied as described there. If $\gamma_1 < 10^{-20}$ the algorithm is recommenced with new different initial vectors - if any other γ is less than 10^{-20} control is passed to COLDF2 where the roots of the tridiagonal matrix obtained thus far are sought. If (with no $\gamma < 10^{-20}$) after 2n/3 or more Lanczos steps have been carried out the symmetry ratio (redefined later) is found to be greater than 0.6 control is transferred to COLDF2. During the execution of this section of the algorithm the values of $v_i^T v_1^*$, i=1,..,n as well as $h_{1,r}$, r=3,..,n (see chapter 1 for the definition of h_{ir}) are computed and printed as a running check on the biorthogonality of the computed vectors and on the tridiagonality of the supposedly upper hessenberg form.

II : Program for the solution of the Poisson equation by the matrix-tau-lines technique.

```
         IMPLICIT REAL*8 (A-H,O-Z)
         REAL*8 TCHEB,GAM
         REAL*8 KKK(16),LLL(16),M(16,32)
         DIMENSION RPP(16),SSR(16),A(16,16),B(16,16),CHEB1(22)
         REAL*8 SI(16,16),FFF(16),GGG(16),V(16),CHEB(21)
         DIMENSION XB(18),XBB(18),TK(16),TL(16),TP(16),TR(16)
         DIMENSION XX(16),XINC(16)
         COMMON S,H,H2,N
         LOGICAL EVEN
         N=-10                                 | Requires no of y-steps and
1000     READ(9,400)NN,NTMIN,NTMAX,INC          | range of Cheby perturbations.
         IF(NN.EQ.N)GO TO 5555                 |
         N=NN
 400     FORMAT(4Y)
         CONS=5.D0/6.D0
         CONS1=1.D0/12.D0
         N2=N*2
         NP1=N+1
         NM1=N-1
         H=1.D0/DFLOAT(NP1)
         H2=H*H
         NP2=N+2
         DO 5555 I=1,NP2                                      | Definition of
         XB(I)=DSQRT(0.25-(DFLOAT(I-1)*H-0.5)**2)             | boundary curves.
 5555    XBB(I)=2.-XB(I)                                      |
         AMIN=1.D12
         AMAX=-1.D12
         DO 1002 I=1,NP2
         IF(AMIN.GT.XB(I))AMIN=XB(I)
         IF(AMAX.LT.XBB(I))AMAX=XBB(I)
 1002    CONTINUE
         DO 1005 I=2,NP1
         HY=(XBB(I)-XB(I))/8.D0
         U(1)=XB(I)
         DO 1004 J=2,9
 1004    U(J)=U(J-1)+HY
 1005    WRITE(9,7)(U(J),J=1,9)
         BMA=AMAX-AMIN
         DO 1003 I=2,NP1
         XB(I-1)=(XB(I)-AMIN)/BMA
         XBB(I-1)=(XBB(I)-AMIN)/BMA
 1003    XINC(I-1)=(XBB(I-1)-XB(I-1))/2.D0
         WRITE(9,7)(XB(I),I=1,N)
         WRITE(9,7)(XBB(I),I=1,N)
         DO 2 I=1,N                                           |
         DO 1 J=1,N                                           |
         M(I,J)=0.D0                                          |
         IF(I.EQ.J-1)M(I,J)=-DFLOAT(I)/DFLOAT(J)              |
  1      CONTINUE                                             |
  2      M(I,I)=1.D0                                          |
         DO 5 I=1,N                                           |
         DO 4 J=1,I                                           |
  4      M(I,J+N)=-DFLOAT(J)/DFLOAT(I+1)                      |
         IF(I.EQ.N)GO TO 5                                    |
         IP1=I+1                                              |
         DO 50 J=IP1,N                                        | Construction
 50      M(I,J+N)=0.D0                                        |
  5      CONTINUE                                             | of matrix  M^-1
  7      FORMAT(10D12.5)                                      |
         DO 8 II=1,NM1                                        |
         I=N-II                                               |
         DO 8 J=NP1,N2                                        |
  8      M(I,J)=M(I,J)-M(I,I+1)*M(I+1,J)                      |
         DO 120 I=1,N                                         |
         DO 12 J=1,N                                          |
         A(I,J)=0.D0                                          |
         IF((I.EQ.J+1).OR.(I.EQ.J-1))A(I,J)=CONS1             |
 12      CONTINUE                                             |
 120     A(I,I)=CONS                                          |
         DO 1200 I=1,N                                        |
         DO 1200 J=1,N                                        |
 1200    A(I,J)=A(I,J)/BMA**2                                 |
         DO 13 J=1,N                                          |
         DO 13 I=1,N                                          |
 13      A(I,J)=M(I,J+N)                                      |
```

Construction of matrix S.

```
      DO 14 I=1,N
      DO 14 J=1,N
      B(I,J+N)=A(I,1)*A(1,J)
      DO 14 K=2,N
14    B(I,J+N)=B(I,J+N)+A(I,K)*A(K,J)
      DO 15 I=1,N
      DO 15 J=1,N
15    S(I,J)=B(I,J+N)
5555  DO 1001 NT=NTMIN,NTMAX,INC
      EVEN=.FALSE.
      IF(NT.EQ.NT/2*2)EVEN=.TRUE.
      NTP1=NT+1
      NTP2=NT+2
      SIGN=-1.D0
      HH=1.D0
      DO 190 I=1,N
      KKK(I)=0.D0
      LLL(I)=0.D0
      PPP(I)=0.D0
190   RRR(I)=0.D0
      DO 1900 I=1,NTP2
      CHEB1(I)=TCHEB(NTP1,I-1)
      IF(I.EQ.NTP2)GO TO 1900
      CHEB(I)=TCHEB(NT,I-1)
1900  CONTINUE
      NP132=(NT+1)/2
      NPB21=NP132+1
```

Construction of vectors k,l,p,r.

```
      DO 212 II=1,NPB21
      I=II-1
      SIGN=-SIGN
      IF(I.EQ.1)GO TO 202
      IF(I.GT.1)GO TO 204
      DO 213 IJ=1,N
      DO 214 J=1,N
214   SI(IJ,J)=0.D0
213   SI(IJ,IJ)=1.D0
      GO TO 207
202   DO 203 IJ=1,N
      DO 203 J=1,N
203   SI(IJ,J)=S(IJ,J)
      GO TO 207
204   DO 205 IJ=1,N
      DO 205 J=1,N
      B(IJ,J+N)=SI(IJ,1)*S(1,J)
      DO 205 K=2,N
205   B(IJ,J+N)=B(IJ,J+N)+SI(IJ,K)*S(K,J)
      DO 206 IJ=1,N
      DO 206 J=1,N
206   SI(IJ,J)=B(IJ,J+N)
207   DO 2070 J=1,N
      TK(J)=0.D0
      TL(J)=0.D0
      TP(J)=0.D0
2070  TR(J)=0.D0
      I2P1=I*2+1
      DO 500 MMM=I2P1,NTP2
      MM=MMM-1
      G=GAM(MM,I)
      CK=G*CHEB1(MMM)
      IF(MM.GT.NT)GO TO 501
      CL=G*CHEB(MMM)
501   DO 500 J=1,N
      IF(XB(J).EQ.0.D0)GO TO 5010
      X1=0.D0
      IF(MM.EQ.2*I)X1=1.D0
      GO TO 5011
5010  X1=XB(J)**(MM-2*I)
5011  IF(XBB(J).EQ.0.D0)GO TO 5012
      X2=0.D0
      IF(MM.EQ.2*I)X2=1.D0
      GO TO 5013
5012  X2=XBB(J)**(MM-2*I)
5013  TL(J)=TL(J)+CK*X1
      TR(J)=TR(J)+CK*X2
      IF(MM.GT.NT)GO TO 500
      TK(J)=TK(J)+CL*X1
      TP(J)=TP(J)+CL*X2
500   CONTINUE
```

```
      DO 502 IJ=1,N
      DO 502 J=1,N
      LLL(IJ)=LLL(IJ)+SIGN**HH*SI(IJ,J)*TL(J)
      RRR(IJ)=RRR(IJ)+SIGN**HH*SI(IJ,J)*TR(J)
      IF((EVEN.EQ..FALSE.).AND.I.EQ.NP1/2)GO TO 502
      KKK(IJ)=KKK(IJ)+SIGN**HH*SI(IJ,J)*TK(J)
      PPP(IJ)=PPP(IJ)+SIGN**HH*SI(IJ,J)*TP(J)
502   CONTINUE
      HH=HH+2
212   CONTINUE
      DO 281 I=1,N
      DO 280 J=1,4
280   SI(I,J)=0.D0
      DO 281 K=1,N
      SI(I,1)=SI(I,1)+A(I,K)*KKK(K)
      SI(I,2)=SI(I,2)+A(I,K)*LLL(K)
      SI(I,3)=SI(I,3)+A(I,K)*PPP(K)
281   SI(I,4)=SI(I,4)+A(I,K)*RRR(K)
      DO 215 I=1,N
      KKK(I)=SI(I,1)
      LLL(I)=SI(I,2)
      PPP(I)=SI(I,3)*H2
215   RRR(I)=SI(I,4)*H2
      WRITE(9,2150)NT
2150  FORMAT(///,' NT=',I3)
      WRITE(9,700)
700   FORMAT(' K,L,P,R ARE ...')
      WRITE(9,7)(KKK(J),J=1,N)
      WRITE(9,7)(LLL(J),J=1,N)
      WRITE(9,7)(PPP(J),J=1,N)
      WRITE(9,7)(RRR(J),J=1,N)
      CALL EVALU(XK,TK,0)
      CALL EVALU(XL,TL,2)
      CALL EVALU(XP,TP,0)
      CALL EVALU(XR,TR,2)
      PI=4.D0*DATAN(1.D0)
      DO 216 I=1,N
      FFF(I)=-TL(I)+TK(I)
216   GGG(I)=-TR(I)+TP(I)+1.
      DO 217 I=1,N
      LLL(I)=LLL(I)/KKK(I)
217   FFF(I)=FFF(I)/KKK(I)
      DO 218 I=1,N
      RRR(I)=RRR(I)-LLL(I)*PPP(I)
218   GGG(I)=GGG(I)-FFF(I)*PPP(I)
      DO 2180 I=1,N
2180  GGG(I)=GGG(I)/RRR(I)
      DO 219 I=1,N
219   FFF(I)=FFF(I)-GGG(I)*LLL(I)
      WRITE(9,220)
220   FORMAT(//,' THE 2 SETS OF TAUS ARE..AMENDED..')
      WRITE(9,7)(FFF(I),I=1,N)
      WRITE(9,7)(GGG(I),I=1,N)
      WRITE(9,420)
420   FORMAT(///,' ***THE RESULTS ARE***')
      DO 2900 IJ=1,N
2900  XX(IJ)=XR(IJ)
      DO 311 IX=1,9
      SIGN=-1.D0
      HH=1.D0
      DO 290 I=1,N
      PPP(I)=0.D0
290   RRR(I)=0.D0
      DO 306 II=1,[illegible]
      I=II-1
      SIGN=-SIGN
      IF(I.EQ.0)GO TO 302
      IF(I.EQ.1)GO TO 304
      DO 300 IJ=1,N
      DO 300 J=1,N
      A(IJ,J+N)=SI(IJ,1)*S(1,J)
      DO 300 K=2,N
300   A(IJ,J+N)=A(IJ,J+N)+SI(IJ,K)*S(K,J)
      DO 301 IJ=1,N
      DO 301 J=1,N
301   SI(IJ,J)=A(IJ,J+N)
      GO TO 306
```

Construction of the vectors k,l,p,r.

Definition of boundary condts f and g.

Solution of the system of linear algebraic equations for the taus.

Evaluation of the solution at the specified points.

```
302   DO 303 IJ=1,N
      DO 3030 J=1,N
3030  SI(IJ,J)=0.D0
303   SI(IJ,IJ)=1.D0
      GO TO 306
304   DO 305 IJ=1,N
      DO 305 J=1,N
305   SI(IJ,J)=S(IJ,J)
306   C1=SIG**M
      I2P1=2*I+1
      DO 3060 IJ=1,N
      KKK(IJ)=0.D0
3060  LLL(IJ)=0.D0
      DO 3062 MMM=I2P1,MTP2
      MM=MMM-1
      G=GAM(MM,I)
      FLL=CHEB1(I,MM)*G
      IF(MM.GT.MT)GO TO 3061
      FKK=CHEB2(I,MM)*G
3061  DO 3062 J=1,N
      IF(XX(J).NE.0.D0)GO TO 3063
      X=0.D0
      IF(MM.EQ.2*I)X=1.D0
      GO TO 3064
3063  X=XX(J)**(MM-2*I)
3064  LLL(J)=LLL(J)+FLL*X
      IF(MM.GT.MT)GO TO 3062
      KKK(J)=KKK(J)+FKK*X
3062  CONTINUE
      DO 307 IJ=1,N
      DO 307 J=1,N
      RRR(IJ)=RRR(IJ)+SI(IJ,J)*LLL(J)*C1
      IF((EVEN.EQ..FALSE.).AND.I.EQ.MP1P2)GO TO 307
      PPP(IJ)=PPP(IJ)+SI(IJ,J)*KKK(J)*C1
307   CONTINUE
      HH=HH*H2
303   CONTINUE
      DO 310 IJ=1,N
      U(IJ)=0.D0
      DO 309 K=1,N
309   U(IJ)=U(IJ)+(PPP(K)*FFF(IJ)+RRR(K)*GGG(IJ))*T(IJ,K)
310   U(IJ)=U(IJ)*H2
      CALL EVALO(XX,TK,2)
      CALL EVALO(XX,TL,0)
      DO 3101 J=1,N
3101  U(J)=U(J)+TK(J)-TL(J)
      WRITE(9,7)(U(J),J=1,N)
      DO 3100 IJ=1,N
3100  XX(IJ)=XX(IJ)+XINC(IJ)
311   CONTINUE
1001  CONTINUE
      GO TO 1000
      STOP
      END
```

Evaluation of the solution at the specified points.

```
      REAL FUNCTION GAM*8(M,I)
      REAL*8 GAM,S,SS
      S=1.D0
      SS=1.D0
      IF(M.LE.1)GO TO 2
      DO 1 II=1,M
1     S=S*DFLOAT(II)
2     JJ=M-2*I
      IF(JJ.LE.1)GO TO 4
      DO 3 II=1,JJ
3     SS=SS*DFLOAT(II)
4     GAM=S/SS
      RETURN
      END
```

Construction of γ.

```
      REAL FUNCTION TCHEB*8(M,N)
      REAL*8 TCHEB,SIGN,TO,FUN
      SIGN=1.D0
      DO 1 I=1,N
1     SIGN=-SIGN
      TCHEB=SIGN
      IF(N.EQ.0)RETURN
      DO 2 J=1,N
2     SIGN=-SIGN
      TO=2.D0**(2*M-1)
      TCHEB=TO*(2.D0*FUN(M+N,M-N)-FUN(M+N-1,M-N))*SIGN
      RETURN
      END
```

Definition of the
coefficients of the
shifted Cheby poly.

```
      REAL FUNCTION FUN(P,Q)
      REAL*8 S,SS,SSS,FUN
      INTEGER P,Q
      S=1.D0
      SS=1.D0
      SSS=1.D0
      IF(P.LE.1)GO TO 2
      DO 1 I=2,P
 1    S=S*DFLOAT(I)
 2    IF(Q.LE.1)GO TO 4
      DO 3 I=2,Q
 3    SS=SS*DFLOAT(I)
 4    P=P-Q
      IF(P.LE.1)GO TO 6
      DO 5 I=2,P
 5    SSS=SSS*DFLOAT(I)
 6    FUN=S/(SS*SSS)
      RETURN
      END
```

Routine required by the function TCHEB.

```
      SUBROUTINE EVALQ(X,Q,MM)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 A(16,32),X1(16)
      REAL*8 S(16,16),X(16),SI(16,16),XX(16),Q(16)
      COMMON S,N,H2,M
      IF(MM.LE.1)GO TO 18
      DO 1 I=1,M
 1    Q(I)=0.
      MM2P1=MM/2+1
      HH=1.D0
      SIGN=-1.D0
      DO 150 II=1,MM2P1
      I=II-1
      SIGN=-SIGN
      IF(I-1)2,5,7
 2    DO 4 IJ=1,N
      DO 3 J=1,N
 3    SI(IJ,J)=0.D0
 4    SI(IJ,IJ)=1.D0
      GO TO 10
 5    DO 6 IJ=1,N
      DO 6 J=1,N
 6    SI(IJ,J)=S(IJ,J)
      GO TO 10
 7    DO 8 IJ=1,N
      DO 8 J=1,N
      A(IJ,J+N)=SI(IJ,1)*S(1,J)
      DO 8 K=2,N
 8    A(IJ,J+N)=A(IJ,J+N)+SI(IJ,K)*S(K,J)
      DO 9 IJ=1,N
      DO 9 J=1,N
 9    SI(IJ,J)=A(IJ,J+N)
 10   DO 13 IJ=1,N
      IF(X(IJ).NE.0.D0)GO TO 11
      XX(IJ)=0.D0
      GO TO 13
 11   IF(MM.NE.2*I)GO TO 12
      XX(IJ)=1.D0
      GO TO 13
 12   XX(IJ)=X(IJ)**(MM-2*I)
 13   CONTINUE
      DO 14 IJ=1,N
      X1(IJ)=SI(IJ,1)*XX(1)
      DO 14 J=2,N
 14   X1(IJ)=X1(IJ)+SI(IJ,J)*XX(J)
      C=SIGN*H*2*GAM(MM,I)
      DO 15 J=1,N
 15   Q(J)=Q(J)+C*X(J)
 150  HH=HH*-2
      GO TO 180
 18   DO 19 J=1,N
      IF(X(J).EQ.0.D0)Q(J)=0.D0
      IF(X(J).NE.0.D0)Q(J)=1.D0
 19   CONTINUE
 180  DO 16 I=1,N
      XX(I)=S(I,1)*Q(1)
      DO 16 J=2,N
 16   XX(I)=XX(I)+S(I,J)*Q(J)
      DO 17 J=1,N
 17   Q(J)=H2*XX(J)
      RETURN
      END
```

Subroutine which evaluates the vector MM-th canonical polynomial Q at the points X.