Preconditioning and variational methods for solving linear equations

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:


Metadata Record: https://dspace.lboro.ac.uk/2134/33133

Publisher: © Abdul Wahab bin Josoh

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
IN THE NAME OF ALLAH
THE MERCIFUL THE COMPASSIONATE.
Mengenangi serta menghargai jasa mu,
ayahanda ku, Jusoh
isteriku, Rohani
dan anak-anak ku
Alif Arif Iskandar
dan Halina Sofia.
ACKNOWLEDGEMENT

My earnest thanks and gratitude to my supervisor, Professor D.J. Evans, for his excellent guidance and encouragement. He has demonstrated that given good supervision, research can be undertaken in the most meaningful way.

I wish also to express my thanks to the Universiti Sains Malaysia, for granting me the study leave and for her financial support to enable me to execute this research. I would also like to thank the Commonwealth Scholarship Department for her financial assistance during part of my stay here.

To my wife, I wish to express my profound gratitude for her patience and moral encouragement throughout the duration of this research.

During the period of my research and writing this thesis, I received helpful assistance from many people whom I cannot acknowledge by name. To all of them, I accord my sincere gratitude.

My greatest and ultimate debt and gratitude is due to Allah, the Creator of the heavens and the earth. May He pardon and forgive my failings and weaknesses, strengthen and enliven my faith in Him and endow me with knowledge and wisdom.
# CONTENTS

## Chapter 1: BASIC MATHEMATICAL CONCEPT

1.1 Partial Differential Equations - Basic Concept .............................. 1
1.2 Finite-Difference Approximation to Derivatives .......................... 7
1.3 Mathematical Background in Matrix Theory ............................... 12
1.4 Diagonally Dominance and Positive Definite ............................ 13
1.5 Vector and Matrix Norms .................................................. 15
1.6 Eigenvalues of a Matrix .................................................... 18
1.7 Properties of Real Symmetric Matrices ................................... 20
1.8 Convergence of Sequences of Matrices ................................. 23
1.9 Eigenvalues of a Special Matrix .......................................... 24
1.10 The Power Method ......................................................... 25

## Chapter 2: ELLIPTIC PROBLEMS IN ONE-DIMENSION, TWO-DIMENSIONS AND THREE-DIMENSIONS

2.1 Elliptic Problems
   - One-Dimensional Case .................................................. 27
   - Two-Dimensional Case .................................................. 29
   - Three-Dimensional Case ............................................... 31
2.2 The Model Problems
   - Two-Dimensional Laplace Model Problem ................................ 33
   - Three-Dimensional Laplace Model Problem ............................ 35
   - Biharmonic One-Dimensional Problem ................................ 37
   - Biharmonic Two-Dimensional Problem ................................ 40

## Chapter 3: THE NUMERICAL SOLUTION OF LINEAR EQUATIONS AND ILL-CONDITIONED SYSTEMS

3.1 Introduction ........................................................................ 47
3.2 Direct Methods
   - Gauss Elimination Process ............................................. 49
   - Gauss-Jordan Elimination .............................................. 51
   - LU Decomposition .......................................................... 53
   - Equivalence of Gaussian Elimination with LU Decomposition .... 57
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doolittle's Method with Row Interchanges</td>
<td>59</td>
</tr>
<tr>
<td>The Crout Form of Factorization for Symmetric Matrices</td>
<td>61</td>
</tr>
<tr>
<td>Iterative Methods</td>
<td>65</td>
</tr>
<tr>
<td>Basic Iterative Methods</td>
<td>65</td>
</tr>
<tr>
<td>Convergence of Point Iterative Method</td>
<td>67</td>
</tr>
<tr>
<td>Descent Methods</td>
<td>76</td>
</tr>
<tr>
<td>Conjugate Direction</td>
<td>79</td>
</tr>
<tr>
<td>Conjugate Gradient</td>
<td>82</td>
</tr>
<tr>
<td>Chapter 4: GENERAL THEORY OF PRECONDITIONING</td>
<td></td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>85</td>
</tr>
<tr>
<td>4.2 Errors in Computation</td>
<td>86</td>
</tr>
<tr>
<td>4.3 Measures of Condition of a Matrix</td>
<td>93</td>
</tr>
<tr>
<td>4.4 Application to Direct Methods of Solution</td>
<td>96</td>
</tr>
<tr>
<td>4.5 Application to Iterative Methods of Solution</td>
<td>104</td>
</tr>
<tr>
<td>a. Splitting of A</td>
<td>105</td>
</tr>
<tr>
<td>b. Approximate Factorisation of A</td>
<td>110</td>
</tr>
<tr>
<td>c. Preconditioning by Incomplete Factorisation</td>
<td>111</td>
</tr>
<tr>
<td>- Numerical Example</td>
<td>119</td>
</tr>
<tr>
<td>Chapter 5: ANALYSIS OF VARIATIONAL METHODS</td>
<td></td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>123</td>
</tr>
<tr>
<td>- Variational of a Functional and Extremal Curves</td>
<td>124</td>
</tr>
<tr>
<td>- Energy Method</td>
<td>130</td>
</tr>
<tr>
<td>5.2 The Ritz Method</td>
<td>132</td>
</tr>
<tr>
<td>5.3 Methods for Solving the Ritz System</td>
<td>137</td>
</tr>
<tr>
<td>5.4 The Bubnov-Galerkin Method</td>
<td>139</td>
</tr>
<tr>
<td>5.5 The Trefftz Method</td>
<td>143</td>
</tr>
<tr>
<td>5.6 The Kantorovich Method</td>
<td>147</td>
</tr>
<tr>
<td>5.7 The Method of Weighted Residuals</td>
<td>149</td>
</tr>
<tr>
<td>- Collocation Method</td>
<td>149</td>
</tr>
<tr>
<td>- Over-Determined Collocation</td>
<td>151</td>
</tr>
<tr>
<td>- Least Squares</td>
<td>153</td>
</tr>
<tr>
<td>- Galerkin (w.p.)</td>
<td>156</td>
</tr>
<tr>
<td>5.8 On the Choice of Basis Functions</td>
<td>157</td>
</tr>
</tbody>
</table>
Chapter 6: THE PRECONDITIONED VARIATIONAL METHOD AND FINITE ELEMENT METHOD

6.1 Introduction 159
   - One-Dimensional Problems 162

6.2 Problem 1 162
   - A Comparison of Different Basis Functions in the Least Squares Method 176

6.3 Problem 2 183
   - A comparison of Different Basis Functions in the Least Squares Method 194
   - Two Dimensional Problem 200

6.4 Problem 3 200
   - Selection of the 'Best' Basis Functions 209

6.5 The Finite Element Method 216

6.6 TWODEPEP Package 226
   - Problem 4 228
   - Problem 5 230
   - Problem 6 232

Chapter 7: APPROXIMATE PRECONDITIONING OF THE MODIFIED CONJUGATE GRADIENT METHOD

7.1 Introduction 237
   - Algorithm 7.A 238
   - Algorithm 7.B 239

7.2 Computational Approach 244
   - Numerical Results 248

7.3 Biharmonic Problem 248
7.4 Laplace Problems 260
7.5 Conclusions 267

References 269

Appendix: PROGRAMS 276
CHAPTER 1

BASIC MATHEMATICAL CONCEPT
1.1 PARTIAL DIFFERENTIAL EQUATIONS - BASIC CONCEPT

Many important scientific and engineering problems fall into the field of partial differential equations. Their numerical treatment involves the rates of change of unknown quantities (derivatives) with respect to two or more independent variables and must be represented by a partial differential equation, (henceforth abbreviated as p.d.e.), by which term we mean an equation involving partial derivatives.

The highest order of any of the partial derivatives in the equation is the order of differential equation.

From the standpoint of mathematical physics, the most important and most thoroughly studied equations are those of second order. In the case of two independent variables, a second-order equation can be written in the following general form:

\[ F(x,y,U, \frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, \frac{\partial^2 U}{\partial x^2}, \frac{\partial^2 U}{\partial x \partial y}, \frac{\partial^2 U}{\partial y^2}) = 0, \]  

(1.1.1)

where \( F \) is a given function of

(i) independent variables \( x \) and \( y \)

(ii) the "unknown" function \( U \)

and (iii) partial derivatives of \( U \) with respect to \( x \) and \( y \).

Equation (1.1.1) is said to be linear if it is linear in the unknown function and all its derivatives. A linear second-order equation with two independent variables has the following general form:

\[ A(x,y)\frac{\partial^2 U}{\partial x^2} + 2B(x,y)\frac{\partial^2 U}{\partial x \partial y} + C(x,y)\frac{\partial^2 U}{\partial y^2} + a(x,y)\frac{\partial U}{\partial x} + b(x,y)\frac{\partial U}{\partial y} + c(x,y)U = f(x,y), \]  

(1.1.2)

where \( A(x,y), B(x,y), C(x,y), \ldots, c(x,y) \) and \( f(x,y) \) are given functions of the variables \( x \) and \( y \).

The exact solution \( U(x,y) \) to the above p.d.e. in some region \( R \) satisfies the equation (1.1.2) at every point in \( R \) and matches the 'boundary conditions' on \( C \).
Equation (1.1.2) is conventionally classified with respect to the sign of the quantity of discriminant \( \Delta = B^2 - 4AC \). Specifically, it is defined to be

- **Elliptic**, if \( \Delta < 0 \)  
  \[(1.1.3)\]
- **Parabolic**, if \( \Delta = 0 \)  
  \[(1.1.4)\]
- **Hyperbolic**, if \( \Delta > 0 \)  
  \[(1.1.5)\]

for all \( x, y \) in the region \( R \) under consideration.

This classification scheme is rather interesting since the values of \( A, B \) and \( C \) depend on the independent variables, thus it is possible for a partial differential equation to change its classification within the different region of the domain for which the problem is defined. For example, the equation,

\[
y \frac{\partial^2 u}{\partial x^2} + 2x \frac{\partial^2 u}{\partial x \partial y} + y \frac{\partial^2 u}{\partial y^2} = 0, \tag{1.1.6}
\]

is elliptic in the region where \( x^2 - y^2 < 0 \), parabolic along the lines \( x^2 - y^2 = 0 \) and hyperbolic in the region where \( x^2 - y^2 > 0 \).

A list of more familiar p.d.e.'s is given below,

\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \begin{cases} g(x,y), & \text{Poisson's equation} \\ 0, & \text{Laplace's equation} \end{cases} \tag{1.1.7}
\]

\[
\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = 0, \quad \text{Diffusion or heat conduction equation} \tag{1.1.9}
\]

\[
\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0, \quad \text{Wave equation} \tag{1.1.10}
\]

Equation (1.1.7) is said to be **inhomogeneous** whereas equation (1.1.8), (1.1.9) and (1.1.10) are said to be **homogeneous**. In general, a linear p.d.e. is said to be homogeneous if each term contains either the dependent variable or one of its derivatives. It is inhomogeneous otherwise.

The two elliptic p.d.e.'s (1.1.7)-(1.1.8) are generally associated with steady-state or equilibrium problems. For example, Laplace's equation describes the velocity potential for the steady flow of incompressible, non-viscous fluid, and is the mathematical expression of the physical law that
the rate at which such fluid enters a given region is equal to the rate at which it leaves it.

Such problems are called boundary value problems, since it is usual to have the function \( U \) (say) specified either explicitly or implicitly (by means of the normal derivative) at all points on/boundary of the region. Boundary value problems frequently occur in applications such as reservoir problems, reactor studies, numerical weather forecasting, etc.

Parabolic and hyperbolic p.d.e.'s in general, result from diffusion, equalization or oscillatory processes and the usual independent variables are time and space.

The simplest parabolic p.d.e. given by equation (1.1.9) governs the flow of heat in a thin homogeneous bar or rod, assuming that the radiation and convection are neglected. The temperature distribution along the bar or rod is usually known at some instant in time. This is termed 'the initial condition'. The boundary condition consists of appropriate end conditions, which are either the temperature given at two ends of the bar or rod or some measure of the diffusion from the ends. Such problems are called 'initial-boundary value problems'. (Sometimes only referred to as 'initial value problems').

The simplest hyperbolic equation (1.1.10) give the transverse displacement at a given distance from one end of a vibrating string of given length after a certain time, the initial condition, usually \( U \) and \( \frac{\partial U}{\partial t} \), are given at some instant in time and the boundary conditions are given on two lines \((x=a,b)\).

For arbitrarily-shaped regions and general boundary conditions, it is usually impossible to determine an exact solution to a given p.d.e. Only in the simplest cases can a solution be analytical either in implicit form or that involving a finite formula.

The approximate methods which have been developed to tackle this problem can be divided into two groups:
(i) **Analytical approximate methods**, in which the approximate solution is expressed in an analytical form. Although these methods can provide useful information about the character of the solution for critical values of the dependent variables however they tend to be more difficult to apply than the numerical methods [Smith, 1964]. They will not be discussed further in this thesis.

(ii) **Numerical approximation method**, in which approximate values of the required solution can be found at various points of the region under consideration in a tabular form. Finite difference methods are approximate in the sense that derivatives at a point are approximated by difference quotients over a small interval, i.e. \( \frac{\partial U}{\partial x} \) is replaced by \( \frac{\delta U}{\delta x} \) where \( \delta x \) is small, but the values of the solution not approximate in the sense of being crude estimates. They are especially suited to solving problems in regular regions.

Finite element methods involve making discrete approximations to the connectivity of the elements of the region itself. In the one-dimensional case this means that a curve (the solution) is approximated by a series of straight-line segments. The methods are particularly useful for dealing with irregular regions. This class of method is relatively recent and rapidly gaining ground not only in solid mechanics but also in fluid mechanics, heat transfer problems and other application areas [Zienkiewicz, O.C., 1971]. However, other methods exist which include the integration and variational formulation methods [Varga, 1962]. We will discuss the variational method in relative detail in Chapter 5.

**Well-Posed Problems**

For any p.d.e. problem describing a stable situation, one would expect that small variations in the data should result in correspondingly small
variations in the solution. Problems which do not have this property cannot, in general, be attacked successfully with numerical methods. The proposition of finding in mathematical terms which problems are acceptable models of the physical world and which are not has led to the concept of a well-posed problem (Hadamard, 1932).

Definition 1.1

A p.d.e. is well posed in the sense of Hadamard if and only if its solution exists, is unique, and dependent continuously on the data prescribed.

This definition shall be clarified by the following example. The elliptic equation (1.1.8) associated with initial value (Cauchy) data

\[
\begin{align*}
U(x,0) &= 0 ; \quad \frac{\partial U}{\partial y}(x,0) = \frac{1}{k} \sin (kx) \quad (1.1.11) \\
\text{has the solution,} \quad U(x,y) &= \frac{1}{k} \sin(kx) \sinh(ky) \quad (1.1.12)
\end{align*}
\]

As \( k \to \infty \), (1.1.11) approaches zero uniformly, whereas for \( y > 0 \) the solution (1.1.12) oscillates between limits that increase indefinitely. This problem is not well posed by the preceding definition. The appropriate way is to formulate elliptic equations as boundary-value problems.

Boundary Value Problems

We consider the elliptic problem (1.1.8) in a connected region \( R \) in the \( XOY \) plane. Let \( \bar{R} = R + C \) be the closure of the considered region \( R \) with the boundary \( C \).

a) The First Boundary Value Problem (also called the Dirichlet problem) when \( g(x,y) = 0 \).

The solution \( U \) is to satisfy (1.1.8) in \( R \) and take on prescribed values,

\[
U = U_B(x,y) \quad (1.1.13)
\]

on the boundary \( C \) of that region.

b) The Second Boundary Value Problem (often called the Neumann problem)

The solution is to possess prescribed normal derivatives,
\[ \frac{\partial U}{\partial n} = U_B', \]  
\[ (1.1.14) \]
on the surface C boundary R.

c) The Third Boundary Value Problem (mixed or Robin's problem)

Boundary conditions of mixed form,

\[ \frac{\partial U}{\partial n} + HU = G \]
on C are called conditions of the third kind.

It is often the case that an elliptic problem is specified by boundary conditions that are of different kind along different parts of C.
1.2 FINITE DIFFERENCE APPROXIMATION TO DERIVATIVES

In order to illustrate how difference methods are used to solve p.d.e.'s we consider without loss of generality, that the problem to solve is the elliptic equation (1.1.8) with independent variables x,y in a connected region R, in the (X O Y) plane. Let \( \overline{R}=\mathbb{R}+C \) be the closure of the considered region R with boundary C. We overlay \( \overline{R} \) with a system of rectangular mesh lines formed by two families of equally spaced straight lines, which are parallel to OX and OY. If we choose any convenient point \((x_0, y_0)\) in the region as the "mesh coordinates" i,j, then,

\[
x = x_0 + ih, \quad i=0, \pm 1, \pm 2, \ldots
\]

\[
y = y_0 + jh, \quad j=0, \pm 1, \pm 2, \ldots
\]

The intersection of these points are referred to as the (unknown) mesh points. Other names used in the literature are grid, lattice, nodes and pivots. The distances between the parallel lines are called mesh sizes (lengths).

A mesh point \(p_{i,j}\) is called 'regular' if the four adjacent points \(p_{i+1,j}, p_{i-1,j}, p_{i,j+1}, p_{i,j-1}\) are also mesh points contained in \(R\); otherwise \(p_{i,j}\) is called 'irregular' (see Figure 1.2.1).

![Figure 1.2.1](image-url)
The p.d.e. is approximated at each of \( n^2 \) mesh points, \( p_{1,1}, p_{1,2}, \ldots, p_{1,j}, \ldots \) and the problem is now reduced to solving a set of \( n^2 \) algebraic equations (linear, if the differential equations are linear) involving approximate values of \( U \) at the \( n \)-mesh points in \( R \).

We first give the important theorems (without proof) which are basic in the derivation of finite difference approximation.

**Theorem 1.2.1 (Taylor's Theorem)**

Let \( U(x) \) have \( s+1 \) continuous derivatives on \( a \leq x \leq b \) for some \( s \geq 0 \) and let \( x, x_0 \in [a, b] \). Then,

\[
P_s(x) = U(x_0) + \frac{1}{1!} (x-x_0) \frac{dU(x_0)}{dx} + \ldots + \frac{1}{s!} \frac{d^s U(x_0)}{dx^s} + R_{s+1}(x)
\]

for some \( \xi \) between \( x_0 \) and \( x \).

**Theorem 1.2.2 (Taylor's Theorem in two dimensions)**

Let \( (x_0, y_0) \) and \( (x_0 + \xi, y_0 + \eta) \) be two given points and assume \( U(x,y) \) is \( (s+1) \) times continuously differentiable for all \( (x,y) \) in some neighbourhood of \( P(x_0, y_0; x_0 + \xi, y_0 + \eta) \). Then,

\[
U(x_0 + \xi, y_0 + \eta) = U(x_0, y_0) + \sum_{j=1}^{s} \frac{1}{j!} (\xi \frac{\partial^j U}{\partial x^j}(x_0, y_0) + \eta \frac{\partial^j U}{\partial y^j}(x_0, y_0)) x = x_0, y = y_0
\]

\[
+ \frac{1}{(s+1)!} (\xi \frac{\partial^s U}{\partial x^s}(x_0, y_0) + \eta \frac{\partial^s U}{\partial y^s}(x_0, y_0)) x = x_0 + \delta \xi, y = y_0 + \delta \eta
\]

for some \( 0 \leq \delta \leq 1 \). The point \( (x_0 + \delta \xi, y_0 + \delta \eta) \) is an unknown point on the line \( P(x_0, y_0; x_0 + \xi, y_0 + \eta) \).

By using the above theorems we can now write,

\[
U(x_0 + \delta \xi, y_0 + \delta \eta) = U(x_0, y_0) \frac{\partial U}{\partial x}(x_0, y_0) + \frac{1}{2} \delta \frac{\partial^2 U}{\partial x^2}(x_0, y_0) + \frac{1}{2} \delta \frac{\partial^2 U}{\partial y^2}(x_0, y_0) + \delta \frac{\partial^2 U}{\partial x \partial y}(x_0, y_0)
\]

for \( 0 \leq \delta \leq 1 \).
\[ U(x, y^{+h}) = U(x, y) + h \left( \frac{\partial U}{\partial x} \right)_{x,y} + \frac{h^2}{2!} \left( \frac{\partial^2 U}{\partial x^2} \right)_{x,y} + o(h^2) \] (1.2.6)

\[ o(\epsilon) \leq 1. \]

The value of \( \frac{\partial U}{\partial x} \) at the mesh point \((x, y)\) may be calculated by a number of different finite difference approximations. Provided \( h \) is small enough, one way is to rewrite (1.2.4) as,

\[
\frac{U(x+h, y) - U(x, y)}{h} = \frac{\partial U}{\partial x} \left( x,y \right) + \frac{h \left( \frac{\partial^2 U}{\partial x^2} \right)_{x,y}}{2!} + o(h) \]

(1.2.7)

or

\[
\frac{\partial U}{\partial x} \left( x,y \right) = \frac{U(x+h, y) - U(x, y)}{h} + o(h), \text{ as } h \rightarrow 0. \]

(1.2.8)

This is called \textit{forward difference approximation} to the \( U \) function with respect to \( x \) variable at the mesh point \((x, y)\). Where there is no ambiguity the subscript \( x,y \) on \( \frac{\partial U}{\partial x} \) will be dropped.

Similarly for the \textit{backward difference approximation}, we have,

\[
\frac{\partial U}{\partial x} \left( x,y \right) = \frac{U(x, y) - U(x-h, y)}{h} + o(h), \text{ as } h \rightarrow 0. \]

If \( U(x+h, y) \) and \( U(x-h, y) \) are expanded about \( U(x, y) \) up to the third power of \( h \), i.e.,

\[
U(x^{+h}, y) = U(x, y) + h \left( \frac{\partial U}{\partial x} \right)_{x,y} + \frac{h^2}{2!} \left( \frac{\partial^2 U}{\partial x^2} \right)_{x,y} + \frac{h^3}{3!} \left( \frac{\partial^3 U}{\partial x^3} \right)_{x,y} + o(h^3) \]

(1.2.9)

\[ o(\epsilon) \leq 1, \text{ then, } \]

\[
\frac{\partial U}{\partial x} \left( x,y \right) = \frac{U(x+h, y) - U(x-h, y)}{2h} + o(h^2), \]

(1.2.10)

and is called the \textit{central difference approximation}. It can be seen that the error for (1.2.10) is much smaller than the first two types of approximations.

From (1.2.9) we can approximate the second order derivative from

\[
\frac{\partial^2 U}{\partial x^2} = \frac{U(x+h, y) - 2U(x, y) + U(x-h, y)}{h^2} + o(h^2), \]

(1.2.11)

Similar to (1.2.5) and (1.2.6) after expanding \( U(x, y^{+h}) \) and \( U(x, y^{-h}) \) about \( U(x, y) \) we obtain,

\[
\frac{\partial U}{\partial y} = \frac{U(x, y^{+h}) - U(x, y^{-h})}{2h} + o(h^2), \quad \text{(1.2.12)}
\]

and

\[
\frac{\partial^2 U}{\partial y^2} = \frac{U(x, y^{+h}) - 2U(x, y) + U(x, y^{-h})}{h^2} + o(h^2) \quad \text{(1.2.13)}
\]
Let us denote $U(x,y)$ by $U_{i,j}$ for a general mesh point $(x,y)=(ih,jh)$. Then, Poisson's equation,

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = g(x,y),$$

using formula (1.1.1), (1.1.8) and (1.2.13), can be replaced at the point $(x_i, y_j)$ by,

$$\frac{1}{h^2} \left( U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j} \right) = g_{i,j} + \frac{h^2}{12} \left( \frac{\partial^4 U}{\partial x^4} + \frac{\partial^4 U}{\partial y^4} \right)_{i,j} + ...$$

or equivalently,

$$4U_{i,j} - U_{i+1,j} - U_{i-1,j} - U_{i,j+1} - U_{i,j-1} = -h^2 g_{i,j} - \frac{h^4}{12} \left( \frac{\partial^4 U}{\partial x^4} + \frac{\partial^4 U}{\partial y^4} \right)_{i,j} - ...$$

The quantity,

$$\frac{h^4}{12} \left( \frac{\partial^4 U}{\partial x^4} + \frac{\partial^4 U}{\partial y^4} \right)_{i,j} + ...$$

is defined as the local truncation error (henceforth abbreviated as L.T.E.) of the formula (1.2.11) and the principal part of this L.T.E. is,

$$\frac{h^4}{12} \left( \frac{\partial^4 U}{\partial x^4} + \frac{\partial^4 U}{\partial y^4} \right)_{i,j}$$

Note that the quantity $O(h)$ (for example equation 1.2.8) has been used as an asymptotic notation for the truncation error.

If we scan over the mesh points with formula (1.2.11) omitting the local truncation error, we obtain a set of simultaneous equations whose solution $\{u_{i,j}\}$ is a finite-difference approximation to the exact solution $\{U_{i,j}\}$ at the internal mesh points.

In the matrix notation these equations can be written as,

$$Au = b.$$  \hspace{1cm} (1.2.17)

In general, if there are $n^2$-internal mesh points, then $u$ and $b$ are $(n^2 \times 1)$ and $A$ is a matrix of order $n^2$ (see Section 1.3). From (1.2.15) it can be easily seen that better accuracy is obtained as the mesh-size $h$ tends to zero. However, a decrease in $h$ means an increase in the number of simultaneous equations to be solved - implying a large computer storage is required.
Fortunately, the matrix $A$ is usually *sparse* (many of its elements are zero) and only non-zero elements are retained in the computer. We shall see later that we can work with the generated rather than stored form of the matrix in the computer (Chapter 7).
1.3 MATHEMATICAL BACKGROUND IN MATRIX THEORY

Notation

A 

square matrix of order \( n \)

\( a_{i,j} \) 

scalar, which is the element in the \( i \)th row and \( j \)th column of matrix \( A \)

\( A^{-1} \) 
inverse of \( A \)

\( A^T \) 
transpose of \( A \)

\( |A| \) 
determinant of \( A \)

\( I \) 
unit matrix of order \( n \)

\( 0 \) 
null matrix

\( X \) 
column vector with elements \( x_i \) (\( i=1,2,\ldots,n \))

\( X^T \) 
row vector with elements \( x_i \) (\( i=1,2,\ldots,n \))

\( ||A|| \) 
norm of \( A \)

\( ||x|| \) 
norm of \( x \)

Definitions

The matrix \( A \) is

Non-singular if \( |A| \neq 0 \)

symmetric if \( A = A^T \)

orthogonal if \( A^{-1} = A^T \)

null if \( a_{i,j} = 0 \) (\( i,j=1,2,\ldots,n \))

diagonal if \( a_{i,j} = 0 \) (\( i \neq j \))

tridiagonal if \( a_{i,j} = 0 \) for \( |i-j| > 1 \)

upper triangular if \( a_{i,j} = 0 \), \( i > j \)

lower triangular if \( a_{i,j} = 0 \), \( j > i \).
1.4 DIAGONALLY DOMINANCE AND POSITIVE DEFINITE

Definition 1.4.1

An $n \times n$ matrix $A$ is **diagonally dominant** if

$$|a_{i,i}| > \sum_{j=1, j \neq i}^{n} |a_{i,j}| \quad \text{for all } 1 \leq i \leq n$$  \hspace{1cm} (1.4.1)

The matrix $A$ is said to be **strictly diagonally dominant** if strict inequality is satisfied for all $(i, j) \in \mathbb{N}$ in (1.4.1).

Definition 1.4.2

A permutation matrix $P = (p_{i,j})$ is a matrix with exactly one non-zero element, namely unity in each row and each column. Thus, for example,

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$ \hspace{1cm} (1.4.2)

is a permutation matrix of order 4.

For any permutation matrix $P$ we have,

$$P^T P = P P^T = I,$$ \hspace{1cm} (1.4.3)

$$P^T = P^{-1}.$$ \hspace{1cm} (1.4.4)

Definition 1.4.3

For $n \geq 2$, an $n \times n$ matrix $A$ is **reducible** if there exists an $n \times n$ permutation matrix $P$ such that,

$$P A P^T = \begin{bmatrix} A_{1,1} & A_{1,2} \\ 0 & A_{2,2} \end{bmatrix}$$ \hspace{1cm} (1.4.5)

where $A_{1,1}$ is an $r \times r$ submatrix and $A_{2,2}$ is an $(n-r) \times (n-r)$ submatrix where $1 \leq r \leq n$. If no such matrix exists, then $A$ is **irreducible**.

Definition 1.4.4

If $A$ is real and $x$ is complex, then $A$ is **positive definite** if

$$(x, Ax) > 0 \quad \text{for all } x \neq 0.$$  \hspace{1cm} (1.4.6)

N.B. 1) If $x, y$ are complex, then $(x, y) = \sum_{i=1}^{n} x_i \overline{y}_i$, where $\overline{y}_i$ is the complex conjugate of $y_i$.)
2) If the inequality in (1.4.6) is not strict inequality, then $A$ is called non-negative definite.

Next, we state an important theorem (without proof) which is sometimes used as a definition of positive (non-negative) definiteness.

**Theorem 1.4.1**

A real matrix is positive (non-negative) definite if and only if it is symmetric and all its eigenvalues are positive (non-negative, if at least one eigenvalue is equal to zero).

If $A$ is positive definite, it can be written as $A = G J G^{-1}$ where $J$ is a positive diagonal matrix. Also, $G$ can be taken to be an orthogonal matrix, (i.e. $G^T = G^{-1}$) [Young (1971)]. If $J^\frac{1}{2}$ denotes the diagonal matrix whose elements are the positive square roots of the element of $J$ then $A^\frac{1}{2} = G J^\frac{1}{2} G^{-1}$ is positive definite by Theorem 1.4.1. It should be noted that $(A^\frac{1}{2})^2 = (G J^\frac{1}{2} G^{-1})^2 = A$.

**Theorem 1.4.2**

A real symmetric matrix $A$ is positive (non-negative) definite if and only if it can be written in the form $A = P^T P$ where $P$ is some non-singular (singular) matrix.

**Proof**

(i) Assume $A$ is positive definite (and real). We know $A = A^\frac{1}{2} A^\frac{1}{2}$ where $A^\frac{1}{2}$ is defined above. Since $A^\frac{1}{2}$ is symmetric, $A = (A^\frac{1}{2})^T (A^\frac{1}{2})$. Since $A^\frac{1}{2}$ is also positive definite, $\det A^\frac{1}{2} \neq 0$. Thus putting $P = A^\frac{1}{2}$ gives the required condition.

(ii) Assume $A = P^T P (\det P \neq 0)$. Then for any vector $x \neq 0$

$$x^T A x = x^T P^T P x = (P x)^T P x > 0.$$  

Hence, $A$ is positive definite by Definition 1.4.4.

The proof for the case of $A$ non-negative definite, follows in a similar manner.
1.5 VECTOR AND MATRIX NORMS

In the discussion that follows, we need some meaningful way to measure
the size of a vector or the distance between two vectors. This measure is
called a norm.

A norm on $\mathbb{R}^n$ is a real-valued function $\| \cdot \|$ defined on $\mathbb{R}^n$ and
satisfying the three conditions below:

(i) $\| x \| \geq 0$, and $\| x \| = 0$, if and only if $x = 0$,

(ii) $\| ax \| = |a| \| x \|$, for all scalars $a$ and vectors $x$,

(iii) $\| x + y \| \leq \| x \| + \| y \|$, for all vectors $x$ and $y$.

Three useful examples of norms are the so-called $\ell_p$ norms, $\| \cdot \|_p$, for $\mathbb{R}^n$,
p=1,2,\infty:

\[
\| x \|_1 = |x_1| + |x_2| + \ldots + |x_n|,
\]

\[
\| x \|_2 = \sqrt{|x_1|^2 + |x_2|^2 + \ldots + |x_n|^2},
\]

\[
\| x \|_\infty = \max\{|x_1|, |x_2|, \ldots, |x_n|\}.
\]

($\| x \|_2$ is often called the length of $x$).

We define $\| x_c - x_t \|$ to be the absolute error and define the quantity
$\| x_c - x_t \| / \| x_t \|$ to be the relative error. The vectors $x_c$ and $x_t$ are the
computed solution and the "true" solution respectively.

We measure the corresponding size of a matrix $A$ by

\[
\| A \| = \max_{\| x \| \neq 0} \frac{\| Ax \|}{\| x \|},
\]

which is equivalent to,

\[
\| A \| = \max_{\| x \| = 1} \frac{\| Ax \|}{\| x \|}.
\]

It can be shown that this maximum exists for every matrix $A$ (and for any
choice of the vector norm). [S.D. Conte, 1980].

Definition 1.5.1

A matrix norm constructed by means of (1.5.4) is said to be subordinate
to the corresponding vector norm.
Definition 1.5.2

A matrix norm is said to be compatible with a vector norm $||\mathbf{x}||$ if

$$||\mathbf{Ax}|| \leq ||A|| ||\mathbf{x}||.$$  \hfill (1.5.6)

The following properties can be shown to hold for the matrix norm (1.5.4).

(i) $||A|| \geq 0$, and $||A||=0$, if and only if $A=0$  \hfill (1.5.7)

(ii) $||aA|| = |a| ||A||$, for all $A$ and all scalars $a$.  \hfill (1.5.8)

(iii) $||A+B|| \leq ||A||+||B||$, for all $A$ and $B$.  \hfill (1.5.9)

In addition,

(iv) $||AB|| \leq ||A|| ||B||$.  \hfill (1.5.10)

If $A$ is invertible (i.e. det $A \neq 0$), then,

$$\frac{||\mathbf{x}||}{||A||} \leq ||\mathbf{Ax}|| \leq ||A|| ||\mathbf{x}||$$  \hfill (1.5.11)

and both inequalities are sharp, i.e, each can be made an equality by an appropriate choice of a (non-zero) $\mathbf{x}$.

Definition 1.5.3

Let $A$ be an $(n \times n)$ matrix with eigenvalues $\lambda_i$, $1 \leq i \leq n$, then,

$$\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|,$$  \hfill (1.5.12)

is the spectral radius of $A$.

For any $\lambda$ the following inequality is satisfied:

$$|\lambda| ||\mathbf{x}|| = ||\lambda \mathbf{x}|| = ||\mathbf{Ax}|| \leq ||A|| ||\mathbf{x}||$$  \hfill (1.5.13)

which indicate that,

$$||A|| \geq \rho(A),$$  \hfill (1.5.14)

for any $A$ and any norm.

Definition 1.5.4

The subordinate norms associated with the $l_p$ norms are:-

$$||A||_1 = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{i,j}|$$ (maximum absolute column sum) \hfill (1.5.15)

$$||A||_2 = (\rho(A^T A))^\frac{1}{2}$$ (spectrum norm) \hfill (1.5.16)
If $A$ is symmetric, then

$$||A||_2 = \rho(A).$$

(1.5.18)

Proof:

Since $A$ is symmetric,

$$||A||_2^2 = \rho(A^TA) = \rho(A^2) = \rho^2(A),$$

proving that $||A||_2 = \rho(A)$.

For a more detailed analysis of norms and their corresponding proofs see for example, Faddeev and Faddeeva (1963).
1.6 EIGENVALUES OF A MATRIX

The eigenvalues of A are the roots $\lambda_i \ (i=1,2,\ldots,n)$ of the characteristic

\[ p_A(\lambda) = \det(A-\lambda I) = 0 , \]

where $p_A(\lambda)$ is a polynomial of degree n.

An eigenvector $x^{(i)}$ for each $\lambda_i$ is given by,

\[ Ax^{(i)} = \lambda_i x^{(i)} \quad (x^{(i)} \neq 0) . \]  \hspace{1cm} (1.6.1)

Two matrices A and B are similar if there is a non-singular matrix P

such that,

\[ B = P^{-1}AP \]  \hspace{1cm} (1.6.2)

$P^{-1}AP$ is a similarity transformation of A.

Similar matrices have important properties which are summarized in the

definitions below.

**Theorem 1.6.1**

If two matrices A and B satisfy (1.6.2) then A and B have the same
eigenvalues. If $x$ is any eigenvector of A then B has the corresponding
eigenvector $y=P^{-1}x$.

**Proof:**

Let $\lambda$ be any eigenvalue of A and $x$ be the corresponding eigenvector.

Then,

\[ Ax = \lambda x , \]

but

\[ B = P^{-1}AP. \]

Thus, if $y = P^{-1}x$, we obtain,

\[ By = P^{-1}AP(P^{-1}x) = P^{-1}Ax = P^{-1}(\lambda x) \]

\[ = \lambda P^{-1}x = \lambda y . \]

Hence $\lambda$ is also an eigenvalue of B and $y$ is the corresponding eigenvector.

We say that the eigenvalues of a matrix are preserved under similarity

transformation.

This theorem reduces the problem of finding the eigenvalues of a given

matrix A to that of finding the eigenvalues of another matrix $B=P^{-1}AP$. A
suitable choice of \( P \) ensures that \( B \) has a simpler structure than \( A \).

A square matrix \( A \) is said to be Hermitian if \( A^T = A \) (where \( A \) denotes the matrix whose elements are the complex conjugates of those of \( A \)).

For real matrices, the term Hermitian may be replaced by symmetric, i.e. \( A^T = A \).

**Definition 1.6.1**

A square matrix \( A \) is said to be Hermitian if \( \overline{A^T} = A \) (where \( \overline{A} \) denotes the matrix whose elements are the complex conjugates of those of \( A \)).

For real matrices, the term Hermitian may be replaced by symmetric; i.e. \( A^T = A \).

**Theorem 1.6.1**

A matrix \( A \) is positive definite (non-negative definite) if and only if it is Hermitian and all its eigenvalues are positive (non-negative).

**Proof:** See Young (1971).

**Theorem 1.6.2** (Gerschgorin)

Let \( A \) have \( n \) eigenvalues \( \lambda_i, i=1,2,...,n \). Then each \( \lambda_i \) lies in the minor of the \( n \) discs,

\[
|z-a_{i,i}| \leq \sum_{j=1}^{n} |a_{i,j}|, \quad j \neq i
\]

in the \( z \)-plane.

From the Gerschgorin Theorem we obtain,

\[
\rho(A) \leq \min \left( \max_i \sum_j |a_{i,j}|, \max_j \sum_i |a_{i,j}| \right).
\]

**Proof follows from** (1.5.14), (1.5.15) and (1.5.17).
1.7 PROPERTIES OF REAL SYMMETRIC MATRICES

Real symmetric matrices occur frequently in our work. It is therefore appropriate to list some of their important properties.

Theorem 1.7.1

The eigenvalues of a real symmetric matrix are all real.

Proof:

Let \( \lambda \) be an eigenvalue and \( \mathbf{x} \) be an eigenvector of a symmetric matrix \( \mathbf{A} \).

Then,
\[
\mathbf{A}\mathbf{x} = \lambda \mathbf{x},
\]
and so
\[
\mathbf{x}^T\mathbf{A}\mathbf{x} = \lambda \mathbf{x}^T\mathbf{x}.
\]
This gives,
\[
\mathbf{x}^T\mathbf{A}\mathbf{x} = (x_1^2, x_2^2, \ldots, x_n^2) = \sum_{i=1}^{n} x_i^2,
\]
which is real and positive.

Taking the complex conjugate of (1.7.1) transposing and post-multiplying by \( \mathbf{x} \) gives,
\[
(\mathbf{x}^T\mathbf{A}^T)\mathbf{x} = (\lambda\mathbf{x})^T\mathbf{x},
\]
But \( \mathbf{A}^T = \mathbf{A} \) (A real and symmetric)
so
\[
\mathbf{x}^T\mathbf{A}\mathbf{x} = \lambda \mathbf{x}^T\mathbf{x},
\]
Comparing (1.7.1) and (1.7.3) shows that,
\[
\lambda \mathbf{x}^T\mathbf{x} = \lambda \mathbf{x}^T\mathbf{x},
\]
Since \( \mathbf{x}^T\mathbf{x} \neq 0 \) we obtain immediately \( \lambda = \bar{\lambda} \).

Since all the eigenvalues of a symmetric matrix are real it follows immediately that the eigenvectors are also real.

Theorem 1.7.2 (Orthogonality property)

If \( \mathbf{x}_1 \) and \( \mathbf{x}_2 \) are eigenvectors of a real symmetric matrix \( \mathbf{A} \), corresponding to distinct eigenvalues \( \lambda_1 \) and \( \lambda_2 \), then \( \mathbf{x}_1^T\mathbf{x}_2 = \mathbf{x}_2^T\mathbf{x}_1 = 0 \).

Proof:
\[
\mathbf{A}\mathbf{x}_1 = \lambda_1 \mathbf{x}_1 \quad \text{and} \quad \mathbf{A}\mathbf{x}_2 = \lambda_2 \mathbf{x}_2,
\]
so
\[
\mathbf{x}_1^T\mathbf{A} = \lambda_1 \mathbf{x}_1^T
\]
Thus,
\[
(\mathbf{x}_1^T\mathbf{A})\mathbf{x}_2 = \lambda_1 \mathbf{x}_1^T\mathbf{x}_2 = \mathbf{x}_1^T(\mathbf{A}\mathbf{x}_2) = \lambda_2 \mathbf{x}_1^T\mathbf{x}_2 = 0,
\]
and so
\((\lambda_1 - \lambda_2)x_1^T x_2 \neq 0\).
Since \(\lambda_1 \neq \lambda_2\) we must have \(x_1^T x_2 = 0\). (i.e. \(x_1 \cdot x_2 = 0\)).

**Theorem 1.7.3**

If \(A\) is symmetric, diagonally dominant matrix with positive diagonal elements, it is positive definite.

**Proof:**

Since \(A\) is symmetric, the eigenvalues of \(A\) are real (Theorem 1.6.1).

By application of Gerschgorin's theorem, the eigenvalues of \(A\) are all positive since \(A\) is diagonally dominant with positive diagonal elements. Therefore by Definition 1.1.3, \(A\) is positive definite.

Each real symmetric matrix \(A\) can be associated with a *quadratic form*

\[ \phi(x_1, x_2, \ldots, x_n) = x^T Ax. \]

So,

\[ \phi = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j \]  \hspace{1cm} (1.7.4)

**Definition 1.7.1**

A real symmetric matrix is positive definite if its quadratic form is positive for all \(x \neq 0\) i.e.

\[ x^T Ax > 0, \quad \forall x \neq 0. \]

See [Jennings, 1977].

**Theorem 1.7.4**

Let \(\lambda\) be an eigenvalue of \(A\) with eigenvector \(x\). Then:

1. \(a\lambda\) is an eigenvalue of \(aA\) with eigenvector \(x\).
2. \(\lambda - \mu\) is an eigenvalue of \(A - \mu I\) with eigenvector \(x\).
3. if \(A\) is non-singular, then \(\lambda \neq 0\) and \(\lambda^{-1}\) is an eigenvalue of \(A^{-1}\) with eigenvector \(x\).

**Proof:**

The equation \(Ax = \lambda x\) implies that \(A x = a \lambda x\) and \((A - \mu I)x = (\lambda - \mu)x\). This indicates part 1 and 2. For part 3, note that \(\lambda = 0\) implies \(Ax = \lambda x = 0\).
Hence, the homogeneous equation $Ax = 0$ has a non-trivial solution and $A$ is singular. Since $A$ is assumed non-singular we must have $\lambda \neq 0$. Then $Ax = \lambda x$ implies that $A^{-1}x = \lambda^{-1}x$ (G.W. Stewart (1973)).
1.8 CONVERGENCE OF SEQUENCES OF MATRICES

The matrix $A$ is convergent to zero if the sequence of matrices $A, A^2, A^3, \ldots$ converges to the null matrix $0$.

Theorem 1.8.1 gives the sufficient condition for a matrix $A$ to converge whereas Theorem 1.8.2 gives the necessary and sufficient condition.

**Theorem 1.8.1**

$$\lim_{r \to \infty} A^r = 0 \text{ if } ||A|| < 1$$

**Proof:**

$$||A^r|| = ||AA^{r-1}||$$
$$\leq ||A|| ||A^{r-1}||$$
$$\leq ||A||^2 ||A^{r-2}||$$
$$\vdots$$
$$\leq ||A||^r$$

and hence the result follows.

**Theorem 1.8.2**

If $A$ is an arbitrary square matrix, then $A$ is convergent if and only if $\rho(A) < 1$.

**Proof:**

Consider the Jordan canonical form of $A$. A Jordan submatrix of $A$ is of the form,

$$J_1 = \begin{pmatrix}
\lambda_1 & 1 & 0 \\
 & \lambda_1 & 1 \\
 & 0 & \lambda_1
\end{pmatrix}$$

where $\lambda_1$ is an eigenvalue of $A$. If the Jordan submatrix is raised to the power $r$, then the result tends to the null matrix as $r \to \infty$, if and only if $|\lambda_1| < 1$. (This proof is given in greater detail in Varga (1962) p.13-15).
### 1.9 THE EIGENVALUES OF A SPECIAL MATRIX

The eigenvalues of the $n \times n$ matrix,

$$
\begin{pmatrix}
  a & b \\
  c & a & b \\
  & c & a & b \\
  & & 0 & b \\
  & & & 0 & c & a
\end{pmatrix}
$$

are

$$
\lambda_i = a + 2 \sqrt{(bc)^2} \cos \left( \frac{i\pi}{n+1} \right), \quad i=1,2,\ldots,n.
$$

(1.9.1)

where, $a, b$ and $c$ may be real or complex. A proof is given in (Smith G.D., 1978, p.113).
1.10 THE POWER METHOD

This is an iterative method, proceeding via a sequence of successive approximations. In its basic form it yields the eigenvalue of the largest modulus together with the corresponding eigenvector. In our work we require just the largest and the smallest eigenvalue (see Chapter 3), which make the method ideal to be used.

Theory of the Method

Our basic assumptions are that $A$ is a real $(n \times n)$ matrix, with $n$ real, distinct eigenvalues $\lambda_i$, $i=1,2,\ldots,n$. $A$ thus has a full set of linearly independent eigenvectors $v_i$, $i=1,2,\ldots,n$. These form the basis for the space, $\mathbb{R}^n$. So any vector $x$ can be written as,

$$x = \sum_{i=1}^{n} \alpha_i v_i,$$

We shall also assume that the eigenvalues are ordered so that,

$$|\lambda_1| > |\lambda_2| > |\lambda_3| \geq \ldots \geq |\lambda_n|$$

For each $i$,

$$Av_i = \lambda_i v_i$$

so that,

$$Ax = A\left(\sum_{i=1}^{n} \alpha_i v_i\right)$$

$$= \alpha_1 \lambda_1 v_1 + \alpha_2 \lambda_2 v_2 + \ldots + \alpha_n \lambda_n v_n.$$

More generally, for any positive integer $k$,

$$A^k x = \alpha_1 \lambda_1^k v_1 + \alpha_2 \lambda_2^k v_2 + \ldots + \alpha_n \lambda_n^k v_n.$$

If, we now assume that $\alpha_1 \neq 0$, we obtain $|\lambda_1^k| >> |\lambda_i^k|$ for all $i>2$ and all sufficiently large $k$.

In (1.10.4) $\alpha_1 \lambda_1^k v_1$ will be dominant, and so $A^k x$ provides a good approximation for a multiple of $v_1$.

The disadvantage is that if $|\lambda_1|$ is large then after a few iterations the elements of $A^k x$ will be very large, and overflow may soon occur.

In order to keep the elements of $A^k x$ within reasonable bounds during computation to prevent overflow, it is usual to normalise the vector at each
iteration by dividing all its elements by the element of largest modulus. Thus (1.10.4) after \( k \) iterations becomes,

\[
\frac{1}{\lambda_1^k} A^k x = \alpha_1 v_1 + \sum_{i=2}^{n} \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^k v_i.
\] (1.10.5)

The rate of convergence of the Power method is now clearly proportional to the rate at which \( \frac{\lambda_2}{\lambda_1} \) \( \to 0 \).

Remarks

1) Convergence will be slow for matrices having two large eigenvalues of nearly equal modulus.

2) Powers of \( A \) are never actually computed in the implementation of the Power method, nor is it necessary to know the \( \alpha_i \)'s and/or the \( v_i \)'s in (1.10.1) as given in the algorithm below.

3) Even if \( A \) does not have \( n \) linearly independent eigenvectors, it is still possible to establish that the power method is convergent provided that \( A \) has a single dominant eigenvalue \( \lambda_1 \). [Johnson & Riess, 1982].

The Algorithmic Procedure

Although in the theoretical account of the method, we divide after each iteration by \( \lambda_1 \) (1.10.5) in practice, of course, \( \lambda_1 \) is not known beforehand. All that is necessary is to normalise \( A^k x \). In this algorithm, we divide this vector by the \( \lambda_2 \) vector norm. The Power method then proceeds for \( i=1, 2, \ldots \).

1. Let \( y_{i-1} = A x_{i-1} \)
2. Set \( \beta_i = y_{i}^T x_{i-1} \)
3. Let \( \eta_i = \left( \frac{\beta_i}{y_{i}^T y_i} \right)^{1/2} \)
4. Set \( x_i = y_i / \eta_i \) and return to step 1 until the stopping criteria is achieved.

Note that \( \beta_i \) is the approximation to \( \lambda_1 \) and that each \( x_i \) is an approximate eigenvector.
CHAPTER 2

ELLiptic PROBLEMS in one-dIMension,

two-dIMensIon and-three dIMension
2.1 ELLIPTIC PROBLEMS

Since many problems solved in this thesis are of elliptic type, it is essential we give an application of the method of finite differences to approximate the solution for the 'self-adjoint' one-dimensional, two-dimensional and three-dimensional problems.

One-Dimensional Case

We will be concerned first with the one-dimensional 'self-adjoint' problem defined by,

$$\frac{\partial}{\partial x}(c_1(x) \frac{\partial u}{\partial x}) + \sigma(x)u(x) + f(x) = 0,$$

(2.1.1)

along the line $R=(a,b)$ and,

$$u(a) = u_a, \quad u(b) = u_b,$$  

(2.1.1a)

The functions $c_1$, $\sigma$ and $f$ are assumed to be 'sufficiently smooth' functions (continuous and differentiable) and satisfy the conditions,

$$c_1(x) > 0, \quad \sigma(x) \leq 0,$$

(2.1.1b)

in $(a,b)$.

Consequently, from (1.1.3) it follows that the equation (2.1.1) is of elliptic type. This section provides information for incorporation with or comparison with the new computational techniques for solving certain linear systems of equations developed in Chapter 6 and Chapter 7.

As described in Section 1.2, we superimpose a mesh over the line $a<x<b$ and we assume a mesh size $h$ on the interval. Now, using Taylor's theorem we can derive in an analogous manner to equation (1.2.4),

$$\frac{\partial}{\partial x} \left[ c_1(x) \frac{\partial u}{\partial x} \right] = \frac{c_1(x+h) \frac{\partial u}{\partial x} - c_1(x-h) \frac{\partial u}{\partial x}}{h}$$

$$= \frac{c_1(x+h) [u(x+h)-u(x)] - c_1(x-h) [u(x)-u(x-h)]}{h}$$

finally we obtain a finite difference representation of the form,

$$h^2 \frac{\partial}{\partial x} \left[ c_1(x) \frac{\partial u}{\partial x} \right] = u(x+h) [c_1(x+\frac{h}{2}) - u(x)] [c_1(x+\frac{h}{2}) + c_1(x-\frac{h}{2})] - u(x-h) [c_1(x-\frac{h}{2})].$$

(2.1.3)
provided $x$ is a regular mesh point. The Taylor series expansion of $U(x,y)$ about $x$ which is irregular may still be used, giving the difference equation similar to equation (1.2.7-1.2.11) and (2.1.3). Let $u_i$ denote $u(\text{ih})$, substitution of the above finite-difference approximations for the derivatives in (2.1.3) leads to the following three-point finite-difference equation at the point $(\text{ih})$,

$$-t_i u_{i+1} + d_i u_i - s_i u_{i-1} = b_i,$$

where $d_i, t_i, s_i > 0$ and $d_i > t_i + s_i$ for all $i$ with equality at interior if $\sigma = 0$.

Equation (2.1.4) may be pictorially represented by a computational 'molecule' (stencil or sten) in Figure 2.1.1.

![FIGURE 2.1.1](image)

which when applied at each regular grid point, yields a set of inhomogeneous, linear, simultaneous, difference equations, which can be expressed in matrix notation as,

$$Au = b,$$

where the vectors $u$ and $b$ consist of unknown approximate solution $u_i$ and the known boundary value plus the quantity $h^2 f(x)$ respectively.

The coefficient matrix $A$ is a sparse, real, tridiagonal, $(n \times n)$ matrix (n being the number of unknown mesh points) whose diagonal entries are $\{-h^2 \frac{\partial^2}{\partial x^2} + C_1(x + \frac{h}{2}) + C_1(x - \frac{h}{2})\}$ and the off-diagonal elements consist of the quantities $-C_1(x + \frac{h}{2}), -C_1(x - \frac{h}{2})$ which do not correspond to boundary point, or zeros.

Other properties of the coefficient matrix $A$ are:

i. symmetric if $C_1 = \text{const}$ or the self-adjoint property is true

ii. positive definite

iii. diagonally dominant

\[ (2.1.6a) \]

\[ (2.1.6b) \]

\[ (2.1.6c) \]
iv. Irreducible, and v. It has positive diagonal and non-positive off diagonal elements. 

If all the interior mesh points in the line $R$ are regular, then $A$ will be symmetric, then it has real eigenvalues. By Theorem 1.4 and 1.5, it will be positive definite. Property v. results from the diagonal entries of $A$ which are given previously.

Properties iii. and iv follow from Definition 1.1 and Theorem in (Varga, 1962, p.20) respectively.

The Two-Dimensional Case

We consider now a class of problems defined by the self-adjoint P.D.E. in two space dimensions, namely:

$$
\frac{\partial}{\partial x}(C_1(x,y) \frac{\partial u}{\partial x}) + \frac{\partial}{\partial y}(C_2(x,y) \frac{\partial u}{\partial y}) + \sigma(x,y) u(x,y) + f(x,y) = 0
$$

(2.1.7a)

in $R$, and,

$$
U(x,y) = u_0(x,y), \quad (x,y) \in C,
$$

(2.1.7b)

where the region $R$ and its boundary $C$ are defined in Section 1.1. The functions $C_1, C_2, \sigma$ and $f$ are assumed to be 'sufficiently smooth' functions and satisfy in $R = R + C$ the conditions,

$$
C_1(x,y) > 0, \quad C_2(x,y) > 0, \quad \sigma(x,y) \leq 0.
$$

(2.1.8)

Consequently, from (1.13) it follows that equation (2.1.7) is of elliptic type.

As described in Section 1.2 we superimpose a mesh over the region $R$, and we assume (unless otherwise stated) a constant mesh size $h$ in both coordinate directions. Now, using Taylor's theorem we can derive in an analogous manner to equations (2.1.2) finite difference representations of the form,

$$
2 h \frac{\partial}{\partial x} \left[ C_1(x,y) \frac{\partial u}{\partial x} \right] = C_1(x+h/2,y) \left[ u(x,y) - u(x+h,y) \right]
$$

$$
- C_1(x-h/2,y) \left[ u(x,y) - u(x-h,y) \right],
$$

(2.1.9)

provided $(x,y)$ is a regular mesh point. Analogously, a similar expression
for $\frac{\partial^2}{\partial y^2} c_2(x,y) \frac{\partial u}{\partial y}$ can be obtained. In the case of an irregular mesh point, the Taylor series expansion of $u(x,y)$ about such a point may still be used, giving approximate formulae similar to (1.2.7)-(1.2.11) and (2.1.9). Let $u_{i,j}$ denote $u(ih,jh)$, substitution of the above finite difference approximations for the derivatives in (2.1.9) leads to the following five-point difference equation at the point $(ih,jh)$.

\[
d_{i,j} u_{i,j} - k_{i} u_{i,j-1} + l_{i} u_{i,j+1} - k_{i+1} u_{i+1,j} + l_{i+1} u_{i-1,j} = b_{i,j} \quad (2.1.10)
\]

where $d_{i,j}, t_{i,j}, r_{i,j}, k_{i,j}, l_{i,j} > 0$ and $d_{i,j}, t_{i,j}, r_{i,j}, k_{i,j}, l_{i,j}$ for all $i,j$ with equality at the interior point if $\sigma = 0$.

Equation (2.1.10) may be pictorially represented by a computational 'molecule' i.e. Figure 2.1.2 which when applied at each regular mesh point,

![Figure 2.1.2](image)

assuming that a fixed labelling is considered, yields a set of inhomogeneous, linear, simultaneous, symmetric difference equations, which can be expressed as the system (2.1.5).

Here $A$ is a real, sparse, banded, $(n^2 \times n^2)$ matrix, $n^2$ being the number of unknown mesh points) whose diagonal entries are $\{-h^2 \sigma(x,y)+c_1(x+\frac{h}{2},y)+c_1(x-\frac{h}{2},y)+c_2(x,y+\frac{h}{2})+c_2(x,y-\frac{h}{2})\}$, and whose off-diagonal elements consist of the quantities $-c_1(x,\frac{h}{2},y)$, $-c_2(x,\frac{h}{2},y)$, $-c_2(x,y+\frac{h}{2})$, $-c_2(x,y-\frac{h}{2})$, respectively which do not correspond to boundary points, or zeros. The vector $u$ consists of unknown approximate solutions $u_{i,j}$ and $b$ is a vector consisting of the known boundary values plus the quantities $h^2 f(x,y)$. Properties of (2.1.6) are also satisfied by the matrix $A$. 
Three Dimensional Case

For the three-dimensional case, the self-adjoint P.D.E. is defined by

$$
\frac{\partial}{\partial x}(C_1(x,y,z) \frac{\partial U}{\partial x}) + \frac{\partial}{\partial y}(C_2(x,y,z) \frac{\partial U}{\partial y}) + \frac{\partial}{\partial z}(C_3(x,y,z) \frac{\partial U}{\partial z}) +
\sigma(x,y,z) U(x,y,z) + f(x,y,z) = 0
$$

(2.1.11)

\((x,y,z) \in \mathbb{R}, \text{ together with,}\)

$$
U(x,y,z) = U_0(x,y,z), \quad (x,y,z) \in \mathbb{C}
$$

(2.1.11a)

where the region \(\mathbb{R}\) and its boundary \(\mathbb{C}\) are defined in Section 1.1. The coefficient of (2.1.11) are assumed to be 'sufficiently' smooth functions.

The region under consideration \(\mathbb{R}\) is covered by a volumetric mesh system \(\mathbb{R}_h\), with a constant mesh size \(h\) in the three coordinate directions. Analogous to the previous cases, we obtain discrete approximations for \(x\)-derivative of the form,

$$
h^2 \frac{\partial}{\partial x} \left( C_1(x,y,z) \frac{\partial U}{\partial x} \right) = C_1(x-h/2,y,z) \left[ u(x+h,y,z) - u(x,y,z) \right]
- C_1(x+h/2,y,z) \left[ u(x,y,z) - u(x-h,y,z) \right]
$$

(2.1.12)

for a regular mesh point \((x,y,z)\). Analogously, a similar expression for

$$
\frac{\partial}{\partial y} \left( C_2(x,y,z) \frac{\partial U}{\partial y} \right) \quad \text{and} \quad \frac{\partial}{\partial z} \left( C_3(x,y,z) \frac{\partial U}{\partial z} \right)
$$

can be obtained.

Let \(u_{i,j,k}\) denote \(u(ih,jh,kh)\). When the seven-point, three-dimensional molecule (Figure 2.1.3) is used,
the following finite-difference equation is obtained,

\[
\begin{align*}
D_{i,j,k}^0& u_{i,j,k}^{T} - D_{i,j,k}^0 L_{i,j,k} = b_{i,j,k} \\
D_{i,j,k}^0 & = a_{i,j,k}
\end{align*}
\]  

(2.1.13)

(representing a mnemonic abbreviation for point Inplane, Outplane, Top, Bottom, Right and Left of the point \( (ih,jh,kh) \) and)

\[
\begin{align*}
D_{i,j,k}^0 & u_{i,j,k}^{T} - D_{i,j,k}^0 L_{i,j,k} = b_{i,j,k} \\
D_{i,j,k}^0 & = a_{i,j,k}
\end{align*}
\]  

(2.1.14)

for all \( i,j,k \) with equality at interior points if \( \sigma = 0 \). Under the same consideration as in the 1D and 2D-cases, the set of symmetric difference equations obtained may be expressed as the system (2.1.5). Properties (2.1.6) are also valid for the coefficient matrix \( A \).

Henceforth, throughout this thesis the solution of the system \( Au = b \) will be restricted to matrices \( A \) with properties (2.1.6) unless otherwise specified.
2.2 THE MODEL PROBLEMS

We will consider first the one-dimensional Laplace model problem.

Laplace's equation in one-direction is defined by,
\[ \frac{\partial^2 u}{\partial x^2} = 0, \]  
(2.2.1)
along the line \( R=(0,1) \), with the boundary condition,
\[ u(0) = 0, \quad u(1) = 0. \]  
(2.2.2)

The interval \( 0 \leq x \leq 1 \) is covered by mesh points \( (x_i) \) where \( x_i = i h, \ i = 0, 1, \ldots, N \) where \( h \) is a distance between two adjacent points. Note that this is a special case \( c_1 = 1, \ \sigma = \tau = 0 \) in (2.1.1).

We now approximate (2.2.1) by the finite difference method. The following three point formula,
\[ 2u_i - u_{i-1} - u_{i+1} = 0 \]  
(2.2.3)
is obtained.

If we now superimpose a mesh over the region \( R \) row-wise (Figure 2.2.1)

![Figure 2.2.1](image)

the coefficient matrix \( A \) of the obtained system (2.1.5) is a real, square, tridiagonal matrix of order \( n = (N-1) \) and is given by
\[
A = \begin{bmatrix}
2 & -1 & & & \\
-1 & 2 & -1 & & \\
& -1 & 2 & -1 & \\
& & -1 & 2 & \\
& & & & 0
\end{bmatrix}
\]  
(2.2.4)

It is easily seen that the matrix \( A \) has properties (2.1.6).

Two-Dimensional Laplace Model Problem

Let us consider the following 2D-model problem.

We wish to obtain an approximation to the function \( U(x,y) \) which satisfies,
\[
\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0 \text{ (Laplace's equation)} \tag{2.2.5}
\]

in the unit square \( R \) with boundary \( C \), subject to Dirichlet boundary conditions,

\[ U(x, y) = 0, \quad (x, y) \in C \tag{2.2.5a} \]

The region \( R \) under consideration is covered by a rectilinear net with mesh spacing \( h \) in the \( x \) and \( y \) directions and mesh points \( (x_i, y_j) \) where \( x_i = ih, \ i = 0, 1, \ldots, N; \ j = 0, 1, \ldots, N \). Note that this is the special case \( c_1 = c_2 = 1 \), \( \sigma, \tau = 0 \) in (2.1.7). We now approximate (2.2.5) by the finite difference method, we obtain the following five-point formula,

\[ 4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = 0, \tag{2.2.6} \]

If we order the \( (N-1)^2 \) internal points row-wise (Fig. 2.2.2)

![Figure 2.2.2](image)

the coefficient matrix \( A \) of the obtained system (2.1.5) is a real, square, banded, sparse matrix of order \( n^2 = (N-1)^2 \) and of the general form,

\[
A = \begin{bmatrix}
A_1 & -I \\
-I & A_2 & -I \\
-I & -I & \ddots & \ddots \\
0 & -I & \ddots & \ddots & \ddots \\
& & & -I & A_n
\end{bmatrix} \tag{2.2.7}
\]
where I, is the unit matrix of order n and A_i, i ∈ (1, n) matrices of order n, given by,

\[
A_i = \begin{bmatrix}
4 & -1 & & & \\
-1 & 4 & -1 & & \\
& -1 & 4 & -1 & \\
& & & 4 & -1 \\
& & & -1 & 4
\end{bmatrix}, \quad i \in (1, n),
\]

(2.2.8)

The \((n^2 \times 1)\) column vectors \(u\) and \(b\) are defined as before.

We notice that the matrix \(A\) so obtained has properties (2.1.6).

Three-Dimensional Laplace Model Problem

We consider the Laplace equation in three space dimensions

\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0, \quad (x,y,z) \in \mathbb{R},
\]

(2.2.9)

subject to the Dirichlet boundary conditions,

\[
u(x,y,z) = 0, \quad (x,y,z) \in \mathbb{C}
\]

(2.2.9a)

where \(\mathbb{R} = (0,1) \times (0,1) \times (0,1)\) and \(\mathbb{C}\) its boundary.

The region under consideration \(R\) (unit cube) is covered by an equally spaced three-dimensional grid, defined by,

\[
R_h = \{(ih,jk,kh): \quad 0 \leq i,j,k \leq N\},
\]

(2.2.10)

where \(N^3h = 1\) and \(h\) the grid spacing.

Analogous to (2.2.6), discrete approximation to the derivatives in (2.2.9) leads to the following seven-point finite difference equation,

\[
6u_{i,j,k} - u_{i+1,j,k} - u_{i-1,j,k} - u_{i,j+1,k} - u_{i,j-1,k} - u_{i,j,k+1} - u_{i,j,k-1} = 0,
\]

(2.2.11)

Figure 2.2.3 gives the order of the \((N-1)^3\) internal points of \(R_h\) with increasing values of \(i\), then \(j\) then \(k\).
the coefficient matrix of the obtained system (2.1.5) is a real, square, banded, sparse matrix of order \(n^3 = (N-1)^3\) and of the general form,

\[
A = \begin{bmatrix}
A_1 & -I \\
-I & A_2 & -I & 0 \\
-I & 0 & -I & -I \\
0 & -I & 0 & -I
\end{bmatrix}
\]

where \(I\) is the unit square matrix of order \(n^2\) and \(A_i, i \in (1,n)\) are matrices of order \(n^2\) given by,

\[
A_i = \begin{bmatrix}
B_{j+1} & -J \\
-J & B_{j+2} & -J & 0 \\
-J & 0 & -J & -J \\
0 & -J & 0 & B_{j+n}
\end{bmatrix}
\]

where \(J\) is now the unit matrix of order \(n\) and \(B_i, i \in (1,n)\) are matrices of order \(n\) given by,
Biharmonic One-Dimensional Problem

Preceding sections of this chapter have dealt only with the approximation of second-order elliptic problems and so we turn our attention to equations of fourth order (with mixed partial derivatives for the 2D-case). We use as our first model problem the one-dimensional biharmonic equation,

\[ \frac{\partial^4 u}{\partial x^4} = 0, \]  

(2.2.15)

on the real line \( \mathbb{R} = (0,1) \) and \( C = 0,1 \) its endpoints (boundary). If \( \partial / \partial n \) denotes differentiation in an outward normal direction to the boundary \( C \), the appropriate boundary condition usually takes one of the following forms,

I. \[ \begin{align*}
& u = s(x) \\
& \frac{\partial u}{\partial n} = g(x)
\end{align*} \]  

(2.2.15a)  

(2.2.15b)

or

II. \[ \begin{align*}
& u = s(x) \\
& \frac{\partial^2 u}{\partial n^2} = g(x)
\end{align*} \]

(2.2.16a)  

(2.2.16b)

The line \( \mathbb{R} \) under consideration is divided into the mesh points \( x_i = i h, \) \( i = 0,1,2, ..., N \) with spacing distance \( h \) apart.

Using Taylor's theorem for points \( (i+2) \) and \( (i-2) \) (Figure 2.2.4),

\[ \begin{align*}
& \phantom{-1} -1 \phantom{6} 6 \phantom{-1} 0 \\
& 6 \phantom{-1} -1 \phantom{6} -1 \\
& -1 \phantom{6} -1 \phantom{-1} 6 \\
& 0 \phantom{-1} 6 \phantom{-1} -1
\end{align*} \]

(2.2.14)

The \((n^3 \times 1)\) column vectors \( u \) and \( b \) are defined as before. The matrix \( A \) so obtained has properties (2.1.6).
we have,
\[ U_{i+2} = U_i + 2h \frac{\partial^2 U}{\partial x^2} i + 4h^2 \frac{\partial^2 U}{\partial x^4} i + \frac{8h}{3} \frac{\partial^3 U}{\partial x^3} i + \frac{16h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^5) \]
\begin{equation}
(2.2.17)
\end{equation}
and
\[ U_{i-2} = U_i - 2h \frac{\partial^2 U}{\partial x^2} i + 4h^2 \frac{\partial^2 U}{\partial x^4} i + \frac{8h}{3} \frac{\partial^3 U}{\partial x^3} i + \frac{16h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^5) \]
\begin{equation}
(2.2.18)
\end{equation}
respectively.

Let \( S_1 \) be the summation of (2.2.17) and (2.2.18). Therefore,
\[ S_1 \equiv U_{i+2} - U_{i-2} = 2U_i + \frac{8h^2}{2!} \frac{\partial^2 U}{\partial x^2} i + \frac{32h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^6) \]
\begin{equation}
(2.2.18a)
\end{equation}
It is convenient to rewrite the expressions for the points \((i+1)\) and \((i-1)\) namely,
\[ U_{i+1} = U_i + h \frac{\partial U}{\partial x} i + \frac{h^2}{2!} \frac{\partial^2 U}{\partial x^2} i + \frac{h^3}{3!} \frac{\partial^3 U}{\partial x^3} i + \frac{h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^5) \]
\begin{equation}
(2.2.19)
\end{equation}
and
\[ U_{i-1} = U_i - h \frac{\partial U}{\partial x} i + \frac{h^2}{2!} \frac{\partial^2 U}{\partial x^2} i + \frac{h^3}{3!} \frac{\partial^3 U}{\partial x^3} i + \frac{h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^5) \]
\begin{equation}
(2.2.20)
\end{equation}
Let \( S_2 \) be the summation of (2.2.19) and (2.2.20). Therefore,
\[ S_2 \equiv U_{i+1} + U_{i-1} = 2U_i + \frac{2h^2}{2!} \frac{\partial^2 U}{\partial x^2} i + \frac{2h^4}{4!} \frac{\partial^4 U}{\partial x^4} i + O(h^6) \]
\begin{equation}
(2.2.20a)
\end{equation}
To approximate (2.1.5) we subtract \( 4S_2 \) from \( S_1 \), i.e.,
\[ S_1 - 4S_2 = -6U_i + \frac{\partial^4 U}{\partial x^4} i + O(h^6) \]
\begin{equation}
(2.2.21)
\end{equation}
Therefore
\[ \frac{\partial^4 U}{\partial x^4} i = 6U_i - 4(U_{i+1} + U_{i-1}) + U_{i+2} + U_{i-2} \]
\begin{equation}
(2.2.22)
\end{equation}
or written as a five-point formula,
\[ 6U_i - 4U_{i+1} - 4U_{i-1} + U_{i+2} + U_{i-2} = 0 \]
\begin{equation}
(2.2.23)
\end{equation}
The presence of the terms \( U_{i+2} \) in equation (2.2.23) requires special
 provision to be made when it is applied at points near to the boundary
 \( P_1 \) in Figure 2.2.5. We shall consider only boundary condition I:
\( (2.2.15a \text{ and } b) \).

When (2.2.23) is applied at the mesh point \( P_1 \) (Figure 2.2.5) it
becomes,
6u -4u -4u +u = 0,
1 2 3 0 -1

From (2.2.15a) \( u_0 = s_0 \),
and \( \frac{\partial u}{\partial n} - \frac{\partial u}{\partial x} = -\frac{(u_{1} - u_{-1})}{2h} = g_0 \),
giving, \( u_{-1} = u_1 + 2hg_0 \).

Combining (2.2.23), (2.2.24) and (2.2.26) we obtain,

\[ 7u_1 - 4u_2 + u_3 = 4s_0 - 2hg_0, \]

in which the left hand side of (2.2.28) involves only grid values of \( u \) lying inside \( R \) and its right hand side consists of known function values. A similar expression can be obtained when (2.2.23) is applied at mesh point \( N \).

For our model problem, we take \( s(x) = g(x) = 0 \), then (2.2.28) becomes,

\[ 7u_1 - 4u_2 + u_3 = 0. \]

If we order the \((N-1)\) internal points (Figure 2.2.5) the coefficient matrix \( A \) of the obtained system (2.1.45) is a real, square, five-diagonal, sparse matrix of order \( n = (N-1) \) and of the form,

\[
A = \begin{bmatrix}
7 & -4 & 1 & 0 & 0 & 0 \\
-4 & 6 & -4 & 1 & 0 & 0 \\
1 & -4 & 6 & -4 & 1 & 0 \\
0 & 1 & -4 & 6 & -4 & 1 \\
1 & 0 & 1 & -4 & 6 & -4 \\
1 & 1 & 0 & 1 & -4 & 7 \\
\end{bmatrix}
\]

The matrix \( A \) so obtained has the properties (2.1.6).
Biharmonic Two-Dimensional Problem

We consider now the equation of the fourth order with mixed partial derivatives. As a model problem we consider the biharmonic equation in two-dimensions,

\[
\frac{\partial^4 U}{\partial x^4} + 2 \frac{\partial^4 U}{\partial x^2 \partial y^2} + \frac{\partial^4 U}{\partial y^4} = 0,
\]

in the bounded region \( R = (0,1) \times (0,1) \) and \( C \) its boundary. If \( \partial / \partial n \) denotes differentiation in an outward normal direction to the boundary \( C \), the appropriate boundary conditions usually take one of two forms,

I. \[ U = s(x,y) \] for \((x,y) \in C\) \hfill (2.2.31a)

\[ \frac{\partial U}{\partial n} = g(x,y) \] \hfill (2.2.31b)

or

II. \[ U = s(x,y) \] for \((x,y) \in C\) \hfill (2.2.32a)

\[ \frac{\partial^2 U}{\partial n^2} = g(x,y) \] \hfill (2.2.32b)

The region \( R \) under consideration is covered by a rectilinear net mesh spacing \( h \) in the \( X \) \& \( Y \) directions and mesh points \((x_i, y_j)\) where \( x_i = ih, i=0,1,...,N \)

\( y_j = jh, j=0,1,...,N. \)

The desired approximation for a given region \( R \) of the \( X \) \& \( Y \) plane is obtained by superimposing a mesh of lines parallel to the reference axes of the region. For ease of discussion, it is convenient to rewrite the mesh point \((i,j)\) [in Laplace 2-Dimension] as the point \( O \) (Figure 2.2.6).

![Figure 2.2.6](image-url)
and we shall assume that it is the origin of the x and y axes and that the
value \( U(x,y) \) at that point is \( U_0 \). By virtue of Figure 2.2.6, the following
can be shown to be valid:

\[
\begin{align*}
\frac{\partial^2 u}{\partial x^2} \bigg|_0 &= \frac{u_1 - u_3}{2h} \\
\frac{\partial^3 u}{\partial x \partial y} \bigg|_0 &= \frac{u_2 - u_4}{2h} \\
\frac{\partial^2 u}{\partial y^2} \bigg|_0 &= \frac{u_1 - 2u_0 + u_3}{2h} \\
\frac{\partial^3 u}{\partial x \partial y} \bigg|_0 &= \frac{u_2 - 2u_0 + u_4}{h^2}
\end{align*}
\] (2.2.33 - 2.2.36)

To obtain the difference approximation for each term of (2.2.31) we use
Taylor's theorem to express the general formula,

\[
U_{i+r,j+s} = U_{i,j} + \frac{3U_{i,j}}{3!} r \Delta x + \frac{3U_{i,j}}{3!} s \Delta y
\]

\[
+ \frac{1}{2!} \left[ \frac{3U_{i,j}}{3!} r^2 \Delta x^2 + 2 \frac{3U_{i,j}}{3!} r s \Delta x \Delta y + \frac{3U_{i,j}}{3!} s^2 \Delta y^2 \right]
\]

\[
+ \frac{1}{3!} \left[ \frac{3U_{i,j}}{3!} r^3 \Delta x^3 + \frac{3U_{i,j}}{3!} r^2 s \Delta x^2 \Delta y + \frac{3U_{i,j}}{3!} r s^2 \Delta x \Delta y^2 + \frac{3U_{i,j}}{3!} s^3 \Delta y^3 \right]
\]

\[
+ \frac{1}{4!} \left[ \frac{3U_{i,j}}{3!} r^4 \Delta x^4 + \frac{3U_{i,j}}{3!} r^3 s \Delta x^3 \Delta y + \frac{3U_{i,j}}{3!} r^2 s^2 \Delta x^2 \Delta y^2 \\
+ \frac{3U_{i,j}}{3!} r s^3 \Delta x \Delta y^3 + \frac{3U_{i,j}}{3!} s^4 \Delta y^4 \right]
\]

+ higher order terms. (2.2.39)

We rewrite,

\[
U(x+kh) = U(x)e^{Dx} ,
\] (2.2.40)

where,

\[
D_x = \frac{3}{3} \frac{\partial}{\partial x}
\]

(similarly for the y direction). Then we express the symmetric sum (see

\[\text{ Analogous to equation (1.2.6), Section 1.2]}

Vichnevsky, 1981) obtained by taking values of $U$ on a concentric circle of radius $h$,

$$ S_1 = U_1 + U_2 + U_3 + U_4 $$

$$ = (e^{hD_x} + e^{hD_y} + e^{-hD_x} + e^{-hD_y}) U_0 $$

$$ = [2 \cosh(hD_x) + 2 \cosh(hD_y)] U_0 $$

$$ = [4 + 4h^2(D_x^2 + D_y^2) + \frac{h^4}{4!}(D_x^4 + D_y^4) + \frac{h^6}{6!}(D_x^6 + D_y^6)] U_0 \quad (2.2.41) $$

From this we derive the well known formula,

$$ \frac{U_1 + U_2 + U_3 + U_4 - 4U_0}{h^2} = D_x^2 + D_y^2 + O(h^2) \quad (2.2.42) $$

The next symmetric sum is,

$$ S_2 = U_5 + U_6 + U_7 + U_8 $$

$$ = [e^{h(D_x + D_y)} + e^{h(-D_x - D_y)} + e^{h(D_x - D_y)} + e^{h(-D_x + D_y)}] U_0 $$

$$ = 4(\cosh(hD_x) \cosh(hD_y)) U_0 $$

$$ = [4 + \frac{4h^2}{2!}(D_x^2 + D_y^2) + \frac{4h^4}{4!}(D_x^4 + 6D_x^2 D_y^2 + D_y^4) $$

$$ + \frac{4h^6}{6!}(D_x^6 + 15D_x^4 D_y^2 + 15D_x^2 D_y^4 + D_y^6)] U_0 $$

$$ = 4 + 2h^2 V^2 + \frac{1}{6} h^4(V^4 + 4D_x^4 D_y^4) $$

$$ + \frac{1}{180} h^6 [V^6 + 12V^2 D_x^4 D_y^2 + \ldots] U_0 $$

$$ + \text{higher order terms} \quad (2.2.43) $$

where $V^2 = D_x^2 + D_y^2$

$$ V^4 = D_x^4 + 2D_x^2 D_y^2 + D_y^4 \quad \text{etc.} $$

We may combine these expressions to obtain higher-order approximations and we find, for instance,

$$ \frac{4S_1 + S_2 - 2U_0}{6h^2} = V^2 U_0 + \frac{1}{12} h^2 V^4 U_0 + O(h^4) \quad (2.2.44) $$

The next sum is,

$$ S_3 = U_9 + U_{10} + U_{11} + U_{12} $$

$$ = (e^{2hD_x} + e^{2hD_y} + e^{-2hD_x} + e^{-2hD_y}) U_0 $$
\[
2^2 + 4^2 + 4^4 + 4^4 + 4^4 + 4^4 + 4^4 = (4 + 4h^2 + \frac{4}{3} h^4 (V^4 - 2D_x D_y^2) + \frac{8}{45} (V^6 - 3V^2 D_x D_y^6) + \ldots)U_0
\]

(2.2.45)

+ higher order terms.

We may use \( U_0, S_1, S_2 \) and \( S_3 \) to approximate \( V^4 U \) (see Figure 2.2.7)

\[
\frac{\nabla^4 U_0}{h^4} = 20U_0 - 8S_1 + 2S_2 + S_3 + o(h^2)
\]

(2.2.46)

\begin{figure}[h]
\centering
\includegraphics[width=0.6\textwidth]{figure2.2.7}
\caption{FIGURE 2.2.7}
\end{figure}

or written as the 13-point finite difference formula,

\[
20u_{i,j} - 8(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1})
+ 2(u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1})
+ u_{i+2,j} + u_{i-2,j} + u_{i,j+2} + u_{i,j-2} = 0,
\]

(2.2.48)

The presence of the terms \( u_{i+2,j} \) and \( u_{i,j+2} \) in equation (2.2.48) requires special provision to be made when it is applied at mesh points outside the boundary. We shall consider boundary condition I only (2.2.31a & 31b).

When (2.2.48) is applied at the mesh point \( (h, 2h) \) (i.e. \( P_2 \) in Figure 2.2.8) it becomes,
\[ 2u_{1,2} - 8(u_{2,2} + u_{0,2} + u_{1,3} + u_{1,1}) + 2(u_{2,3} + u_{2,1} + u_{0,3} + u_{0,1}) + u_{1,2} + u_{1,1} + u_{1,4} = 0, \]

From the boundary condition (2.2.31a) we obtain,

\[ u_{0,1} = s_{0,1}, u_{0,2} = s_{0,2}, u_{0,3} = s_{0,3}, \text{ and } u_{1,0} = s_{1,0}. \]  

Further, since \( \partial u/\partial n = -\partial u/\partial x \) on \( x=0 \), the boundary condition (2.2.31b) is replaced by,

\[ -(u_{1,2} - u_{1,1}) \frac{\partial u}{\partial x}
\]

Combining (2.2.48), (2.2.49) and (2.2.50) we have,

\[ 2u_{1,2} - 8(u_{2,2} + u_{1,3} + u_{1,1}) + 2(u_{2,3} + u_{2,1} + u_{3,2} + u_{1,4}) =
\]

\[ 8s_{0,2} - 2(s_{0,1} + s_{0,1}) - s_{1,0} - 2h \delta_{0,2}, \]

in which the left hand side of (2.2.51) involves only the grid values of \( u \) lying inside \( R \) and its right hand side consists of known function values.

For our model problem, we take \( s(x,y) = g(x,y) = 0 \). Then (2.2.51) becomes,

\[ 21u_{1,2} - 8(u_{2,2} + u_{1,3} + u_{1,1}) + 2(u_{2,3} + u_{2,1} + u_{3,2} + u_{1,4}) = 0. \]  

\[ \text{FIGURE 2.2.8} \]
A similar procedure is adopted for all mesh points adjacent to the boundary. For the corner point \( P_1 \) in Figure 2.2.8, the approximation to the boundary condition \( I \) must be invoked at both points \( B_1 \) and \( B_4 \). At this point (2.2.52) becomes,

\[
22u_{1,1} - 8(u_{1,2} + u_{0,1} + u_{2,1}) + 2u_{2,2} + u_{3,1} + u_{1,3} = 0, \quad (2.2.53)
\]

A similar procedure is adopted for all corner points.

If we order \((N-1)^2\) internal points rowwise as in Figure 2.2.2, the coefficient matrix \( A \) of the obtained system (2.1.5) is a real square, banded, sparse matrix of order \( n^2 = (N-1)^2 \) and of the general form,

\[
\begin{pmatrix}
A_1 & B_1 & I \\
B_1 & A_2 & B_2 & I \\
I & B_2 & A_3 & I \\
& I & B_{n-1} & A_n \\
& & & I
\end{pmatrix}
\]

where \( I \) is the unit square matrix of order \( n \), \( A_1 \) and \( A_n \) are the matrices of order \( n \) given by,

\[
A_1 = \begin{pmatrix}
22 & -8 & 1 \\
-8 & 21 & -8 & 1 \\
1 & -8 & 21 & -8 & 1 \\
& 1 & -8 & 21 & -8 & 1 \\
& & 1 & -8 & 21 & -8 \\
& & & 1 & -8 & 22
\end{pmatrix}
\]

and \( A_i \), \((i=2,3,\ldots,n-1)\) are the matrices of order \( n \) given by,
and $B_i$, $i \in (1,n-1)$ matrices of order $n$ given by,

$$B_i = \begin{bmatrix} -8 & 2 & & & & \\ & -8 & 2 & & & \\ & & -8 & 2 & & \\ & & & -8 & 2 & \\ & & & & -8 & 2 \\ & & & & & -8 \end{bmatrix} i \in (1,n-1) \quad (2.2.57)$$

The $(n \times 1)$ column vectors $u$ and $b$ are defined as before.

We notice that the matrix $A$ so obtained has properties (2.1.6).
CHAPTER 3

THE NUMERICAL SOLUTION OF LINEAR EQUATIONS

AND ILL-CONDITIONED SYSTEMS
3.1 **INTRODUCTION**

We will consider solving the system

\[ Au = b, \]

(3.1.1)

where \( A \) is a square matrix and \( b \) is a given column matrix while \( u \) is an unknown column matrix. We will assume that a unique solution exists. For such systems, \( A \) must be non-singular.

Practical methods for the solution of such systems fall mainly into two classes, direct and iterative. Direct methods are those which, in the absence of round-off or other errors, will yield the exact solution in a finite number of steps. In practice, because a computer works with a finite word length, direct methods do not lead to exact solutions. In such methods a sequence of operations is performed, in general, once only which results in an approximation to the true results. The approximation enters only because multiplications and divisions are stored and subsequently used with rounding errors. A large part of numerical analysis is concerned with why and how these errors arise, and with the search for methods which minimize such errors.

Iterative methods are those which start with an initial approximation and which, by applying a suitably chosen algorithm, leads to successively better approximations. The number of sequences of approximations may be infinite whose limit is the exact solution. In practical terms, we can only hope to obtain an approximate solution, by specifying the convergence criteria, that is, we will terminate the iteration if the difference between two successive approximations is within a given tolerance \( \epsilon \).

For large, sparse matrices having a few non-zero elements, they are ideally suited to solution by iterative methods which take advantage of the sparse nature of the matrix involved. Iterative methods do not change the structure of the original matrix and so sparsity is preserved. The main
advantage of the iterative methods is that sparse matrix techniques can be used to store only the non-zero elements of the coefficient matrix and hence optimise the amount of storage used. For dense matrices, the best strategy is to use direct methods.
3.2 DIRECT METHODS

Gauss Elimination Process

Gauss elimination is the familiar variable elimination technique whereby the variables are eliminated one at a time to reduce the original system to an equivalent triangular system. The latter system can then be solved by 'back-substitution'.

We rewrite (3.1.1) in the following form:

\[ \begin{align*}
    a_{11}u_1 + a_{12}u_2 + \ldots + a_{1n}u_n &= b_1 \\
    a_{21}u_1 + a_{22}u_2 + \ldots + a_{2n}u_n &= b_2 \\
    \vdots & \\
    a_{n1}u_1 + a_{n2}u_2 + \ldots + a_{nn}u_n &= b_n
\end{align*} \]  

(3.2.1)

as the system of equations. The first step in this procedure is to replace the ith equation by an equation that is the result of multiplying the first equation by \((-a_{i1}/a_{11})\) and adding it to the original ith equation. Proceeding thus for \(i=2,3,\ldots,n\), we obtain a system of equations equivalent to (3.2.1):

\[ \begin{align*}
    a_{11}u_1 + a_{12}u_2 + \ldots + a_{1n}u_n &= b_1 \\
    a_{22}u_2 + \ldots + a_{2n}u_n &= b_2^{(1)} \\
    \vdots & \\
    a_{n2}u_2 + \ldots + a_{nn}u_n &= b_n^{(1)}
\end{align*} \]  

(3.2.2)

We assume here that \(a_{11} \neq 0\). If \(a_{11} = 0\), we find an element \(a_{i1} \neq 0\), interchange the first and the ith row, and proceed in the same fashion. A useful strategy to avoid (if possible) such zero divisions is to rearrange the equations so as to put the coefficient of largest magnitude on the diagonal at each step. This is called pivoting and will be discussed later. The \((r-1)\)st system has the following form:

\[ \begin{align*}
    a_{11}^{(r-1)}u_1 + a_{12}^{(r-1)}u_2 + \ldots + a_{1,r-1}^{(r-1)}u_{r-1} + a_{1r}^{(r-1)}u_r + \ldots + a_{1n}^{(r-1)}u_n &= b_1^{(r-1)} \\
    a_{22}^{(r-1)}u_2 + \ldots + a_{2,r-1}^{(r-1)}u_{r-1} + a_{2r}^{(r-1)}u_r + \ldots + a_{2n}^{(r-1)}u_n &= b_2^{(r-1)} \\
    \vdots & \\
    a_{n2}^{(r-1)}u_2 + \ldots + a_{nn}^{(r-1)}u_n &= b_n^{(r-1)}
\end{align*} \]  

(3.2.3)
\[
\begin{pmatrix}
(a_{r1} \ldots a_{r(r-1)}) & u_r \\
(a_{n1} \ldots a_{n(r-1)}) & u_n \\
\vdots & \vdots \\
(a_{nr} \ldots a_{n(r-1)}) & u_n
\end{pmatrix} = \begin{pmatrix} b_{r}\end{pmatrix}
\]

In words, the first \( r \) equations are already in upper triangular form, while the last \( n-r \) equations involve only the unknown \( u_r, \ldots, u_n \). We continue in this manner until the system is an equivalent upper triangular form of which the solution can easily be found by backsubstitution; that is, by solving the last equation for \( u_n \), then the \((n-1)\)st equation for \( u_{n-1} \), and continuing until each \( u_r, r=n,n-1, \ldots, 1 \) is determined.

As we have seen, the basis of the Gauss method is the use at each stage of a particular element, generally termed the pivot to generate zero's in its column in all rows beneath it. Conte and De Boor (1980) discuss the possible source for which Gauss elimination process could produce completely erroneous answers. It is clear then that the accuracy of the final solution depends on the choice of the pivots. We consider two of the most important pivotal strategies commonly used.

**Partial Pivoting:** Partial pivoting involves choosing for the pivot in each reduced matrix the element of largest magnitude in the relevant column, elements of rows which have previously been pivotal being excluded from consideration. A search is made at the \( r \)th step of the elimination process only amongst the elements \( a_{ir}, (i=r,r+1,\ldots,n) \) and the largest element in absolute value is selected as the pivot.

**Complete Pivoting:** In this strategy, a search is made at the start of the \( r \)th \((r=1,2,\ldots,n-1)\) step of elimination to find the maximum element in absolute value of the submatrix \((n-r-1)\times(n-r+1)\) which is formed by eliminating the \( r \)th row and column \((r=2,3,\ldots,n-1)\) of the matrix and it is used as a pivotal element for the stage.

The advantage of partial pivoting is that the pivotal row is always multiplied by a number of magnitude less than unity before its subtraction.
from other rows in the elimination process. Any rounding errors present in the pivotal row are thereby decreased in absolute magnitude, the propagated effect of these errors being decreased correspondingly. [R.J. Gault et al., 1974]. Although for full pivoting the propagation of rounding errors is theoretically much better than by partial pivoting, some of its disadvantages are the greater organisational problems, the longer search for the largest pivot, and the fact that matrices of certain important particular forms, containing a high proportion of zero elements arranged in a special pattern, may have this pattern preserved by partial pivoting but destroyed by full pivoting. Most authors of the literature recommend that the simpler partial pivoting strategy is to be preferred for most purposes.

Gauss-Jordan Elimination

There are many variants to the Gaussian elimination scheme. One variant that is sometimes used is the Gauss-Jordan elimination process. In it, the elements above the diagonal are made zero at the same time that zeros are created below the diagonal. This method then, leads to a final reduced diagonal rather than triangular form.

In step $r$ of the Jordan elimination, choose the pivot element as before. Then, define,

$$ a_{rj}^{(r+1)} = \frac{a_{rj}^{(r)}}{a_{rr}^{(r)}} , \quad j=r,...,n $$  \hspace{1cm} (3.2.4)

$$ b_r^{(r+1)} = \frac{b_r^{(r)}}{a_{rr}^{(r)}} . $$  \hspace{1cm} (3.2.5)

Eliminate the unknown $u_r$ in equations both above and below equation $r$. Define,

$$ a_{ij}^{(r+1)} = a_{ij}^{(r)} - a_{ir}^{(r)} a_{rj}^{(r+1)} $$  \hspace{1cm} (3.2.6)

$$ b_i^{(r+1)} = b_i^{(r)} - a_{ir}^{(r)} b_r^{(r+1)} $$  \hspace{1cm} (3.2.7)

for $j=r,...,n$, $i=r,...,n$, $i\neq r$.

Another way of describing the Gauss-Jordan method is as follows; The $r$th step in forward elimination by the method is equivalent to pre-
multiplication of the set of equations by a matrix,

\[
M_r = \begin{bmatrix}
1 & 0 & \cdots & \alpha_{1r} & 0 & \cdots & 0 \\
0 & 1 & \cdots & \alpha_{2r} & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 & 0 & \cdots & 0 \\
0 & 0 & \cdots & \alpha_{i,r} & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \alpha_{n,r} & 0 & \cdots & 1 \\
\end{bmatrix}
\] (3.2.8)

for \( i=1, \ldots, n \).

This procedure converts the system \( (A \mid b) \) to \( (I \mid b^{(n)}) \), so that at the completion of the above elimination, \( u = b^{(n)} \) and no back-substitution process is required.

Since the execution time and the error caused by rounding-off in the computed solution are both related to the total number of arithmetic operations, we will now compare both of the above methods in terms of measure of work.

The Gaussian elimination method is usually programmed to accommodate \( k \) right hand side vectors, in which case the storage required is \( n(n+k) \) locations plus a possible further \( nk \) for the final solution vectors. Finally, the amount of work involved can be shown to be [Fox, 1964].

<table>
<thead>
<tr>
<th></th>
<th>Gauss</th>
<th>Gauss-Jordan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Division</td>
<td>( n )</td>
<td>( n )</td>
</tr>
<tr>
<td>Multiplication</td>
<td>( n \left( \frac{n^2 + 1}{3} \right) )</td>
<td>( n \left( \frac{n^2 + 1}{2} \right) )</td>
</tr>
<tr>
<td>Addition</td>
<td>( n \left( \frac{n^2 + 1}{6} \right) )</td>
<td>( n \left( \frac{n^2 + 1}{2} \right) )</td>
</tr>
</tbody>
</table>

Thus, we note that, the work of the Gauss-Jordan elimination process is substantially greater, in ratio three to two, than the work of Gauss elimination for any sufficiently large \( n \).
In practice, however, in the solution of sets of equations resulting from finite difference approximations to elliptic partial differential equations, this value of work involved is never actually attained as the matrix \( A \) is never full. Instead, \( A \) is in general a banded matrix, i.e. 
\[
A = (A_{i,j}) \text{ where,} \\
A_{i,j} = 0 \quad \text{if } i-j > q \\
\]
so that the number of non-zero elements in each row is at most \( 2q+1 \) the bandwidth of the matrix. In Gauss elimination the number of multiplications is of order \( 2q^2 n \) in contrast to the factor \( n^3 / 3 \) for full matrix.

**LU-Decomposition**

In the two elimination methods discussed above, our aim was to convert the matrix \( A \), by an ordered sequence of operations, to a form, triangular or diagonal, from which the solution can be obtained with relative ease. We need, for the former, however, only the final upper triangular matrix and the corresponding final right-hand vector in order to complete the solution by back-substitution. The question arises as to whether we can obtain the triangular matrix and the right hand vector without calculating and recording the elements of the intermediate matrices and vectors.

This is achieved in the methods associated with the names of Doolittle, Crout and Cholesky which we now illustrate for a general \((n \times n)\) matrix. For the relation \( A = LU \) or 
\[
\begin{bmatrix}
al_{1,1} & \cdots & a_{1,j} & \cdots & a_{1,n} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{i,1} & \cdots & a_{i,j} & \cdots & a_{i,n} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
a_{n,1} & \cdots & a_{n,j} & \cdots & a_{n,n}
\end{bmatrix} = 
\]

(L is lower triangular and U is upper triangular) all the coefficients in L and U are initially unknown. They are found by the following rules based on matrix multiplication:

\[
\begin{align*}
\text{for } j = i, i+1, \ldots, n : & \quad u_{i,j} = a_{i,j} - \sum_{k=1}^{i-1} l_{i,k} u_{k,j}, \\
\text{for } j = i+1, i+2, \ldots, n : & \quad l_{i,j} = \frac{1}{u_{i,i}} \left[ a_{j,1} - \sum_{k=1}^{i-1} l_{j,k} u_{k,1} \right]
\end{align*}
\]  

(3.2.9)

This factorisation method fails only if one of the diagonal elements of \( u \) is zero which are used as divisors in the second part of (3.2.9).

As in the Gaussian elimination process, similar pivotal strategies must be used to ensure sufficiently accurate results. We will shortly outline a row interchanging modification of the Doolittle factorisation which overcomes this problem.

The following theorems illustrate that the \( LU \) decomposition uniquely exists if \( A \) is a non-singular matrix.

**Theorem 3.2.1**

The \((n \times n)\) matrix \( A \) has an \( LU \) decomposition where \( L \) and \( U \) are non-singular, if the submatrices,

\[
A_k = \begin{bmatrix}
    a_{i,1} & \cdots & a_{i,k} \\
    \vdots & \ddots & \vdots \\
    a_{k,1} & \cdots & a_{k,k}
\end{bmatrix}
\]  

(3.2.10)

are all non-singular (\( A_k \) is called the leading principal matrix of order \( k \)).
Proof:

Let us assume an inductive hypothesis such that

\[ A_{k-1} = L_{k-1}U_{k-1}, \]

where \( L_{k-1} \) is a unit lower triangular matrix.

Let \( A_k \) be partitioned as,

\[
A_k = \begin{pmatrix}
A_{k-1} & b \\
\bar{a}^T & a_{k,k}
\end{pmatrix}
\]

where,

\[
\bar{a}^T = (a_{k,1}, \ldots, a_{k,k-1}) \quad \text{and} \quad b = \begin{bmatrix} a_{1,k} \\ \vdots \\ a_{k-1,k} \end{bmatrix},
\]

and let \( L_k \) and \( U_k \) be defined by,

\[
L_k = \begin{pmatrix}
I_{k-1} & 0 \\
\ell \ell^T & 1
\end{pmatrix} \quad \text{and} \quad U_k = \begin{pmatrix}
U_{k-1} & u \\
0 & u_{k,k}
\end{pmatrix}
\]

where,

\[
\ell = (\ell_{k,1}, \ldots, \ell_{k,k-1}) \quad \text{and} \quad u = \begin{bmatrix} u_{1,k} \\ \vdots \\ u_{k-1,k} \end{bmatrix}
\]

then \( \ell^T, u \) and \( u_{k,k} \) can be uniquely determined such that,

\[
L_k U_k = \begin{pmatrix}
I_{k-1} U_{k-1} & L_{k-1} u \\
\ell \ell^T U_{k-1} & \ell \ell^T u + u_{k,k}
\end{pmatrix}
\]

\[= \begin{pmatrix}
A_{k-1} & b \\
\bar{a}^T & a_{k,k}
\end{pmatrix} = A_k. \tag{3.2.12}
\]

This follows since \( L_{k-1} U_{k-1} = A_{k-1} \) by hypothesis and then

\[
\ell = \bar{a}^T u_{k-1}, \quad u = \ell_{k-1}^{-1} b,
\]
and 

\[ u_{k,k} = a_{k,k} - \frac{1}{u_{k,k-1}} u_{k,k-1} \]

is uniquely determined since \( L_{k-1} \) and \( U_{k-1} \) are non-singular under the hypothesis. Now,

\[ A_1 = (a_{1,1}) = (1)(a_{1,1}) \tag{3.2.13} \]

defines the decomposition of \( A_1 \). Hence, by induction, \( A = A_n \) has an LU decomposition.

We may note that if \( A \) is a real matrix then \( L \) and \( U \) are also real. Row or column diagonally dominant matrices satisfy the hypotheses of the theorem.

**Theorem 3.2.2**

If there exists an LU decomposition of a non-singular matrix \( A \), where \( L \) is unit lower triangular matrix, then this decomposition is unique.

**Proof:**

Let \( A = L(1)^T U(1) = L(2)^T U(2) \) be two decompositions, where the matrices are non-singular. Since \( A \) is non-singular, then,

\[ L^{-1}(2)L(1) = U^{-1}(2)U(1) = I \tag{3.2.14} \]

since \( L^{-1}(2)L(1) \) is unit lower triangular and \( U^{-1}(2)U(1) \) is upper triangular.

Once the matrix \( A \) can be factored (or decomposed) into \( LU \) matrix, then the solution of (3.1.1) can be achieved in two steps:

**Step 1:** Define the intermediate vector \( y = \{y_i\} \) as the solution of

\[ Ly = b \tag{3.2.15} \]

**Step 2:** Having thus computed \( y \), it may be observed that \( u \) is the solution of,

\[ Uu = y \tag{3.2.16} \]

By way of proof, it suffices to multiply (3.2.16) by \( L \)

\[ Wu = Au = Ly = b \tag{3.2.17} \]

which is identical with (2.1.5).

One important element in this procedure is that since \( L \) and \( U \) are both triangular the solution of (3.2.15) consists of a simple explicit forward substitution; whereas the solution of (3.2.16) consists of a simple explicit
backward substitution. Thus, having derived and stored L and U, the actual solution of (3.1.1) consists of two simple sweeps. We can see that after any element of $A$, $a_{ij}$, is once used, it never again appears in the equations. Hence its place in the original $n \times n$ array $A$ can be used to store an element of either $L$ or $U$. Array $A$ can be transformed and becomes,

$$
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
& & \ddots & \vdots \\
a_{n1} & a_{n2} & \cdots & a_{nn}
\end{pmatrix}
\rightarrow
\begin{pmatrix}
u_{11} & u_{12} & \cdots & u_{1n} \\
\ell_{21} & u_{22} & \cdots & u_{2n} \\
\ell_{31} & \ell_{32} & \ddots & \vdots \\
\ell_{n1} & \ell_{n2} & \cdots & u_{nn}
\end{pmatrix}
$$

Because we can condense the $L$ and $U$ matrices into one array and store their elements in the space $A$, this method is often called a compact scheme.

A special advantage of the LU method is its ability to solve the system with new right hand side vectors with great economy of effort. A numerical example that illustrates using the LU method with multiple right hand sides can be found in [Gerald 1978, Section 2.7]. We note in particular that these new right hand sides do not have to be known in advance as is the case with Gaussian elimination.

Another important aspect of this procedure is that when the matrix $A$ is banded, then both of the matrices $L$ and $U$ are also banded, with a bandwidth which does not exceed that of the original matrix $A$. This has important consequences in storage-space requirements for large systems.

**Equivalence of Gaussian Elimination with LU Decomposition**

Define the matrix $M_r$ by

$$
M_r = \begin{pmatrix}
1 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 1 & \cdots & 0 & \cdots & 0 \\
0 & 0 & \cdots & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & \cdots & 1 \\
0 & 0 & \cdots & 0 & \cdots & 0
\end{pmatrix}
$$
i.e. identity matrix with additional element $-m_{i,r}$ in $(i,r)^{th}$ position for $r+1 \leq i \leq n$. Then, premultiplication of a matrix $A$ by $M_r$ has the effect of $-m_{i,r}$ or row $r$ being added to row $i$ for $r+1 \leq i \leq n$.

The Gaussian elimination algorithm without pivoting can then be described theoretically by,

$$M_{n-1}^{-1}M_{n-2}^{-1}...M_{k}^{-1}...M_{1}^{-1}A =$$

$$\begin{bmatrix}
\begin{array}{cccc}
a_{11}^{(1)} & \cdots & a_{1,r}^{(1)} & a_{1,n}^{(1)} \\
\vdots & \ddots & \vdots & \vdots \\
\vdots & & \ddots & \vdots \\
a_{r,r}^{(r)} & \cdots & a_{r,n}^{(r)} & 0 \\
\vdots & \ddots & \vdots & \vdots \\
a_{n,n}^{(n)} & \cdots & 0 & 0 \\
\end{array}
\end{bmatrix} = U, \text{ (say),}
$$

(3.2.18)

where $m_{i,r} = a_{i,r}^{(r)}/a_{r,r}^{(r)}$.

Thus, we may write $MA = U$ where $M = M_{n-1}^{-1}M_{n-2}^{-1}...M_{k}^{-1}...M_{1}^{-1}$, or,

on defining, $M^{-1} = L$, 

$$A = LU . \quad (3.2.19)$$

Here, $L$, being the inverse of a lower triangular, is itself lower triangular.

Now, $M_r^{-1}$ proves to be simply $M_r$ itself with the sign of its off-diagonal elements reversed, and $L$ is given by,

$$L = \begin{bmatrix}
1 & 0 & \cdots & 0 \\
m_{21} & 1 & \cdots & 0 \\
m_{31} & m_{32} & \ddots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
m_{n1} & m_{n2} & m_{n3} & m_{n,n-1}
\end{bmatrix}
$$

It can be shown that the amount of work in the LU decomposition is the same as for Gauss's method.

However, the disadvantage of the Gaussian elimination process in comparison with the LU decomposition is that each time an element of the
reduced matrix is computed and stored a rounding error generally occurs. In the LU decomposition a rounding error can often be avoided by the use of double-precision arithmetic in the calculation of L and U from equation (3.2.9). These results may then be rounded to single precision on completion of each calculation. The removal of the need for computing and recording several intermediate matrices, has, therefore, localised what might otherwise be a significant source of error to a single step in the determination of each element of L and U. The use of double precision arithmetic in this step leads to a degree of accuracy comparable with that attained if the entire Gaussian elimination process were carried out with double-precision arithmetic. This latter is an unattractive proposition because it would require twice as much computer storage as the corresponding single-precision solution.

Doolittle's Method with Row Interchanges

We describe now the equivalence of partial pivoting in fixed point arithmetic. It consists of \( n \) major steps in the \( r \)th of which we determine the \( r \)th column of \( L \), the \( r \)th row of \( (U \mid c) \) and \( r' \). The configuration at the beginning of the \( r \)th step for the case \( n=5, r=3 \) is given by,

\[
\begin{array}{cccccc}
u_{11} & u_{12} & u_{13} & u_{14} & u_{15} & c_1 & s_1 \\
l_{21} & u_{22} & u_{23} & u_{24} & u_{25} & c_2 & s_2 \\
l_{31} & l_{32} & a_{33} & a_{34} & a_{35} & b_3 & s_3 \\
l_{41} & l_{42} & a_{43} & a_{44} & a_{45} & b_4 & s_4 \\
l_{51} & l_{52} & a_{53} & a_{54} & a_{55} & b_5 & s_5 \\
\end{array}
\]

\[ (3.2.20) \]

The \( r \)th major step is as follows:

(i) For \( t=r, r+1, \ldots, n \). Calculate \( a_{tr} = l_{1r} u_{1r} - l_{2r} u_{2r} - \cdots - l_{tr} u_{r-1, r} \), storing the result as a double-precision number \( s_t \) at the end of row \( t \).

(ii) Suppose \( \mid s_{t'} \mid \) is the maximum of the \( \mid s_t \mid \) (\( t=r, \ldots, n \)). Then store
r' and interchange the whole of rows r and r' including the
e, r', a, r', b, r, s'. Round the new s to single precision to give
u, rr and overwrite this on a, rr.

(iii) For t=r+1,...,n. Compute s, /aur to give 1, tr and overwrite on a, tr.

(iv) For t=r+1,...,n. Compute a, rt = -r u, t - r, u, 2t ... - r, r-1 u, r-1, t',
accumulating the inner-product and rounding on completion to give
u, rt. Overwrite to u, rt on a, rt.

(v) Compute b, r - r, r', c, r, c, 2 ... - r, r-1 c, r-1, again accumulating the
inner-product and rounding on completion to give c, r. Overwrite
on b, r.

Wilkinson (1965) gives a numerical example of the above procedure.

Each time one of the diagonal elements of u is evaluated, the effect
of row interchanges is examined, with the object of finding the arrangement
which maximises the absolute value of the element concerned.

The use of row interchanges with this method enables us to combine the
virtues of partial pivoting with further enhancement of accuracy when
double-precision arithmetic is used for calculating the elements of L and U.

At first sight, it appears that the introduction of row interchanges
involves in more arithmetic, since several possible values of each diagonal
element of U must be computed and compared in magnitude. However, the
calculation involved in obtaining those values is not wasted, since these
values are used in obtaining the next column of L to be determined. Apart
from the search process, then, the total arithmetic is the same either with
or without row interchanges. It is apparent that the Doolittle triangular
decomposition method with row interchanges is the best method for general
purpose usage of all those discussed here, provided a computer is available
which permits the use of double-precision arithmetic. Otherwise, there is
little to choose between the elimination and decomposition methods.
The Crout Form of Factorization for Symmetric Matrices

This form of factorisation which is used for symmetric matrices will factorise \( A \) into the form \( U^T D U \) in a recursive procedure where at each step of the process, we write \( A \) as,

\[
A = A^{(1)} = \begin{bmatrix} a_1 & S_1^T \\ S_1 & C_1 \end{bmatrix} \tag{3.2.21}
\]

where \( a_1 \) is a single element, \( S_1 \) is an \((n-1)\times1\) vector and \( C_1 \) is an \((n-1)\times(n-1)\) matrix. The first element of the diagonal matrix \( D \), i.e. \( d_{1,1} \) and the first row of the upper triangular matrix \( U \) is now computed as,

\[
\begin{align*}
\{ u_{1,j} = (S_1^T/a_1), & \quad j=2,3,\ldots,n \\
d_{1,1} = a_{1,1}
\end{align*}
\tag{3.2.22}
\]

We then compute \( A^{(2)} \) as,

\[
A^{(2)} = \begin{bmatrix} a_1 & S_1^T \\ S_1 & C_1 - \frac{S_1 S_1^T}{a_1} \end{bmatrix} \tag{3.2.23}
\]

where the matrix \( \frac{S_1 S_1^T}{a_1} \) is subtracted element by element from \( C_1 \).

In general, at the \( r \)th step (\( 1 \leq r \leq n-1 \)) we have \( A^{(r)} \) as,

\[
A^{(r)} = \begin{bmatrix} A_r & B_r^{T} \\ B_r & C_r \end{bmatrix} \tag{3.2.24}
\]

where \( A_r \) is a \((r-1)\times(r-1)\) matrix, \( B_r \) is a \((n-r+1)\times(r-1)\) array, \( a_r \) is a single element, \( S_r \) is a \((n-1)\times1\) vector and \( C_r \) is a \((n-r)\times(n-r)\) matrix. We now have \( d_{r,r} = a_r \) and the \( r \)th row of the matrix \( U \) is replaced by \( S_r^T/a_r \). Finally, we compute \( A^{(r+1)} \) as:
Let the matrix $A$ be symmetric and positive definite then we can factorise $A$ in the form $L^T L$ where $L$ is lower triangular. As an illustration,

\[
\begin{bmatrix}
  l_{1,1} & \cdots & 0 \\
  l_{2,1} & l_{2,2} & \cdots \\
  \vdots & \ddots & \ddots \\
  l_{n,1} & \cdots & l_{n,n}
\end{bmatrix}
\begin{bmatrix}
  l_{1,1} \\
  l_{2,1} \\
  \vdots \\
  l_{n,1}
\end{bmatrix}
\begin{bmatrix}
  l_{1,1} \\
  l_{2,2} \\
  \vdots \\
  l_{n,n}
\end{bmatrix}
\]

Then equating coefficients on and below the diagonal we obtain the unknown coefficients on $L$ from the following equation,

for $i=j$, $l_{i,j} = (a_{j,j} - \sum_{k=1}^{j-1} l_{j,k})^{1/2}$

for $i=j+1, j+2, \ldots, n$, $l_{i,j} = \frac{1}{l_{i,i}} (a_{j,i} - \sum_{k=1}^{j-1} l_{i,k} * l_{j,k})$

for $j=1, 2, \ldots, n$. 

(3.2.26)
**Theorem 3.2.3**

Let \( P \) be a real symmetric positive definite matrix then there exists a unique decomposition of the form,

\[
P = LDL^T = LL^T
\]

where \( L \) is a real unit lower triangular matrix and \( D \) is diagonal with real positive terms and \( L \) is a real lower triangular matrix with positive diagonal.

**Proof:**

If \( A \) is a real, symmetric and positive definite then its leading principal matrices \( A_r \) are non-singular and moreover are such that,

\[
\det(A_r) > 0, \quad r=1,2, \ldots, n. 
\]

Thus by Theorem 3.2.2 there exists a unique decomposition

\[
A = LU
\]

where \( L \) is a real lower and \( U \) is a real upper triangular matrix.

Moreover, since,

\[
\begin{cases}
\sigma_{11} = \sigma_{11}', \quad r=1 \\
\frac{\det(A_r)}{\det(A_{r-1})}, \quad r>1
\end{cases}
\]  

(3.2.27)

it follows that \( U \) had positive diagonal terms. Now the decomposition may be rewritten in the unique form

\[
A = LDV,
\]  

(3.2.28)

where \( D = \text{diag}(u_{r,r}) \) is a diagonal matrix with positive terms and \( V = D^{-1}U \) is a real unit upper triangular matrix. Uniqueness and symmetry then imply that \( V = L^T \). Thus,

\[
A = LDL^T = LD D^T L^T = LL^T,
\]  

(3.2.29)

where \( L = LD^T \) and \( D = D^T \), and hence \( L \) can be uniquely chosen to have positive diagonal terms.

It can be seen that the use of symmetric elimination methods, i.e. the Crout and the Choleski methods give a significant saving in computational cost. Both methods require \( O(n^3/6) \) multiplications and \( O(n^3/6) \) additions.

There is one case for which the choice is quite definite and that is
the triangulation of a positive definite symmetric matrix. Here the symmetric Choleski decomposition has all the virtues. No interchanges are needed and the work may conveniently be performed in fixed point. It takes full advantage of the symmetricity and if $A$ is a band matrix, i.e. if $a_{i,j} = 0 \ (|i-j|>q)$, then $L_{ij} = 0 \ (|i-j|>q)$, so that the full advantage is taken of this form. In this case the matrices $L$ and $L^T$ formed by the decomposition are also banded of width $q$. Perhaps the only comment we might make is that $n-1$ square roots are required whereas none is needed in ordinary Gaussian elimination (Wilkinson, 1965). The Choleski method is very popular because many elliptic problems lead to matrices $A$ which are symmetric and positive-definite. Vichnevetsky (1981) shows that for banded matrices, the work of decomposing $A$ into $L$ and $U$ is greatly reduced and requires only $nq^2 + O(nq)$ multiplications with a similar number of additions, assuming that $q \ll n$. (For details of all available triangulation methods and their discussion, see Wilkinson, 1965).
3.3 ITERATIVE METHODS

We now consider the task of computing the solution of a system such as that represented by the matrix equation (3.1.1) using iterative methods. Unlike direct methods where a fixed number of operations were involved, iterative methods start from a first (initial) approximation \( u^{(0)} \) of the solution \( u \). These values \( u^{(0)} \) are then substituted in (3.1.1) leading to a new, and hopefully better, estimate of the solution. If this procedure is repeated for a sufficiently large number of iteration cycles, a point will be reached at which the change in the solution effected in two successive cycles is smaller than a pre-assigned tolerance \( \varepsilon \). The procedure is then said to have converged and the solution thus obtained is accepted as the solution to (3.1.1).

Basic Iterative Methods

Several of the best known iterative methods are built around a partition or splitting of \( A \) of (3.1.1) into the form,

\[
A = D - L - U ,
\]

where \( D \) is the main diagonal of \( A \), and \(-L\) and \(-U\) are strictly lower and upper triangular elements of \( A \) respectively. It should be noted here that we assume the diagonal elements of matrix \( A \) do not vanish. Equation (3.1.1) then becomes,

\[
(D - L - U)u = b .
\]

By an assumption, \( D^{-1} \) exists, so we let,

\[
B = D^{-1}(L+U) \quad \text{and} \quad c = D^{-1}b .
\]

then (3.3.2) can be rewritten as,

\[
u = Bu + c .
\]

The Jacobi method is defined by,

\[
u^{(k+1)} = Bu^{(k)} + c , \quad k \geq 0 ,
\]

where \( u^{(k)} \) denotes the kth approximation to the solution.
Rewrite (3.3.5) in the form,
\[ u^{(k+1)} = u^{(k)} + (c - D^{-1}A) u^{(k)} \]  
(3.3.6)
then, the method of Simultaneous Displacement is defined by,
\[ u^{(k+1)} = u^{(k)} + \alpha D^{-1} (b - A) u^{(k)} \]  
(3.3.7)
where \( \alpha \) is a positive constant chosen to enhance convergence.

In these two methods the order in which one solves for the components \( u_i^{(k+1)} \) of \( u^{(k+1)} \) is of no consequence. In methods such as Gauss-Seidel method, written in matrix notation as,
\[ u^{(k+1)} = (I-L)^{-1} u^{(k)} + c \]  
(3.3.8)
or as,
\[ u^{(k+1)} = (I-L)^{-1} u^{(k)} + (I-L)^{-1} c \]  
(3.3.9)
where
\[ \hat{L} = D^{-1}L \]
\[ \hat{U} = D^{-1}U \]
we order the mesh points such that we use the LATEST estimates \( u_i^{(k+1)} \) of the components of \( u \) where available in determining a new estimate of a component of \( u^{(k+1)} \).

Related to the Gauss-Seidel iteration method is the Successive Over-relaxation method (SOR method). In matrix notation it is
\[ u^{(k+1)} = (1-\omega) u^{(k)} + \omega (L u^{(k+1)} + \hat{u}^{(k)} + c) \]  
(3.3.11)
or
\[ u^{(k+1)} = M u^{(k)} + (I-\omega \hat{L})^{-1} \omega c \]  
(3.3.12)
where
\[ M = (I-\omega \hat{L})^{-1} ((1-\omega) I + \omega \hat{U}) \]  
(3.3.12a)
and it is called the successive over-relaxation iteration matrix.
Here \( \omega \) is a parameter known as an over-relaxation factor. For \( \omega = 1 \), the S.O.R. method reduces to the Gauss-Seidel method.

A number of variations of the preceding techniques are possible.
For instance, among equations derived from finite difference approximations, there is a degree of symmetry over alternate grid points. (See Section 1.2).
For example, in a two-dimensional Laplace equation, each grid-point is coupled to its four neighbours; that is, each variable at an even-numbered grid point is coupled to variables at odd-numbered grid points, and vice-versa.
We now split \( u \) into \( u_{\text{even}} \) and \( u_{\text{odd}} \). Then a cyclic successive over-relaxation procedure can be defined as,

\[
\begin{align*}
\overline{u}^{(k+1)}_v &= \omega_{v} (L+U) \overline{u}^{(k)}_{1-v} + (1-\omega_{v}) \overline{u}^{(k-1)}_v + \omega_{v} c_v \quad (3.3.13)
\end{align*}
\]

where \( v \) assumes the values 0 and 1 over successive iterative steps. The 0's stand for "even" and the 1's stand for "odd".

When the relaxation parameter is changed every step, that is,

\[
\begin{align*}
\overline{u}^{(k+1)}_v &= \omega_{v} (L+U) \overline{u}^{(k)}_{1-v} + (1-\omega_{v}) \overline{u}^{(k-1)}_v + \omega_{v} c_v \quad (3.3.14)
\end{align*}
\]

where \( v=0 \) if \( i \) is even and \( v=1 \) if \( i \) is odd. The method is called Cyclic Chebyshev Method. The relaxation factor \( \omega_i \) is chosen such that

\[
\begin{align*}
\omega_0 &= 1 , \\
\omega_1 &= 1/(1-2\lambda^2_1) , \\
\omega_{k+1} &= 1/(1-2\lambda^2_i \omega_k) , \quad \text{for } k \geq 1
\end{align*}
\]

when \( \lambda_1 \) is the eigenvalue of largest modulus of the Jacobi matrix \( D^{-1}(L+U) \).

Notice that the first step corresponds to the Gauss-Seidel method and, therefore, \( \omega \) gradually increases.

Other iterative methods, more specialized than those discussed in this section have been developed and used successfully for elliptic equations. Among those are Symmetric Successive Overrelaxation (SSOR) method, Line Iterative method [Vemuri and Karplus, 1981] etc.

The method described so far belongs to the class of point iterative methods in which each component of \( u^{(n)} \) is expressed 'explicitly', i.e., can be expressed by itself using already computed approximate values of the other unknowns. Grouping the equations of the original system according to a predetermined rule we can obtain the BLOCK iterative methods. However, since we are concerned in this thesis with point iterative methods, no further discussion of the block iterative methods will be made.

Convergence of Point Iterative Methods

All of the point-iterative methods previously described can be written
in the form,
\[ u^{(k+1)} = Mu^{(k)} + d, \quad (3.3.16) \]
where \( M \) is known as the iteration matrix (of a particular method), and \( d \) is a column vector of constants.

**Definition 3.3.1**

The consistency condition (for an iterative process) is satisfied if, when the solution \( u \) of \( Au = b \) is substituted for \( u^{(k)} \) in (3.3.16), \( u^{(k+1)} \) is also the solution \( u \) of the system, i.e. once the solution is obtained, the iterative procedure makes no further modification of successive iterates.

**Definition 3.3.2**

An iterative method is said to be convergent if for all initial guess vectors \( u^{(0)} \), each component of the successive iterates \( u^{(k)} \) tends to the corresponding component of the solution \( u \) of \( Au = b \) for any given \( b \).

For convergence of an iterative process (3.3.16) we require that
\[ \lim_{k \to \infty} u^{(k)} = u = A^{-1}b, \]
and assuming the method consistent, i.e.,
\[ u = Mu + d. \quad (3.3.17) \]
Let the error in the \( k \)th iterated solution be given by,
\[ e^{(k)} = u^{(k)} - u, \quad (3.3.18) \]
we have from (3.3.16) and (3.3.17),
\[ e^{(k+1)} = Me^{(k)}, \quad (3.3.19) \]
whence,
\[ e^{(k)} = M^k e^{(0)}, \quad (3.3.20) \]
where \( e^{(0)} \) is the error vector associated with the initial vector \( u^{(0)} \). For convergence, we require,
\[ \lim_{k \to \infty} e^{(k)} = 0, \quad (3.3.21) \]
which, since \( e^{(0)} \) in (3.3.20) is arbitrary (being the error in an arbitrary initial approximation \( u^{(0)} \)), equation (3.3.21) can only be true if
\[ \lim_{r \to \infty} M^r = 0 \quad (the \ null \ matrix). \]
A necessary and sufficient condition for this to be so is that all the eigenvalues of $M$ are less than unity in absolute value.

By Theorem 1.8.2 this will be true if and only if $\rho(M) < 1$. We have proved the theorem below.

**Theorem 3.3.1**

A necessary and sufficient condition for convergence of an iterative method which can be expressed in the form of equation (3.3.16) is that $\rho(M) < 1$.

By assuming the condition (2.1.6) on the matrix $A$, it can be shown (see for example (Young 1962)) that the Jacobi and Gauss-Seidel methods ((3.3.5) and (3.3.8)) converged.

It is appropriate that we give two theorems which are related specifically to the Jacobi and Gauss-Seidel methods respectively.

**Theorem 3.3.2**

If $A$ is diagonally dominant, then $A$ is non-singular and the sequence $\{u^{(k)}\}$ defined by the Jacobi method (3.3.5) converges for any initial guess $u^{(0)}$.

**Theorem 3.3.3**

If $A$ is symmetric and positive-definite, then the sequence $\{u^{(k)}\}$ defined by Gauss-Seidel method (3.3.8) converges for any initial guess $u^{(0)}$.

For proofs of both the preceding theorems, see (Johnson and Reiss, 1982).

A convergence theorem for the point S.O.R. method (3.3.11) is as follows.

**Theorem 3.3.4**

If $A$ is symmetric and $a_{ii} > 0$, $i=1,2,\ldots,n$, then $\rho(M) < 1$ iff $A$ is positive-definite and $0 < \omega < 2$.

Proof: (see Young, 1971).
Rate of Convergence

The effectiveness of an iterative method is generally considered from both the computational work per iteration and the number of iterations required for convergence. We assume that an iterative method has converged when,

\[ \| e^{(k)} \| \leq \varepsilon \| e^{(0)} \| , \quad (3.3.21a) \]

where \( \| . \| \) denotes the \( \ell_2 \)-norm, \( e \) is an error vector, \( \varepsilon \) is a pre-assigned positive factor.

From (3.3.20) let \( M \) have eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_n \) and that \( |\lambda_1| > |\lambda_2| > \ldots > |\lambda_n| \) and assume that the corresponding eigenvectors \( v_1, v_2, \ldots, v_n \) are linearly independent. Then we can expand the initial error,

\[ e^{(0)} = \gamma_1 v_1 + \gamma_2 v_2 + \ldots + \gamma_n v_n , \quad (3.3.22) \]

where \( \gamma_i, i=1,2,\ldots,n \) are arbitrary scalars.

Thus,

\[ e^{(k)} = \gamma_1 \lambda_1^k v_1 + \gamma_2 \lambda_2^k v_2 + \ldots + \gamma_n \lambda_n^k v_n . \quad (3.3.23) \]

For \( k \) large, the expression (3.3.23) will be dominated by the eigenvalue of largest modulus. If we want to reduce the size of the error by \( 10^{-q} \), say, then \( k \) must be the least integer for which,

\[ |\lambda_1^k| = |\rho(M)|^k \leq 10^{-q} , \quad (3.3.24) \]

that is,

\[ k \geq \frac{-\log_{10} \rho(M)}{\log_{10} 10^q} \quad (3.3.25) \]

which shows that \( k \) decreases as \( -\log_{10} \rho(M) \) increases.

The positive quantity,

\[ R(M) = -\log_{10} \rho(M) , \quad (3.3.26) \]

is called the asymptotic rate of convergence of the iterative procedure (3.3.16). Its inverse is equal to the number of iterations needed (asymptotically) to reduce the residual error by a factor \( 10^{-q} \). We shall usually refer to \( R(M) \) simply as the rate of convergence.

Definition 3.3.3

We now define,

\[ R_k(M) = -\frac{1}{k} \log_{10} \| M^k \|_2 , \quad (3.3.27) \]
as the average rate of convergence of the iterative procedure (3.3.16).

Comparison of Rates of Convergence

An analysis of the conditions under which the S.O.R. methods are profitable over the basic Jacobi and Gauss-Seidel methods was discussed by Young (1954). In brief, Young's answer to the question of when over-relaxation pays is that it pays very well for a class of matrices which possess what he calls property (A) and are consistently ordered; the properties associated with many boundary value problems derived from elliptic difference operators. We will assume that these two properties hold in any future discussion on the S.O.R. method (unless otherwise stated).

We shall not discuss these concepts here, but simply state without proof the following important definitions and theorem and we shall after that summarise some important results.

Definition 3.3.4

A square matrix A of order n is said to be block-tridiagonal if it is of the form,

\[
A = \begin{bmatrix}
D_1 & U_1 & & \\
& D_2 & U_2 & \\
& & \ddots & \ddots \\
& & & D_n
\end{bmatrix}
\]

(3.3.28)

when the \( D_i, L_i, \) and \( U_i \) are square matrices of order \( m \geq 2 \).

Definition 3.3.5

If A is block-tridiagonal and if, moreover, each \( D_i \) is a diagonal matrix, then A will be called diagonally-block-tridiagonal.

Definition 3.3.6

A square matrix of order n is said to have property (A) if it can be transformed to diagonally-block-tridiagonal form by permutation of rows and column.
If A has property (A), there ordinarily exist more than one diagonally-block-tridiagonal form into which it can be transformed.

If \( \mu \) is an eigenvalue of the Jacobi matrix B (see (3.3.5)), \( \omega(\text{real}) > 0 \), and \( \lambda \) is an eigenvalue of \( M_\omega \) (see (3.3.12a)) then,

\[
\frac{\lambda + \omega - 1}{\omega \lambda^2} = \mu, \tag{3.3.29}
\]

It can be shown that

\[
\bar{\omega} = \frac{2}{1 + \sqrt{1 - \mu^2}} \quad [\bar{\mu} = \rho(B)]. \tag{3.3.30}
\]

(Varga, 1962), is the optimum value of \( \omega \) in the sense that \( \rho(M_\omega) \) is a minimum. It can be shown, further than,

\[
\rho(M_\omega) = \frac{1 - \sqrt{1 - \mu^2}}{1 + \sqrt{1 - \mu^2}} = \bar{\omega} - 1 \tag{3.3.31}
\]

From (3.3.29) and remembering that \( \omega = 1 \) gives the Gauss-Seidel method, we have,

\[
\rho(M_1) = [\rho(B)]^2, \tag{3.3.32}
\]

and so from (3.3.27),

\[
R(M_1) = 2R(B) \tag{3.3.33}
\]

This result shows that the rate of convergence of the Gauss-Seidel method is twice that of the Jacobi method.

By the use of (3.3.31) it can be verified that, asymptotically as \( \rho(B) \) tends to one,

\[
R(M_\omega) \approx 2\sqrt{R(M_1)}. \tag{3.3.34}
\]

We now investigate the rate of convergence of the method of simultaneous Displacements, given by (3.3.7). The optimum value \( \bar{\alpha} \) of \( \alpha \) in the sense that \( \rho(I - \alpha D^{-1}A) \) is minimised, is given by,

\[
\bar{\alpha} = \frac{2}{a + b} \tag{3.3.35}
\]

where we have assumed \( \Omega \geq \lambda \geq \lambda_i \geq \lambda_{\infty} (\lambda_i \) is an eigenvalue of \( D^{-1}A \)). \tag{3.3.36}

(see Forsythe and Wasow (1960), p.225). With this choice of \( \alpha \), the spectral radius of \( (I - \alpha D^{-1}A) \) is

\[
\max_{\lambda \in \text{sign}} |1 - \alpha \lambda_i| \leq \frac{b - a}{b + a} = \frac{b/a - 1}{b/a + 1} = \frac{P - 1}{P + 1}, \tag{3.3.37}
\]

where
Definition 3.3.7

$P = \frac{b}{a}$ is the $P$-condition number of $D^{-1}A$ and is defined as the ratio of the maximum eigenvalue to the minimum eigenvalue of a positive definite matrix.

From (3.3.37) and (3.3.26), the rate of convergence is given by,

$$R(I-cD^{-1}A) = \frac{2}{P}, \quad P \rightarrow \infty .$$

(3.3.38)

A general linear second degree method (those involving two previous iterates) is,

$$u^{(k+1)} = Gu^{(k)} + Hu^{(k-1)} + t .$$

(3.3.39)

Now (3.3.39) can be expressed as,

$$\begin{bmatrix} u^{(k)} \\ u^{(k+1)} \end{bmatrix} = \begin{bmatrix} O & I \\ H & G \end{bmatrix} \begin{bmatrix} u^{(k-1)} \\ u^{(k)} \end{bmatrix} + \begin{bmatrix} O \\ t \end{bmatrix} .$$

(3.3.40)

Let,

$$\hat{M} = \begin{bmatrix} O & I \\ H & G \end{bmatrix} .$$

(3.3.41)

then a necessary and sufficient condition for the convergence of (3.3.39) for all $u^{(0)}$ and $u^{(1)}$ is $\rho(\hat{M}) < 1$. A special case of (3.3.41) for the solution of $Au = b$ is Richardson's second degree method,

$$u^{(k+1)} = u^{(k)} + \alpha(cD^{-1}u^{(k)}) + \beta(u^{(k)} - u^{(k-1)}) .$$

(3.3.42)

The optimum value of $\alpha$ and $\beta$ are

$$\bar{\alpha} = \left(\frac{-2}{\sqrt{a+b}}\right)^2 \quad \text{and} \quad \bar{\beta} = \left(\frac{\sqrt{a+b}}{2}\right)^2 ,$$

(3.3.43)

where $a, b$ are defined by (3.3.36),

$$\rho(\hat{M}) = \sqrt{\beta} \quad \rho(\hat{M}) = \sqrt{\beta} .$$

(3.3.44)

and the rate of convergence is,

$$R = \frac{2}{\sqrt{\beta}} , \quad \text{as} \quad P \rightarrow \infty .$$

(3.3.45)

We give various quantities described above for the Laplace two-dimensional model problem given in Section 2.2. We have from (3.3.36) and (2.2.7),

$$0 < a = 2 \sin^2 \frac{\pi}{2N} \leq \lambda_{i,j} = \sin^2 \left(\frac{\pi}{2N}\right) + \sin^2 \left(\frac{j\pi}{2N}\right) \leq b = 2 \cos^2 \left(\frac{\pi}{2N}\right) , \quad 1 \leq i,j \leq N-1 .$$

(3.3.46)
where \((N-1)^2\) is the order of the matrix \(A\). From (3.3.5) we have,
\[
\overline{u} = \rho(B) = \cos \left( \frac{\pi}{N} \right).
\]

Hence, from (3.3.30),
\[
\overline{u} = \frac{2}{1 + \sin \left( \frac{\pi}{N} \right)} = \frac{2}{1 + \sin (\pi h)} \quad [h = 1/N]
\]

Thus, from (3.3.31), (3.3.32), (3.3.33) and (3.3.47) we obtain,
\[
\begin{align*}
(i) \quad R(B) &= \frac{2}{2N} \\
(ii) \quad R(M_1) &= \frac{\pi}{N} \\
(iii) \quad R(M_\infty) &= \frac{2\pi}{N}
\end{align*}
\]

We now give some useful theorems on the convergence of various iteration schemes.

**Theorem 3.3.5 (Young, 1971)**

If the Jacobi method converges, then the method of simultaneous displacement method (JOR) converges for \(0 < \omega < 1\).

**Theorem 3.3.6 (Kahan, 1958)**

\[
\rho(M_\infty) \geq |\omega - 1|.
\]

Moreover, if SOR method converges, then,
\[
0 < \omega < 2.
\]

**Theorem 3.3.7 (Young, 1971)**

Let \(A\) be an irreducible matrix with weak diagonal dominance. Then,

a) The Jacobi method, and the method of simultaneous displacement converges for \(0 < \omega < 1\).

b) Both the Gauss-Seidel and the SOR methods converge for \(0 < \omega < 1\).

**Theorem 3.3.8 (Young, 1971)**

If \(A\) is symmetric, positive definite and is consistently ordered and if
\[ \bar{\omega} = \frac{2}{1 + (1 - \bar{\mu})^{1/2}} \]

where \( \bar{\mu} \) is the spectral radius of \( B \). Then,

\[ \rho(M_{\omega}) < \rho(M_{\bar{\mu}}) \quad \omega \neq \bar{\omega} \quad (3.3.50) \]

and

\[ \rho(M_{\omega}) = \omega - 1 \text{ for } \bar{\omega} < \omega < 2. \quad (3.3.51) \]
3.4 DESCENT METHODS

The iterative methods described in Section 3.3 all involve the assumption of an arbitrary solution distribution followed by the solution of each of the \( n \) equations, comprising the system of difference equations for one of the unknowns. There results a new set of estimates for the solution at each of the grid points, which is then used to obtain even better estimates, and so on. The descent method begins likewise with the assumption of an arbitrary solution distribution. These values are inserted directly into

\[
Au = b, \quad (3.4.1)
\]

and a calculation is made as to the extent to which the left-hand side of (3.4.1) differs from the right-hand side. An error term is thus calculated for each of the simultaneous equations. The procedure is repeated until convergence has been met.

The system (3.4.1) can be written as the gradient of a quadratic function,

\[
F(u) = \frac{1}{2}(u, Au) - (u, b). \quad (3.4.2)
\]

Thus, the process of finding the solution of (3.4.1) is identical to finding the vector \( u \) which minimizes \( F(u) \) where \( F(u) \) defines an ellipsoid in the \( n \)-dimensional space of the elements of \( u \), whose common centre is \( A^{-1}b \). For an arbitrary vector \( u^{(i)} \), the residual \( r^{(i)} \) is given by,

\[
r^{(i)} = b - Au^{(i)} = -\text{Grad } F(u^{(i)})^*, \quad (3.4.3)
\]

and it is always normal to the surface of the ellipsoid defined by (3.4.2).

We can represent this process by the non-stationary iterative scheme,

\[
u^{(k+1)} = u^{(k)} + \alpha^*_k p^{(k)}, \quad (3.4.4)
\]

where \( \alpha_k \) is a scalar and \( p^{(k)} \) a vector as yet to be defined. The problem now is to determine \( \alpha_k \) such that the quadratic function \( F(u^{(k+1)}) \) will be minimum for a given direction \( p^{(k)} \).

Using (3.4.2) and (3.4.4) we have,

\[
F(u^{(k+1)}) = \frac{1}{2}(u^{(k)} + \alpha^*_k p^{(k)}), A(u^{(k)} + \alpha^*_k p^{(k)})) - (u^{(k)} + \alpha^*_k p^{(k)}), b) \quad (3.4.5)
\]

\[
\text{Grad } F(u^{(i)}) = \frac{\partial F(u^{(i)})}{\partial u_j}, j=1,2,...,n.
\]
therefore,  
\[
\frac{\partial F(u^{(k+1)})}{\partial a_k} = (p^{(k)}, \Delta \psi^{(k)} + a_k \psi^{(k)}) - (p^{(k)}, b) = -(p^{(k)}, \Delta \psi^{(k)}) + (a_k \psi^{(k)}, a_k \psi^{(k)}).
\]  
(3.4.6)

The optimum value of \( a_k \) is obtained when the expression (3.4.6) is equal to zero, which gives,
\[
a_k = \frac{(p^{(k)}, \Delta \psi^{(k)})}{(p^{(k)}, \Delta \psi^{(k)})}.
\]
(3.4.7)

The above choice of \( a_k \) systematically reduces \( F(u^{(k+1)}) \) and the method converges for any given \( p^{(k)} \).

From (3.4.4) and (3.4.7) we have,
\[
(p^{(k)}, \Delta \psi^{(k+1)}) = (p^{(k)}, (b - \Delta \psi^{(k+1)})) = (p^{(k)}, (\Delta \psi^{(k)} - a_k \Delta \psi^{(k)})) = 0,
\]
(3.4.8)
which shows that the direction \( p^{(k)} \) and the residual \( \Delta \psi^{(k+1)} \) are orthogonal.

The best known and most general of the descent methods employing relations (3.4.4) is known as the method of steepest decent. In this technique, the correction term \( a_k \psi^{(k)} \) is given the value that will have the most pronounced minimizing effect upon the error. This is achieved by simply choosing \( p^{(k)} = -\Delta \psi^{(k)} \). Hence, the method of steepest descent (3.4.4) takes the explicit form,
\[
u^{(k+1)} = u^{(k)} + \Delta \psi^{(k)}, \Delta \psi^{(k)} = \frac{(\Delta \psi^{(k)}, \Delta \psi^{(k)})}{(\Delta \psi^{(k)}, \Delta \psi^{(k)})} \Delta \psi^{(k)}.
\]
(3.4.9)

Introducing the function,
\[
E(u) = \frac{1}{2}((u - u^*), A(u - u^*))
\]
where \( u^* \) is the minimum point sought, we have \( E(u) = F(u) + \frac{1}{2}(u^* A u^*) \), which shows that the function \( E \) differs from \( F \) only by a constant. It will be convenient to consider that we are minimizing \( E \) rather than \( F \).

**Lemma 3.4.1**

The iterative process (3.4.9) satisfies,
\[
E(u^{(k+1)}) = \left(1 - \frac{(\Delta \psi^{(k)}, \Delta \psi^{(k)})^2}{(\Delta \psi^{(k)}, \Delta \psi^{(k)}) (\Delta \psi^{(k)}, A^{-1} \Delta \psi^{(k)})} \right) E(u^{(k)}).
\]
(3.4.10)
Proof: We have, setting \( z^{(k)} = u^{(k)} - u^* \),

\[
\frac{E(u^{(k)}) - E(u^{(k+1)})}{E(u^{(k)})} = 2 \lambda_k \frac{(z^{(k)}, A^{-1} z^{(k)}) - \lambda_k (z^{(k)}, A z^{(k)})}{(z^{(k)}, A z^{(k)})}
\]

using \( z^{(k)} = A z^{(k)} \), we have,

\[
\frac{E(u^{(k)}) - E(u^{(k+1)})}{E(u^{(k)})} = \frac{2(z^{(k)}, z^{(k)})^2 - (z^{(k)}, z^{(k)})^2}{(z^{(k)}, A^{-1} z^{(k)}) (z^{(k)}, A z^{(k)})}
\]

\[
= \frac{(z^{(k)}, z^{(k)})^2}{(z^{(k)}, A^{-1} z^{(k)}) (z^{(k)}, A z^{(k)})} \tag{3.4.11}
\]

Kantorovich inequality:

Let \( A \) be a positive definite symmetric \( n \times n \) matrix. For any vector \( u \) there holds,

\[
\frac{(u, u)^2}{(u, Au) (u, A^{-1} u)} \geq \frac{4ab}{(a+b)^2}, \tag{3.4.12}
\]

where \( a \) and \( b \) are the smallest and the largest eigenvalue of \( A \) respectively.

Proof: (see Luenberger, 1973).

Combining (3.4.11) and (3.4.12) we obtain the central result on the convergence of the method of the steepest descent.

Theorem 3.4.1

For any \( u^{(0)} \in \mathbb{R}^n \) the method of steepest descent (3.4.9) converges to the unique minimum point \( u^* \) of \( F \). Furthermore, with \( E(u) = \frac{1}{2}(u-u^*), A(u-u^*) \), there holds at every step \( k \),

\[
E(u^{(k+1)}) \leq \left(1 - \frac{4ab}{(b+a)^2}\right)^2 E(u^{(k)}) = \left(\frac{b-a}{b+a}\right)^2 E(u^{(k)}) \tag{3.4.13}
\]

Proof: By Lemma 3.4.1 and Kantorovich's inequality,

\[
E(u^{(k+1)}) \leq \left\{1 - \frac{4ab}{(b+a)^2}\right\} E(u^{(k)}) = \left(\frac{b-a}{b+a}\right)^2 E(u^{(k)})
\]

It follows immediately that \( E(u^{(k)}) \to 0 \) and hence, since \( A \) is positive definite, that \( u^{(k)} \to u^* \).
It should be noted that the convergence ratio is
\[
\frac{b-a}{b+a} = \left(\frac{p-1}{p+1}\right)^2
\]
which clearly shows that convergence is slow.

Its application to finite difference equations has been limited because of their relatively unfavourable convergence properties. The solution approaches the correct solution rather rapidly at first, but then tends to oscillate about the final value (Vemuri & Karplus, 1981).

**Conjugate Direction**

A better strategy for choosing the direction \( p^{(k)} \) is based on the knowledge that the centre of the ellipsoid lies in the plane conjugate to the given chord. In two dimensions, the level curves \( F(u) = K \), for different values of \( k \), are concentric ellipses (Figure 3.4.1). Suppose \( C \) is the optimal point. The minimum from the point \( A \) in the direction \( AD \) occurs at \( B \). We say that the direction \( BC \) is conjugate to the direction \( AD \), since, for any ellipse \( F(u) = K \), the diameter through \( B \) is conjugate (in the usual geometrical sense) to the diameter parallel to \( AD \). The idea of conjugate directions can be extended to \( n \) dimensions by the following.

**Definition 3.4.1**

Given a symmetric matrix \( A \), two vectors \( p_1 \) and \( p_2 \) are said to be \( A \)-orthogonal, or conjugate, with respect to \( A \) if \( \langle p_1, Ap_2 \rangle = 0 \). (The phrase 'with respect to \( A \)', will be omitted where there is no ambiguity).

Conjugate direction methods, especially the method of conjugate gradients, have proved to be extremely effective in dealing with general objective functions. Conjugate gradient methods have recently gained acceptance as methods for solving finite difference/element equations. The necessary extensions have become known as preconditioned conjugate gradient methods which will be discussed in Chapter 7. We begin, however, with the standard
FIGURE 3.4.1

Conjugate Directions
conjugate direction method. The important properties of conjugate directions are established in the following theorem.

**Theorem 3.4.2**

If the vectors $\mathbf{p}_j$ are mutually conjugate, then they are linearly independent.

**Proof:**

Suppose that,

$$\sum_j a_j \mathbf{p}_j = 0$$

for some scalar $j$. If $k$ is any one of the values of $j$, then

$$(\mathbf{p}_k^T, A \sum_j a_j \mathbf{p}_j) = 0$$

i.e.

$$a_k \mathbf{p}_k^T \mathbf{p}_k = 0$$

Now $A$ is positive definite and $\mathbf{p}_k \neq 0$, hence $a_k = 0$.

Corresponding to the $n \times n$ positive definite matrix $A$ let $\mathbf{p}^{(0)}, \mathbf{p}^{(1)}, \ldots, \mathbf{p}^{(n-1)}$ be $n$ non-zero conjugate vectors. Then the solution vector $u^*$ can be expressed as,

$$u^* = a_0 \mathbf{p}^{(0)} + a_1 \mathbf{p}^{(1)} + \ldots + a_{n-1} \mathbf{p}^{(n-1)}.$$  (3.4.14)

Multiplying by $A$ and then taking the scalar product with $\mathbf{p}_k$ yields directly,

$$a_k = \frac{(\mathbf{p}_k^T, A u^*)}{(\mathbf{p}_k^T, A \mathbf{p}_k)} = \frac{(\mathbf{p}_k^T, \mathbf{p})}{(\mathbf{p}_k^T, A \mathbf{p}_k)}$$  (3.4.15)

This shows that the $a_k$'s and consequently the solution $u^*$ can be found by evaluation of simple scalar products. The end result is,

$$u^* = \sum_{j=0}^{n-1} \frac{(\mathbf{p}_j^T, \mathbf{p})}{(\mathbf{p}_j^T, A \mathbf{p}_j)} \mathbf{p}_j.$$  (3.4.16)

The expansion for $u^*$ can be considered to be the result of an iterative process of $n$ steps where at the kth step $a_k \mathbf{p}_k$ is added. The basic conjugate direction method is obtained.
Theorem 3.3.3

Let \( \{ \mathbf{p}^{(j)} \} \), \( j=0,1,\ldots,n-1 \), be a set of non-zero conjugate vectors. For any \( \mathbf{u}^{(0)} \in \mathbb{R}^n \) the sequence \( \mathbf{u}^{(k)} \) generated according to,
\[
\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \alpha_k \mathbf{p}^{(k)} , \quad k \geq 0 ,
\]
with,
\[
\alpha_k = \frac{\langle \mathbf{r}^{(k)} , \mathbf{p}^{(k)} \rangle}{\langle \mathbf{p}^{(k)} , \mathbf{A} \mathbf{p}^{(k)} \rangle} ,
\]
and
\[
\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A} \mathbf{u}^{(k)} .
\]
converges to the unique solution \( \mathbf{u}^* \), of \( \mathbf{A} \mathbf{u} = \mathbf{b} \) after \( n \) steps, i.e. \( \mathbf{u}^{(n)} = \mathbf{u}^* \).

Proof:

We can write,
\[
\mathbf{u}^* - \mathbf{u}^{(0)} = \alpha_0 \mathbf{p}^{(0)} + \alpha_1 \mathbf{p}^{(1)} + \cdots + \alpha_{n-1} \mathbf{p}^{(n-1)} ,
\]
for some \( \alpha_k \)'s. As we get (3.4.15), thus,
\[
\alpha_k = \frac{\langle \mathbf{r}^{(k)} , \mathbf{A} (\mathbf{u}^* - \mathbf{u}^{(0)}) \rangle}{\langle \mathbf{p}^{(k)} , \mathbf{A} \mathbf{p}^{(k)} \rangle} .
\]
From (3.4.20), we have,
\[
\mathbf{u}^{(k)} - \mathbf{u}^{(0)} = \alpha_0 \mathbf{p}^{(0)} + \alpha_1 \mathbf{p}^{(1)} + \cdots + \alpha_{k-1} \mathbf{p}^{(k-1)} ,
\]
and hence by the conjugency of the \( \mathbf{p}^{(k)} \)'s, it follows that,
\[
\langle \mathbf{p}^{(k)} , \mathbf{A} (\mathbf{u}^{(k)} - \mathbf{u}^{(0)}) \rangle = 0 .
\]
Substitution (3.4.23) in (3.4.21) produces,
\[
\alpha_k = \frac{\langle \mathbf{p}^{(k)} , \mathbf{A} (\mathbf{u}^* - \mathbf{u}^{(0)}) \rangle}{\langle \mathbf{p}^{(k)} , \mathbf{A} \mathbf{p}^{(k)} \rangle} = \frac{\langle \mathbf{r}^{(k)} , \mathbf{p}^{(k)} \rangle}{\langle \mathbf{p}^{(k)} , \mathbf{A} \mathbf{p}^{(k)} \rangle}
\]
which is identical with (3.4.18). 

To generate a sequence of \( \mathbf{A} \)-conjugate directions from the \( \mathbf{p}^{(j)} \)'s, \( j=0,1, \ldots,n-1 \), we use a Gram-Schmidt procedure...

The Conjugate Gradient Method

The conjugate gradient method is a conjugate direction method that is obtained by selecting the successive direction vectors as a conjugate version of the successive gradients obtained as the method progresses. Thus, the directions are not specified beforehand, but rather are determined sequentially...
at each step of the iteration. At step k we evaluate the current negative gradient vector and add to it a linear combination of the previous direction vectors to obtain a new conjugate direction vector along which to move.

The complete algorithm is as follows:

(i) Let \( k=0 \), \( \mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A} \mathbf{u}^{(0)} \) and \( \mathbf{p}^{(0)} = \mathbf{r}^{(0)} \).

For \( k=0,1,2,\ldots \), compute the vectors \( \mathbf{u}^{(k)}, \mathbf{r}^{(k)} \) and \( \mathbf{p}^{(k)} \) from

(ii) \( \mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \alpha_k \mathbf{p}^{(k)} \)

where

\[
\alpha_k = \frac{\mathbf{r}^{(k)} \cdot \mathbf{r}^{(k)}}{\mathbf{p}^{(k)} \cdot \mathbf{A} \mathbf{p}^{(k)}}
\]

(iii) \( \mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} + \alpha_k \mathbf{A} \mathbf{p}^{(k)} \)

(iv) \( \mathbf{p}^{(k+1)} = \mathbf{r}^{(k+1)} + \beta_k \mathbf{p}^{(k)} \)

where

\[
\beta_k = \frac{\mathbf{r}^{(k)} \cdot \mathbf{r}^{(k+1)}}{\mathbf{r}^{(k)} \cdot \mathbf{r}^{(k)}}
\]

(3.4.24)

It can be proved (Mitchell & Griffith, 1980) that,

\[
(\mathbf{r}^{(k+1)} \cdot \mathbf{r}^{(k)}) = 0,
\]

and

\[
(\mathbf{p}^{(k+1)} \cdot \mathbf{A} \mathbf{p}^{(j)}) = 0 , \ (j=0,1,2,\ldots,k).
\]

There are three primary advantages to this method of direction selection.

a. Unless the solution is attained in less than \( n \) steps, the gradient is always non-zero and linearly independent of all previous direction vectors.

b. It is especially a simple formula that is used to determine the new direction vector. This simplicity makes the method only slightly more complicated than the steepest descent method.

c. Because the directions are based on the gradients, the process makes good uniform progress toward the solution at every step.

Although the Conjugate Gradient method theoretically gives an exact answer in not more than \( n \)-iterative steps, however, when solving large systems and rounding errors are present in the calculation, the mutual conjugency
of the sequence $p^{(0)}, p^{(1)}, \ldots$ become degraded, and, consequently, so does the finite termination property. If the obtained solution $u^{(k+1)}$ becomes too inaccurate because of round-off errors then the iteration is either terminated if $x^{(k)}$ is sufficiently small or could be restarted with $u^{(k+1)}$ as initial guess.

The algorithm requires the storage of only the non-zero element of $A$ together with the four vectors $u^{(k)}, \bar{x}^{(k)}, \bar{p}^{(k)}$ and $A \bar{p}^{(k)}$. This is greater than the storage of SOR, for instance, but may be offset by the fact that the parameters $\alpha_k, \beta_k$ are determined automatically, unlike SOR where the optimum relaxation parameter may have to be estimated.

It has been pointed out by Reid (1972) that when $A$ is structured so that,

$$A = \begin{bmatrix} I_1 & F \\ F^T & I_2 \end{bmatrix}$$

where $I_1$ and $I_2$ are unit matrices and $F$ is an arbitrary rectangular matrix, then the work per iteration can be approximately halved.

Further discussion on this class of methods will be presented in Chapter 7.
CHAPTER 4

GENERAL THEORY OF PRECONDITIONING
4.1 INTRODUCTION

Suppose we are required to solve a linear system of equations given in matrix form, by,

\[ Au = b, \]  

(4.1.1)

where \( A \) and \( b \) are known. We are concerned with the solution of (4.1.1) arising from the discretisation by the finite difference/element methods of a boundary value problem involving elliptic P.D.E.'s as discussed in Chapter 2 and from the variational methods to be described in Chapter 5. In this chapter, we will give the applications of preconditioning techniques to both the direct methods and iterative methods. We begin, however with the theory of rounding errors that occurred in the computational process by the elimination method and other direct methods of solution.
4.2 ERRORS IN COMPUTATION

Computer Representation of Numbers

A number is held in a computer in binary form, as a string of 0's and 1's (a 'word'). In ICL 1900 series FORTRAN (Loughborough University) an integer is held in a 24-bit word, which allows integers in the range

\[-838860 \leq I \leq 838859\]  

Integer arithmetic is carried out exactly as long as the integers produced do not become too large ('overflow') or too small ('underflow').

Numbers not held as integers are referred to as real numbers. These may be held in either "fixed-point form" or in "floating-point form". For example, the fraction 1/16 may be written as a decimal either as 0.0625 (fixed-point) or as 0.625\times10^{-1} (floating point); the corresponding binary representations are 0.0001 and 0.1\times2^{-3} respectively. In fixed-point form the computation must be framed so that every computed number \(u\) lies in the range,

\[-1 \leq u < 1.\]  

If a fixed point number is held in a word of length \(t\) bits (binary digits), then it may be held with an error not exceeding

\[|\frac{1}{2} \times 2^{-t}| = 2^{-(t+1)}\]  

Floating Point Numbers

In this form, each number \(u\) is represented by an ordered pair \(a\) and \(b\) such that \(u=ax^b\); \(a\) is called the mantissa and \(b\) the exponent.

Mantissa: The mantissa is restricted to

\[\frac{1}{2} \leq |a| < 1.\]  

Exponent: The exponent is an integer and is therefore held exactly, provided it is not too large. In 1900 FORTRAN, 9 bits are used to store \((256+b)\), and so \(b\) lies in the range \(-256 \leq b < 256\).

The range of values of \(u\) is therefore: \(2^{255} > |u| \geq \frac{1}{2} \times 2^{-256}\), which is approximately \(2.9\times10^{75} > |u| \geq 8.6\times10^{-77}\). Provided \(|u|\) lies in this
range there is no inaccuracy due to the exponent.

We can express the computer representation of \( u \) in the form,
\[
\hat{f}(u) = u(1 + \varepsilon), \quad \text{where } |\varepsilon| \leq 2^{-t}
\] (4.2.5)

In 1900 FORTRAN we have \( t=37 \), so that \( u \) is held with a relative accuracy of approximately 11 decimal digits \( (2^{-5} \approx 2^{-37} \approx 7 \times 10^{-12}) \).

Mainly we are interested in the error in floating-point arithmetic occurring when we compute an inner product as,
\[
\sum_{i=1}^{n} u_i v_i + u_2 v_2 + \ldots + u_n v_n
\] (4.2.6)
We begin however with operations of multiplication (division) and addition (subtraction).

**Multiplication and Division**

If \( \hat{f}(u_1) = a_1 b_1 \) and \( \hat{f}(u_2) = a_2 b_2 \)

then we form \( (a_1 a_2) (b_1 b_2) \) by multiplying together the mantissa and adding together the exponents. This product satisfies,
\[
\frac{1}{2} \leq |a_1 a_2| < 1
\] (4.2.8)
and is therefore normalised if necessary by a shift to the left, the exponent being adjusted accordingly. This process may be illustrated by a simple example of multiplication of numbers in 4-digit floating point decimal arithmetic,
\[
10^{-4} (0.8732) \times 10^6 (0.6135) = 10^2 (0.49989820)
\]
Rounded to give 4-digit mantissa, we have the computed product of \( 10^2 (0.4989) \) with the fractional error introduced is \( 0.0000018 \) or \( 0.18 \times 10^{-5} \).

In general, the resulting \( 2t \)-digit product is then rounded to give the \( t \)-digit mantissa of the computed product with the error not greater than \( 2^{-t} \) in modulus, and we can write,
\[
\hat{f}(u_1 u_2) = u_1 u_2 (1 + \varepsilon), \quad \text{where } |\varepsilon| \leq 2^{-t}
\] (4.2.9)
Similar remarks apply to division:

\( \varepsilon \) is relative error and is obtained by \( 2^{-(t+1)} / 4 \).
Addition and Subtraction

Without loss of generality, we assume that $b_1$ in (4.27) is greater than $b_2$. Then, the integer $(b_1 - b_2)$ is computed. We suppose further that $b_1 - b_2 < t$. (For $b_1 - b_2 > t$, $fi(u_1 + u_2) = u_1$.) Then,

$$a_1^2 + a_2^2 = (a_1 + a_2^2)^2 = (b_1 - b_2)^2,$$

(4.2.11)

where $a_1 > 1$ and hence $a_2^2 = (b_1 - b_2)^2$ is a proper fraction. The sums of (4.2.11) is then calculated exactly and requires less than $2t+1$ digits for its representation. Thus, the mantissa of $fi(u_2)$ is moved to the right $(b_1 - b_2)$ places and added to $a_1$. If necessary, the result is then shifted so that the resulting number lies in the range permitted for the mantissa and $b_1$ is adjusted accordingly. Finally, this $2t$-digit mantissa is rounded to $t$ digits. The resulting mantissa is correct to $t$ bits and a final relative error of not more than $2^{-t}$. We may write

$$fi(u_1 + u_2) = (u_1 + u_2)(1 + \varepsilon), \quad |\varepsilon| < 2^{-t},$$

(4.2.12)

As an example, we have:

$$10^4 (0.6314) + 10^1 (0.3865).$$

$a_2$ is shifted 3 places to the right and addition takes place in 8-figure arithmetic,

$$10^4 \times 0.63140000$$

$$+ 10^4 \times 0.00038650$$

$$10^4 \times 0.63178650$$

The exact sum is rounded to give $10^4 (0.6318)$.

The results for subtraction are analogous with those for addition.

Extended Arithmetic

We consider first the extended product $p_n$. We have,

$$fi(u_1 u_2) = u_1 u_2 (1 + \varepsilon_2), \quad |\varepsilon_2| < 2^{-t}$$

(4.2.13)
\[ fl(u_1 u_2 u_3) = u_1 u_2 (1 + \varepsilon_2) u_3 (1 + \varepsilon_3) , \quad |\varepsilon_i| \leq 2^{-t} , \quad i=2,3 \] (4.2.14)

and so on. Hence we have,

\[ p_n = u_1 u_2, \ldots, u_n (1 + \varepsilon_2)(1 + \varepsilon_3), \ldots, (1 + \varepsilon_n) , \] (4.2.15)

where \( |\varepsilon_i| \leq 2^{-t} \) (i=2,3,...,n). Equation (4.2.15) can be written as

\[ fl(u_1 u_2, \ldots, u_n) = u_1 u_2, \ldots, u_n (1 + E) , \] (4.2.16)

where,

\[ (1 - 2^{-t})^{n-1} \leq 1 + E \leq (1 + 2^{-t})^{n-1} . \] (4.2.17)

For extended sequences of multiplication and addition, we have,

\[ fl(u_1 u_2, \ldots, u_m/v_1 v_2, \ldots, v_n) = (u_1 u_2, \ldots, u_m/v_1 v_2, \ldots, v_n) (1 + E) \] (4.2.18)

where,

\[ (1 - 2^{-t})^{m+n-1} \leq 1 + E \leq (1 + 2^{-t})^{m+n-1} . \] (4.2.19)

Provided (m+n) is small compared to \( 2^t \), \( (2^{30} \approx 10^{10} \), and so is likely), we can write, with negligible error,

\[ |E| \leq (m+n-1)2^{-t} , \] (4.2.20)

the error bounds being proportional to the number of operations carried out.

**Additions**

We now give the analysis when accurate (double-length) additions are used. We have,

\[ fl(u_1 + u_2) = (u_1 + u_2) (1 + \varepsilon_2) , \quad |\varepsilon_2| \leq 2^{-t} \] (4.2.21)

\[ fl(u_1 + u_2 + u_3) = [(u_1 + u_2) (1 + \varepsilon_2) + u_3 (1 + \varepsilon_3) \] (4.2.22)

\[ = u_1 (1 + \varepsilon_2) (1 + \varepsilon_3) + u_2 (1 + \varepsilon_2) (1 + \varepsilon_3) + u_3 (1 + \varepsilon_3) \] (4.2.23)

and,

\[ fl(u_1 + u_2 + \ldots + u_n) = (1 + \eta_1) u_1 + (1 + \eta_2) u_2 + \ldots + (1 + \eta_n) u_n \] (4.2.24)

where,

\[ 1 + \eta_1 = (1 + \varepsilon_2)(1 + \varepsilon_3) \ldots (1 + \varepsilon_n) \] (4.2.25)

and

\[ 1 + \eta_r = (1 + \varepsilon_r)(1 + \varepsilon_{r+1}) \ldots (1 + \varepsilon_n) \quad \text{for } r>1 \] (4.2.26)

\[ (1 + 2^{-t})^{n-1} \leq 1 + \eta_1 \leq (1 + 2^{-t})^{n-1} \] (4.2.27)

\[ (1 + 2^{-t})^{n+1-r} \leq 1 + \eta_r \leq (1 + 2^{-t})^{n+1-r} \quad \text{for } r>1. \] (4.2.28)

With negligible error we may use equation (4.2.28) for \( r=1 \) as well as for \( r>1 \). If, in addition, \( n \) is small compared with \( 2^t \), then,
In general the inequalities of the form,
\[(1-2^{-t})^s \leq 1+\varepsilon \leq (1+2^{-t})^s,\]
can be replaced by the simpler inequality,
\[|\varepsilon| \leq s2^{-t} \text{ (see Wilkinson, 1965)} \tag{4.2.30a}\]
From equation (4.2.28) we note that the bounds themselves are dependent on the order of summation. The upper bound for the error is smallest if the terms are added in order of increasing absolute magnitude.

**Floating Point Accumulation of Inner-Products**

We now consider the error made when an inner-product is computed in floating-point using a) single precision arithmetic and b) double precision accumulator.

a) **Single-precision**

Let
\[s_n = \text{fl}(u_1 v_1 + u_2 v_2 + \ldots + u_n v_n)\]  \tag{4.2.31}

We define,
\[t_r = \text{fl}(u_r v_r)\]  \tag{4.2.32}

We have then the relation
\[s_1 = t_1,\]  \tag{4.2.33}
\[s_r = \text{fl}(s_{r-1} + t_r)\]  \tag{4.2.34}

From equations (4.2.9) and (4.2.12), we deduce,
\[t_r = u_r v_r (1+\rho_r), \quad |\rho_r| \leq 2^{-t},\]  \tag{4.2.35}
\[s_r = (s_{r-1} + t_r) (1+\eta_r), \quad |\eta_r| \leq 2^{-t}.\]  \tag{4.2.36}

Hence,
\[s_n = u_1 v_1 (1+\varepsilon_1) + u_2 v_2 (1+\varepsilon_2) + \ldots + u_n v_n (1+\varepsilon_n)\]  \tag{4.2.37}

where,
\[1+\varepsilon_1 = (1+\rho_1) (1+\eta_1) \ldots (1+\eta_n),\]  \tag{4.2.38}

and
\[1+\varepsilon_r = (1+\rho_r) (1+\eta_r) \ldots (1+\eta_n) \text{ for } r = 2, 3, \ldots, n\]  \tag{4.2.39}

Hence,
\[(1-2^{-t})^n \leq 1+\varepsilon_1 \leq (1+2^{-t})^n\]  \tag{4.2.40}

and
\[(1-2^{-t})^{n-r+2} \leq 1+\varepsilon_r \leq (1+2^{-t})^{n-r+2}, \text{ for } r = 2, 3, \ldots, n.\]  \tag{4.2.41}

It is obvious that,
\[(1+2^{-t})^n \leq (1+2^{-t})^{n+1}, \quad (4.2.42)\]
\[(1-2^{-t})^n \geq (1-2^{-t})^{n+1}, \quad (4.2.43)\]
so we can generally write,
\[s_n = u_1 v_1 (1+\varepsilon_1) + u_2 v_2 (1+\varepsilon_2) + \ldots + u_n v_n (1+\varepsilon_n) \quad (4.2.44)\]
where,
\[(1-2^{-t})^{-n+2} \leq 1+\varepsilon_r \leq (1-2^{-t})^{-n+2}, \text{ for } n=1,2,\ldots,n \quad (4.2.45)\]
From (4.2.30a), we can replace (4.2.45) by,
\[|\varepsilon_r| \leq (n-r+2)2^{-t} \quad (4.2.45a)\]

b) Double Precision

On a computer with a double-precision accumulator the computed sum of two floating-point numbers always has a low relative error. On a computer without such an accumulator, the floating point operations are usually a little less satisfactory. The error bounds obtained for a computer using a single-precision accumulator will, as a rule, be no more than \(10^{-1}\) times (or \(10^{-10}\) for decimal) times those for a computer with a double-precision accumulator (Wilkinson, 1963, p.13) so that we have,
\[\phi_l(u_1 + u_2) \equiv (u_1 + u_2) (1+\varepsilon) \quad |\varepsilon| \leq (10^{-1})2^{-t} \quad (4.2.46)\]
We now consider briefly the errors made when using a double-precision accumulator. We assume that we have only a double-precision accumulator (not quadruple). Let,
\[s_n = \Phi_{\phi_2}(u_1 v_1 + u_2 v_2 + \ldots + u_n v_n), \quad (4.2.47)\]
where \(u_i, v_i, i=1,2,\ldots,n\) are standard single-precision numbers (t-digits).
We have in much the same way,
\[s_n = u_1 v_1 (1+\varepsilon_1) + u_2 v_2 (1+\varepsilon_2) + \ldots + u_n v_n (1+\eta_n) \quad (4.2.48)\]
where,
\[1+\varepsilon_1 = (1+p_1)(1+\eta_1), \ldots, (1+p_n)(1+\eta_n) \quad (4.2.49)\]
and\[1+\varepsilon_r = (1+p_r)(1+\eta_r), \ldots, (1+p_n)(1+\eta_n), \text{ for } r=2,3,\ldots,n \quad (4.2.50)\]
but now,
\[|\varepsilon_r| \leq \frac{3}{2}2^{-2t} \quad (4.2.51)\]
\[|\eta_r| \leq \frac{3}{2}2^{-2t} \quad (4.2.52)\]
We have,
\[
(1 - \frac{3}{2} 2^{-2t})^n \leq 1 + \varepsilon_1 \leq (1 + \frac{3}{2} 2^{-2t})^n
\]  
(4.2.53)
\[
(1 - \frac{3}{2} 2^{-2t})^{n+2-r} \leq 1 + \varepsilon_r \leq (1 + \frac{3}{2} 2^{-2t})^{n+2-r}, \text{ for } r = 2, 3, \ldots, n
\]  
(4.2.54)
Equation (4.2.48) is then rounded to single-precision. So finally, we have,
\[
s_n = [u_1 v_1 (1 + \varepsilon_1) + u_2 v_2 (1 + \varepsilon_2) + \ldots + u_n v_n (1 + \varepsilon_n)](1 + \varepsilon)
\]  
(4.2.55)
where,
\[
1 - 2^{-t} \leq 1 + \varepsilon \leq 1 + 2^{-t}.
\]  
(4.2.56)
4.3 MEASURES OF CONDITION OF A MATRIX

When we solve a set of linear systems,

$$Au = b,$$ (4.3.1)

we hope that the calculated vector $\mathbf{u}_c$ is a close representation of the true solution $\mathbf{u}$. From the previous section we have seen how round-off error can make the computed solution differ from the exact solution. If small relative perturbations in the parameters cause large relative errors in the solutions, then we say the problem is *ill-conditioned*, if the errors are small we say that the problem is *well-conditioned*.

It is meaningless to speak of an ill-conditioned matrix without specifying the problem with respect to which it is ill-conditioned. For example, if we want to solve (4.3.1) the matrix is ill-conditioned if it is nearly singular, that is if small changes in its elements would cause singularity, but such a matrix may occur in a perfectly well-conditioned way in other contexts, such as the determination of eigenvalues.

We use the ideas of matrix and vector norms to find some quantitative measure of the degree of ill-conditioning of a matrix $A$, which in relation to the solution of linear equations depends on the magnitude of the elements of its inverse $A^{-1}$. We shall assume that the computed inverse which we have obtained implicitly when we are solving (4.3.1) is the exact inverse of a perturbation $A+E$ of the original $A$.

We examine the error of an approximate inverse in the following way. We have,

$$\mathbf{(A+E)-A = E}.$$ (4.3.2)

Pre-multiplication and post-multiplication with $(A+E)^{-1}$ and $A^{-1}$ respectively give,

$$A^{-1}-(A+E)^{-1} = (A+E)^{-1}EA^{-1}$$ (4.3.3)

but,

$$(A+E)^{-1} = (I+A^{-1}E)^{-1}A^{-1}$$ (4.3.4)

Taking norms (known results in the algebra of norms are given in section 1.5) we obtain,
PRECONDITIONING AND VARIATIONAL METHODS

FOR SOLVING LINEAR EQUATIONS

BY

ABDUL WAHAB B. JUSOH, B.Sc., M.Sc.

Submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of the University of Technology, Loughborough, October, 1983.

Supervisor: PROFESSOR D.J. EVANS, Ph.D., D.Sc.,
Department of Computer Studies

© B.J. ABDUL WAHAB, 1983.
the latter condition implying only that the perturbation $E$ should have sufficiently small elements compared with those of $A^{-1}$.

Then if,

$$
\frac{||E||}{||A||} = \varepsilon, \quad ||A|| \cdot ||A^{-1}|| = k
$$

we have,

$$
\frac{||A^{-1} - (A+E)^{-1}||}{||A^{-1}||} \leq \frac{k\varepsilon}{1-k\varepsilon}, \quad \text{where } k\varepsilon < 1.
$$

Equation (4.3.7) shows that the relative error for the computed inverse is proportional to $k=||A|| \cdot ||A^{-1}||$, and this is called a conditioned number for the matrix. In the case of spectrum norm ($||.||_2$) see definition 1.6, we define the $k$-condition number as,

$$
k(A) = ||A||_2 \cdot ||A^{-1}||_2 = \frac{\lambda_1}{\lambda_n},
$$

where $\lambda_1, \lambda_n$ are the largest and smallest eigenvalues of $A^H A$ respectively. In particular, if $A$ is symmetric:

$$
k(A) = \frac{\lambda_1}{\lambda_n},
$$

where $\lambda_1, \lambda_n$ are the largest and smallest eigenvalues of $A$ respectively. Equation (4.3.9) is also called the $P$-condition number and was first introduced by Von-Neuman and Goldstine,

$$
P(A) = \frac{\lambda_1}{\lambda_n}.
$$

They established that if $A$ is symmetric and not positive definite then,

$$
||\xi|| \leq 36.58 \cdot P(A) n^2 \varepsilon,
$$

where $\xi$ is the eigenvalue of the largest modulus of $I-AR$, $R$ is the approximate inverse of $A$ obtained by Gaussian elimination, $n$ is the order of the matrix $A$ and $\varepsilon$ is the smallest number recognised by the computer in use. ($2^{-t}$ for a machine word of $t$ bits, Loughborough University $\varepsilon=2^{-38}$).

When $A$ is symmetric and positive definite, then (4.3.11) can be expressed as,

$$
||\xi|| \leq 14.24 \cdot P(A) n^2 \varepsilon,
$$
They further established that when A is approximately singular, its least eigenvalue is bounded by \(10n^2 \varepsilon\) and \(10.5n^2 \varepsilon\) for the non-positive definite and positive-definite A respectively.

Turing first recognised that the condition number should depend symmetrically on both A and \(A^{-1}\) or specifically on the product of their norms. He thus defined the M and N condition number as follows:

\[
M(A) = n \max \{ |a_{ij}| \} \max \{ |a_{ij}| \},
\]

\[
N(A) = n^{-1} \left( \sum_{i,j} a_{i,j}^2 \right)^{1/2} \left( \sum_{i,j} a_{i,j}^{-2} \right)^{1/2},
\]

where the terms \((a_{i,j})\) are the element of \(A^{-1}\) and n is the order of the matrix.

It can be shown that the following relationship holds:

\[
\frac{M(A)}{n} \leq P(A) \leq nM(A)
\]

and,

\[
\frac{M(A)}{n^2} \leq N(A) \leq M(A)
\]

Another measure of ill-conditioning is the following.

We divide the \(i\)\(^{th}\) row of the matrix A by

\[
\left( \sum_{j=1}^{n} a_{i,j}^2 \right)^{1/2}, \quad i=1,2,\ldots,n
\]

to obtain a normalised form of the matrix. The system is said to be ill-conditioned if the determinant of the normalised matrix is very small compared to \(\pm 1\).
4.4 APPLICATION TO DIRECT METHODS OF SOLUTION

Given the system

\[ Au = b, \]  

(4.4.1)

and let \( P \) and \( Q \) be two \( n \times n \) non-singular matrices. Then (4.4.1) can be transformed into the equivalent system,

\[ PAQv = PB, \]  

(4.4.2)

or

\[ Bv = b', \]  

(4.4.3)

\[ u = Qv, \]  

(4.4.4)

where \( B = PAQ \) and \( b' = PB \).

So we can solve (4.4.3) and then from (4.4.4) we calculate the solution \( u \).

The problem is to find \( P \) and \( Q \) such that matrix \( B \) is better conditioned than \( A \). Evans (1968) introduced one of these transformations for large sparse matrices and developed a class of iterative methods which possessed superior convergence rates. (We will go into relative details of his work and others in connection with iterative methods in the next section).

Hatzopoulos (1974) extends this preconditioning technique to the ill-conditioned matrices for which the solution method is the Gauss elimination scheme incorporating a pivoting strategy. Together with Evans, they pursued the effects of the preconditioning strategy on direct methods (1975, 1979 and 1983). Their method of reasoning is as follows. Given (4.4.1), where \( A \) is symmetric, positive definite and has the form,

\[ A = I + L + LT, \]  

(4.4.5)

where \( I \) is the identity matrix and \( L \) is a strictly lower triangular matrix.

The matrices \( P \) and \( Q \) in (4.4.2) are chosen to be \((I+\omega L)\) and \((I+\omega LT)\) respectively, and \( \omega \) is a parameter to be defined later. Then \( B, v \) and \( b' \) in (4.4.2) to (4.4.4) are given by the following:

\[ B_\omega = B = (I+\omega L)^{-1}A(I+\omega LT)^{-1}, \]  

(4.4.6)
\[ b' = \begin{pmatrix} 1 + \omega L & -1 \\ \end{pmatrix} b, \tag{4.4.7} \]
and
\[ v = \begin{pmatrix} 1 + \omega L^T \end{pmatrix}^{-1} u. \tag{4.4.8} \]

Thus, the system (4.4.3) can now be written as
\[ B_{\omega} v = b'. \tag{4.4.9} \]

Since matrix symmetry has been preserved in the transformation (4.4.9), \( B_{\omega} \) is thus a symmetric, positive definite matrix. For \( \omega = 0 \) it can be seen that \( B_{\omega} \) reverts back to the original matrix \( A \) in (4.4.6). Thus, by introducing the transformation above we allow \( \omega \) to play the role of a preconditioning parameter such that as \( \omega \) varies in a restricted range \( 0 < \omega < W \) (say) a minimum value of the condition number of the matrix \( B_{\omega} \) is obtained. Thus, at the optimum \( \omega \) we solve the system (4.4.9) and from (4.4.8) we can calculate the solution \( u \) to the original system.

In fact \( B_{\omega} \) can be written in the form,
\[ B_{\omega} = S_{\omega} + S_{\omega}^T, \tag{4.4.10} \]
where
\[ S_{\omega} = \begin{pmatrix} 1 + \omega L \end{pmatrix}^{-1} (4I + L) \begin{pmatrix} 1 + \omega L^T \end{pmatrix}^{-1}, \tag{4.4.11} \]
and \( S_{\omega} \) can be accurately computed by using simple algorithms of backward and forward substitution on the rows and columns of \( A \).

Evans and Hatzopoulos (1983) have shown that for the given matrix \( B_{\omega} \) the \( P \)-condition number is bounded and satisfies the relationship,
\[ P(B_{\omega}) \leq \sqrt{2(2\beta + 1) \alpha + 1}, \tag{4.4.12} \]
where,
\[ \alpha = \sup_{v \in \mathbb{R}^n} \frac{v^T v}{v^T A v}, \tag{4.4.13} \]
and
\[ \beta = \sup_{v \in \mathbb{R}^n, v \neq 0} \frac{v^T (L L^T + 4I) v}{v^T A v}. \tag{4.4.14} \]

The minimum value of \( P(B_{\omega}) \) in (4.4.12) occurs for an optimum value of \( \omega \) given by,
\[ \omega_{\text{opt}} = \frac{2}{\sqrt{(1 + 2(2\beta + 1)/\alpha)}}. \tag{4.4.15} \]

For the more general case when \( A = D + L + L^T \), we first use a transformation of the form,
which can be regarded as a form of pre-scaling. The preconditioned matrix now has the form,
\[
B_\omega = D^T (D + \omega I)^{-1} A (D + \omega I)^T D^T,
\]
(4.4.17)
and the relations (4.4.12) and (4.4.15) still hold.

If the alternative upper bounds
\[
\alpha = \sup_{V} \frac{V^T D V}{V^T A V}, \quad V \in \mathbb{R}^n,
\]
(4.4.18)

\[
\beta = \sup_{V} \frac{V^T (L D^{-1} L^T - \lambda D) V}{V^T A V}
\]
(4.4.19)
are used and when,
\[
|L D^{-1} L^T| \leq \frac{1}{4} |D|
\]
(4.4.20)
we have
\[
\beta = 0,
\]
and the sharper bounds,
\[
P(B_\omega) \leq \sqrt{\alpha/2} + \beta
\]
(4.4.21)
is obtained.

In order to test the theoretical analysis, we now consider the Hilbert matrix defined by,
\[
A = (a_{i,j}), \quad a_{i,j} = \frac{1}{i+j-1}, \quad i,j=1,2,\ldots,n
\]
(4.4.22)
which enjoy the reputation of being very ill-conditioned matrices with respect to inversion and occur in the solution of boundary value problems by Rayleigh-Ritz variational methods which will be discussed in the next chapter.

We shall consider the system (4.4.1) where the matrix A is the (5x5) segment of the Hilbert matrix.

To obtain the form of (4.4.5) we first use a transformation (4.4.16) to the matrix A. We now proceed to evaluate the preconditioned matrix $B_\omega$ given by (4.4.6). The vector b is chosen such that
\[ b(i) = \sum_{j=1}^{5} a_{ij}, \quad i=1,2,\ldots,5 \]

This ensures that the preconditioned system (4.4.3) has the solution
\[ u^T = (1,1,1,1,1). \]

The results obtained are shown in the accompanying tables.

The tabulation of the maximum and minimum eigenvalues of the pre-conditioned matrix \( B_{\omega} \) and a comparison of the \( P, M \) and \( N \) condition numbers for specific values of the preconditioning parameter \( \omega \) in the range of \( 0 < \omega < 2 \) is given in Table 4.4.1. Then, the preconditioned linear system (4.4.9) was solved using the Gaussian elimination scheme employing single precision arithmetic. Table 4.4.2 gives the relative Euclidean error norm, the maximum error norm and the error sum norm as defined by

\[
\text{relative Euclidean error norm } ||E||_2 = \left\{ \sum_{i=1}^{n} (u_i - u_i)^2 / \|u_i\|^2 \right\}^{1/2},
\]

the maximum error norm \( ||E||_\infty = \max_{i=1}^{n} |u_i - u_i| \),

and error sum norm \( ||E||_1 = \sum_{i=1}^{n} |u_i - u_i| \).

The error norm is reduced by a factor of \( 10^{-2}, 10^{-2} \) and \( 10^{-3} \) respectively from its value at \( \omega = 0 \) (no preconditioning) to its value at \( \omega = 1.4 \) (optimal preconditioning). In addition, the solutions at \( \omega = 0 \) and \( \omega = 1.4 \) are given in Table 4.2.3 and verify further that two more significant figures of accuracy are given when optimal preconditioning of the system is applied.
<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Maximum Eigenvalue $\lambda_1$</th>
<th>Minimum Eigenvalue $\lambda_N$</th>
<th>P-condition Number $p=\lambda_1/\lambda_N$</th>
<th>N-condition Number</th>
<th>M-condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4.4441</td>
<td>0.209559$\times10^{-4}$</td>
<td>2.12031$\times10^{5}$</td>
<td>4.25078$\times10^{4}$</td>
<td>1.28394$\times10^{5}$</td>
</tr>
<tr>
<td>0.2</td>
<td>2.40188</td>
<td>0.258282$\times10^{-4}$</td>
<td>9.29942$\times10^{4}$</td>
<td>1.90201$\times10^{4}$</td>
<td>1.07464$\times10^{5}$</td>
</tr>
<tr>
<td>0.4</td>
<td>1.54823</td>
<td>0.323474$\times10^{-4}$</td>
<td>4.78625$\times10^{4}$</td>
<td>1.00296$\times10^{4}$</td>
<td>8.82964$\times10^{4}$</td>
</tr>
<tr>
<td>0.6</td>
<td>1.19028</td>
<td>0.411432$\times10^{-4}$</td>
<td>2.89301$\times10^{4}$</td>
<td>6.11151$\times10^{3}$</td>
<td>7.10386$\times10^{4}$</td>
</tr>
<tr>
<td>0.8</td>
<td>1.104167</td>
<td>0.534451$\times10^{-4}$</td>
<td>1.94904$\times10^{4}$</td>
<td>4.10676$\times10^{3}$</td>
<td>5.55914$\times10^{4}$</td>
</tr>
<tr>
<td>1.0</td>
<td>1.00000</td>
<td>0.697440$\times10^{-4}$</td>
<td>1.43382$\times10^{4}$</td>
<td>2.98530$\times10^{3}$</td>
<td>4.20793$\times10^{4}$</td>
</tr>
<tr>
<td>1.1</td>
<td>1.01010</td>
<td>0.793325$\times10^{-4}$</td>
<td>1.27325$\times10^{4}$</td>
<td>2.63446$\times10^{3}$</td>
<td>3.61402$\times10^{4}$</td>
</tr>
<tr>
<td>1.2</td>
<td>1.04167</td>
<td>0.899315$\times10^{-4}$</td>
<td>1.15829$\times10^{4}$</td>
<td>2.38223$\times10^{3}$</td>
<td>3.05825$\times10^{4}$</td>
</tr>
<tr>
<td>1.3</td>
<td>1.09889</td>
<td>0.100659$\times10^{-3}$</td>
<td>1.09170$\times10^{4}$</td>
<td>2.23458$\times10^{3}$</td>
<td>2.55713$\times10^{4}$</td>
</tr>
<tr>
<td>1.4</td>
<td>1.19032</td>
<td>0.110623$\times10^{-3}$</td>
<td>1.07601$\times10^{4}$</td>
<td>2.19032$\times10^{3}$</td>
<td>2.46115$\times10^{4}$</td>
</tr>
<tr>
<td>1.5</td>
<td>1.33174</td>
<td>0.118425$\times10^{-3}$</td>
<td>1.2455$\times10^{4}$</td>
<td>2.27866$\times10^{3}$</td>
<td>2.46385$\times10^{4}$</td>
</tr>
<tr>
<td>1.6</td>
<td>1.55141</td>
<td>0.123065$\times10^{-3}$</td>
<td>1.26064$\times10^{4}$</td>
<td>2.54347$\times10^{3}$</td>
<td>2.46245$\times10^{4}$</td>
</tr>
<tr>
<td>1.7</td>
<td>1.89844</td>
<td>0.123276$\times10^{-3}$</td>
<td>1.53999$\times10^{4}$</td>
<td>3.09616$\times10^{3}$</td>
<td>2.46597$\times10^{4}$</td>
</tr>
<tr>
<td>1.8</td>
<td>2.45561</td>
<td>0.120136$\times10^{-3}$</td>
<td>2.04403$\times10^{4}$</td>
<td>4.11903$\times10^{3}$</td>
<td>2.46336$\times10^{4}$</td>
</tr>
<tr>
<td>2.0</td>
<td>4.82195</td>
<td>0.102326$\times10^{-3}$</td>
<td>4.71235$\times10^{4}$</td>
<td>9.42040$\times10^{3}$</td>
<td>2.46847$\times10^{4}$</td>
</tr>
</tbody>
</table>

**TABLE 4.4.1:** Values of the maximum and minimum eigenvalues, and the P, N, M, condition numbers versus the preconditioning parameter $\omega$ for the Hilbert (5x5) matrix.
### TABLE 4.4.2: Values of the Relative Euclidean Norm, Max. Error Norm and Error Sum Norm of the solution vector versus the preconditioning parameter $\omega$ for the Hilbert (5x5) matrix

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Rel Euclidean norm</th>
<th>Max. error norm</th>
<th>Error sum norm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.657210x10^{-2}</td>
<td>0.487471x10^{-2}</td>
<td>0.118988x10^{-1}</td>
</tr>
<tr>
<td>0.2</td>
<td>0.331584x10^{-3}</td>
<td>0.251770x10^{-2}</td>
<td>0.590086x10^{-3}</td>
</tr>
<tr>
<td>0.4</td>
<td>0.790400x10^{-2}</td>
<td>0.143266x10^{-2}</td>
<td>0.344772x10^{-2}</td>
</tr>
<tr>
<td>0.6</td>
<td>0.217807x10^{-2}</td>
<td>0.166440x10^{-2}</td>
<td>0.389373x10^{-2}</td>
</tr>
<tr>
<td>0.8</td>
<td>0.753131x10^{-4}</td>
<td>0.584126x10^{-4}</td>
<td>0.131845x10^{-3}</td>
</tr>
<tr>
<td>1.0</td>
<td>0.149897x10^{-4}</td>
<td>0.120401x10^{-4}</td>
<td>0.261068x10^{-4}</td>
</tr>
<tr>
<td>1.1</td>
<td>0.134309x10^{-3}</td>
<td>0.102758x10^{-3}</td>
<td>0.216246x10^{-3}</td>
</tr>
<tr>
<td>1.2</td>
<td>0.489909x10^{-4}</td>
<td>0.350475x10^{-4}</td>
<td>0.755787x10^{-4}</td>
</tr>
<tr>
<td>1.3</td>
<td>0.124051x10^{-4}</td>
<td>0.104904x10^{-4}</td>
<td>0.213385x10^{-4}</td>
</tr>
<tr>
<td>1.4</td>
<td>0.904871x10^{-4}</td>
<td>0.700951x10^{-4}</td>
<td>0.144482x10^{-3}</td>
</tr>
<tr>
<td>1.5</td>
<td>0.103307x10^{-4}</td>
<td>0.882149x10^{-5}</td>
<td>0.171661x10^{-4}</td>
</tr>
<tr>
<td>1.6</td>
<td>0.286966x10^{-3}</td>
<td>0.222683x10^{-3}</td>
<td>0.511646x10^{-3}</td>
</tr>
<tr>
<td>1.7</td>
<td>0.614694x10^{-4}</td>
<td>0.436306x10^{-4}</td>
<td>0.116467x10^{-3}</td>
</tr>
<tr>
<td>1.8</td>
<td>0.584311x10^{-4}</td>
<td>0.463724x10^{-4}</td>
<td>0.100619x10^{-3}</td>
</tr>
<tr>
<td>2.0</td>
<td>0.304953x10^{-3}</td>
<td>0.212908x10^{-3}</td>
<td>0.571609x10^{-3}</td>
</tr>
<tr>
<td>Exact Solution</td>
<td>Solution at $\omega=0$</td>
<td>Solution at $\omega=1.4$</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.9999055862</td>
<td>0.999994040</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.001068830</td>
<td>1.000008821</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.9962905645</td>
<td>0.999908209</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.004874706</td>
<td>0.999942101</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.9978486300</td>
<td>1.000070095</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rel. Euclidean norm</th>
<th>0</th>
<th>$0.657210 \times 10^{-2}$</th>
<th>$0.904871 \times 10^{-4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. Error norm</td>
<td>0</td>
<td>$0.487471 \times 10^{-2}$</td>
<td>$0.700951 \times 10^{-4}$</td>
</tr>
<tr>
<td>Error Sum norm</td>
<td>0</td>
<td>$0.118988 \times 10^{-1}$</td>
<td>$0.144482 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

**TABLE 4.4.3**: Comparison of the exact solution values obtained for no preconditioning ($\omega=0$) and optimal preconditioning ($\omega=1.4$)
Evans and Hatzopoulos (1983) have also examined the relative merits of the use of preconditioning as opposed to an optimal scaling method for solving ill-conditioned linear systems. From the example on the test matrices used, they have shown that although the two methods decrease the condition numbers by approximately the same factor, there is a factor of $10^{-2}$ between the error norms. This gives an important conclusion of preconditioning direct methods of solution of ill-conditioned systems of linear equations, i.e. the rounding errors incurred in the computational process are minimised, thus possibly avoiding the need for higher precision arithmetic.
4.5 APPLICATION TO ITERATIVE METHODS OF SOLUTION

It was shown earlier in Section 3.3 that the optimum rate of convergence for both the method of Simultaneous Displacements (4.2.7) and Richardson's second degree method (4.2.42) even with the 'optimum' choice of the parameters given by (4.2.35) and (4.2.43) respectively is inversely proportional to the value of the P-condition number (def. 3.3.7) of the coefficient matrix A (3.3.8). Consequently, we require the coefficient matrix of the system (4.4.1) to have a P-condition number as small as possible to achieve a fast convergence rate. To do this, some form of pre-conditioning of the original system must be applied so as to transform it into a new system whose coefficient matrix has a P-condition number which can be minimised and hence increase the rate of convergence.

This concept of preconditioning was first introduced by Evans (1968) who applied it to the iterative methods for solving linear equations with symmetric, positive definite matrices. His method of ideas is as follows. We note that if we premultiply (4.4.1) by \( A^{-1} \), then we obtain immediately its solution \( u = A^{-1}d \). However, there are certain difficulties for computing \( A^{-1} \), especially when (4.4.1) is derived from the approximation of elliptic PDE's, so instead we consider the case where the above system is pre-multiplied by a non-singular matrix \( R^{-1} \), where \( R^{-1} \) is an approximate inverse of \( A \). Thus, the more \( R \) resembles \( A \), the faster the method will converge.

The success of the method will depend on two properties to be satisfied by \( R \):

(i) The spectral condition number of the matrix \( R^{-1}A \) becomes smaller than the spectral condition number of \( A \).

(ii) For any vectors \( s \) and \( t \) it is "computationally convenient" to solve the system \( Rs = t \), i.e. \( R \) is easily solvable. (Evans & Missirlis, 1983).

\[ k(M) = \frac{\|M\|}{\|M^{-1}\|}. \]
Thus, we transform the original system (4.4.1) into the following preconditioned form,
\[ R^{-1}Au = R^{-1}b. \] (4.5.1)

The matrix \( R \) is referred to as the preconditioning matrix and we require \( R \) to satisfy the two conditions above.

Then, we can define the general iterative scheme,
\[ u^{(n+1)} = u^{(n)} + \tau R^{-1}(b - Au^{(n)}), \] (4.5.2)

where \( \tau \) is a real parameter which is consistent with (4.5.1) iff \( R \) is non-singular and \( \tau \neq 0 \).

Evans et al (1981) consider two possible forms which the matrix \( R \) can possess within the context of the above discussion. These are when \( R \) is closely associated with a) the splitting and b) the approximate factorisation of the matrix \( A \).

a) Splitting of \( A \)

If in the given system of equation (4.4.1), the matrix \( A \) satisfies the splitting,
\[ A = D - C_L - C_U, \] (4.5.3)

where \( D \) is a diagonal matrix with the same diagonal elements as \( A \) and \( C_L \) and \( C_U \) are strictly lower and upper triangular matrices respectively, then a suitable form of \( R \) is,
\[ R = (D - \omega C_L)D^{-1}(D - \omega C_U). \] (4.5.4)

When \( R = I \) and \( \tau = 1 \), we can formulate many of the fundamental iterative methods based on the splitting principle which have been used extensively to date and show that their convergence performance inversely depends on the spectral condition number.

For example, if we split
\[ A = I - B, \] (4.5.5)
then, (4.5.2) can be written in the form,
\[ u^{(n+1)} = (I - A)u^{(n)} + b = u^{(n)} + r^{(n)}, \] (4.5.6)

showing that the change in each component is equal to the corresponding component of the residual vector \( r^{(n)} \).
If a constant factor $\alpha$ or a different choice $\alpha_n$ for each iteration is multiplied by each component of the residual vector in (4.5.6), and then added to each component of the present iterate $u^{(n)}$ gives rise to the equations,

\[
\begin{align*}
    \mathbf{u}^{(n+1)} &= \mathbf{u}^{(n)} + \alpha\mathbf{r}^{(n)} = (I-\alpha\mathbf{A})\mathbf{u}^{(n)} + \alpha\mathbf{b}, \\
    \mathbf{u}^{(n+1)} &= \mathbf{u}^{(n)} + \alpha\mathbf{r}^{(n)} = (I-\alpha\mathbf{A})\mathbf{u}^{(n)} + \alpha\mathbf{b},
\end{align*}
\]

(4.5.7) and

(4.5.8)

which are the well known Simultaneous Displacement and Richardson's method respectively (see Section 3.3).

For a different conditioning matrix $\mathbf{R}$ and different values of parameters $\tau$ and $\omega$, we construct Table 4.5.1 (as given by Evans et al (1981) and Evans & Missirlis (1983)) where,

\[
    \mathbf{L} = \mathbf{D}^{-1}\mathbf{C}_L \quad \text{and} \quad \mathbf{U} = \mathbf{D}^{-1}\mathbf{C}_U
\]

(4.5.9)

and $\bar{\alpha}$ and $\bar{\beta}$ are the minimum and maximum eigenvalues of the corresponding preconditioned matrix $\mathbf{R}^{-1}\mathbf{A}$, respectively, i.e.,

\[
    \bar{\alpha} \leq \lambda(\mathbf{R}^{-1}\mathbf{A}) \leq \bar{\beta}.
\]

(4.5.10)

From above, it is concluded that all known convergent iterative methods can be interpreted as improving the "condition" of the given system (4.4.1) by using different types of the conditioning matrices $\mathbf{R}$ in (4.5.1) (Evans & Missirlis, 1983). In particular, from the analogy of the Jacobi and S.D. iterative scheme, we see that for $\omega=0$ the optimum value $\tau_0=2/(\bar{\alpha}+\bar{\beta})$ an improvement in the convergence rate is assured. Thus determining good estimates $\tau_1, \omega_1$ of $\tau$ and $\omega$ respectively for $\mathbf{R}$ given by (4.5.4) is of paramount importance, so that we have an improvement in the convergent rate for the P.S.D. method over the S.S.O.R. method. These strategies have been referred to by Evans (1981) as Sparse Preconditioning since matrix sparseness is retained throughout. His relevant work on elliptic PDE's is given in the above-mentioned paper and is summarised briefly as follows.

We premultiply and post-multiply (4.5.3) by $\mathbf{D}^{-1}$,
<table>
<thead>
<tr>
<th>( \omega )</th>
<th>( \tau )</th>
<th>( R )</th>
<th>Iterative Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>D</td>
<td>Jacobi (J)</td>
</tr>
<tr>
<td>0</td>
<td>( 2/(\bar{a}+\bar{b}) )</td>
<td>D</td>
<td>Jacobi Overrelaxation (JOR)</td>
</tr>
<tr>
<td>0</td>
<td>( 2/(\bar{a}+\bar{b}) )</td>
<td>I</td>
<td>Simultaneous Displacement (SD)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>D(I-L)</td>
<td>Gauss-Seidal (GS)</td>
</tr>
<tr>
<td>0</td>
<td>( \omega )</td>
<td>D(I-( \omega )L)</td>
<td>Successive Overrelaxation (SOR)</td>
</tr>
<tr>
<td>( \omega )</td>
<td>( \omega(2-\omega) )</td>
<td>D(I-( \omega )L)(I-( \omega )U)</td>
<td>Symmetric SOR (SSOR)</td>
</tr>
<tr>
<td>( \omega )</td>
<td>1</td>
<td>&quot;</td>
<td>Preconditioned Jacobi (PJ)</td>
</tr>
<tr>
<td>( \omega )</td>
<td>( \omega )</td>
<td>&quot;</td>
<td>Extrapolated Modified Aitken (EMA)</td>
</tr>
<tr>
<td>( \omega )</td>
<td>( 2/(\bar{a}+\bar{b}) )</td>
<td>&quot;</td>
<td>Preconditioned SD (PSD)</td>
</tr>
<tr>
<td>( \bar{w}_1 )</td>
<td>( \bar{\tau}_1 )</td>
<td>&quot;</td>
<td>Sparse P.S.D.</td>
</tr>
</tbody>
</table>

**TABLE 4.5.1**
and \( A' \) can be written in the form,

\[
A' = D^{-1/2} A D^{-1/2},
\]

This transformation (4.5.11) corresponds to a form of scaling and improves the condition number of the system accordingly. Thus, we can always assume that \( A \) can be written (without loss of generality) as \( A = I - L - U \), where \( L, U \) are strictly lower and upper triangular matrices respectively. In this case, \( R \) in (4.5.4) can be written as,

\[
R = (I - \omega L)(I - \omega U).
\]

We rewrite (4.5.2) in the form,

\[
\begin{align*}
\underline{u}^{(n+1)} &= B_{\tau, \omega} \underline{u}^{(n)} + b_{\tau, \omega},
\end{align*}
\]

where

\[
B_{\tau, \omega} = (I - \tau(I - \omega U))^{-1} (I - \omega L)\text{ and } b_{\tau, \omega} = (I - \omega L)^{-1}(I - \omega U)^{-1} b.
\]

For \( \omega = 0 \), we see that (4.5.15) reverts to the Simultaneous Displacement Method

\[
\underline{u}^{(n+1)} = (I - \tau A)\underline{u}^{(n)} + \tau b.
\]

The value of \( \tau \) for which \( S(B_{\tau, \omega}) \) the spectral radius of \( B_{\tau, \omega} \) attains its minimum value is when

\[
\tau = \tau_1 = 2/(\Lambda(A_w) + \lambda(A_w)),
\]

where \( \Lambda(A_w) \) and \( \lambda(A_w) \) denote the maximum and minimum eigenvalues of \( A_w \) respectively. (see 3.3.35).

It was shown that a good choice of \( w_1 \) was given by minimising \( P(A_w) \), the \( P \)-condition number of \( A_w \). This is achieved by letting \( \omega = w_1 \), i.e.,

\[
\omega_1 = \begin{cases} 
2/[(1 + (1 - 2M + 4S))^{1/2}], & \text{if } M \leq 4S, \\
2/[(1 - 4S)^{1/2}], & \text{if } M > 4S,
\end{cases}
\]

where \( M \) satisfies the relation,

\[
M(B) \leq M_{\min}(1, 2\sqrt{S}) \text{ and } S(LU) \leq \sqrt{S},
\]

and \( M(B) \) denotes the maximum eigenvalue of \( B \), i.e. \( (L + U) \).

Finally, further improvement in the rate of convergence is obtained
from the P.S.D. semi-iterative method, at the expense of an extra vector storage, as,
\[
\mathbf{u}_{n+1} = \mathbf{u}_n + \rho_{n+1} (\mathbf{u}_n - \mathbf{u}_{n-1}) + \rho_{n+1} \mathbf{r}_n \mathbf{L}_{-1}(\mathbf{I} - \omega \mathbf{L}_{-1})^{-1}(\mathbf{I} - \omega \mathbf{L}_{-1})^{-1}(\mathbf{D} - \mathbf{A})^{-1} \mathbf{u}_n,
\]
(4.5.19)
where,
\[
\rho_1 = 1, \rho_2 = (1 - \sigma^2/2)^{-1}, \ldots \rho_{n+1} = (1 - \sigma^2 \rho_n/4)^{-1}, n = 2, 3, \ldots
\]
and
\[
\sigma = (P(A)_w - 1)/(P(A)_w + 1).
\]
The asymptotic average rate of convergence for the PSD-SI method is,
\[
2/\sqrt{P(A)_w}.
\]
(4.5.20)
These results are confirmed by Table 4.5.2 which we reproduce from the results obtained by Evans (1981) for the model Dirichlet problem, i.e. the Laplace equation in the unit square with zero boundary values. The Table also shows a substantial improvement over the S.O.R. method with optimal parameters.

<table>
<thead>
<tr>
<th>h^{-1}</th>
<th>\omega_1</th>
<th>\tau_1</th>
<th>PSD-SI no. of iters.</th>
<th>\omega_b</th>
<th>SOR no. of iters.</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1.7640</td>
<td>0.6993</td>
<td>17</td>
<td>1.7295</td>
<td>61</td>
</tr>
<tr>
<td>40</td>
<td>1.8750</td>
<td>0.4264</td>
<td>24</td>
<td>1.8547</td>
<td>121</td>
</tr>
<tr>
<td>60</td>
<td>1.9157</td>
<td>0.3031</td>
<td>30</td>
<td>1.9005</td>
<td>181</td>
</tr>
</tbody>
</table>

TABLE 4.5.2

Evans and Missirlis (1983) have also considered different splitting of the matrix \( R \) in the Alternating Direction Implicit methods in which \( R \) has the form,
\[
R = (I + rH)(I + rV) = I + rA + r^2 H V,
\]
(4.5.21)
where \( r \) is a real parameter and \( A = H + V \).

Although the methods J and JOR (Table 5.1.1) have identical conditioning matrices, the optimum value \( \tau_0 = 2/\sqrt{(a+b)} \) for the parameter assures an improvement in the rate of convergence of JOR over the J method. Thus, given \( R \), the most effective iterative scheme is obtained if \( \tau \) attains its optimum value \( \tau_0 \). However, this generalisation does not seem to hold true for every case, for example the SSOR and EMA
methods (see Table 4.5.1). It is therefore essential to have an alternative preconditioning strategy to the methods just described.

b) **Approximate Factorisation of A**

In Section 3.2, we described the direct methods of solution of the system (4.4.1). Among them is the LU decomposition. Methods for such factorisation are well known and are given in Table 4.5.3 below as well as in many standard reference books.

<table>
<thead>
<tr>
<th>EXACT METHOD</th>
<th>R CONDITIONING MATRIX</th>
<th>ITERATIVE METHOD 4.5.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Gaussian Elimination ( A=L^{-1}U )</td>
<td>( L^{-1}U )</td>
<td>Evans (1974)</td>
</tr>
<tr>
<td>2. Triangular Factorisation ( A=LU )</td>
<td>( LU )</td>
<td>Stone (1968), Evans &amp; Lipitakis, (1979)</td>
</tr>
<tr>
<td>3. Choleski Square Root ( A=Q^{T}Q )</td>
<td>( QT )</td>
<td>Duport (1968), Meijerink &amp; Van de Vorst (1977), Gustafsson (1978)</td>
</tr>
<tr>
<td>4. Root Free Factorisation ( A=LDU )</td>
<td>( LDU )</td>
<td>Evans &amp; Lipitakis (1982)</td>
</tr>
<tr>
<td>5. Root Free Choleski ( A=LDLT )</td>
<td>( LDLT )</td>
<td>Kershaw (1978)</td>
</tr>
</tbody>
</table>

**TABLE 4.5.3**

If we apply this method to the matrix \( R \), we have

\[
R = \frac{1}{2} T^n T.
\]

(4.5.22)

Then the solution of (4.4.1) can be achieved in two steps (see equation 4.5.2).
The choice of \( R \) most ideal for the iteration process (4.5.2) is \( A \), since only one iteration is needed. However, in the triangular factorisation process of a sparse matrix \( A \), a large number of the zero entries are replaced or "filled in" by non-zero values. Severe disadvantages arise from these entries for not only are they a source of rounding errors in any computation in which they are involved but they greatly increase the amount of work and computer storage involved.

This suggests that we look for matrices \( R = \widetilde{LU} \) which resemble \( A \) whilst still retaining the property \( R = LU \) where \( L, U \) denote the corresponding sparse triangular matrix factors in which \( r \) non-zero 'fill in' vectors have been retained. Thus, in recent years, research has been directed towards a number of indirect or iterative methods in which the strategy is to reduce the fill in terms of the triangular factors i.e. to make \( \widetilde{L} \) and \( \widetilde{U} \) almost as sparse as \( A \). Since the fill-in terms \((r>1)\) are confined to a compact form, these strategies are referred to as Compact Preconditioning. Of course, the strategies by which each method retains its sparseness varies differently and we will only describe one such strategy for constructing \( R \) which satisfies the above-mentioned requirements.

c) Preconditioning by Incomplete Factorisation

In this section we shall consider the factorisation of \( A \) when \( A \) is symmetric, sparse and positive definite. Then it is known [Varga, 1962] that for such a matrix there exists a unique factorisation of the form,

\[
A = LDL^T,
\]

where \( D \) is a positive diagonal matrix and \( L \) a unit lower triangular matrix and \( L^T \) denotes the transpose of \( L \).
To avoid the disadvantage of the direct factorisation methods i.e. L is now full, we do not construct L and D so that (4.5.24) holds. Instead we define a matrix R such that,

\[ R = \tilde{L} \tilde{D}^T, \]

which resembles A by insisting that \( \tilde{L} \) has non-zero entries only in those positions which correspond to the non-zero elements in the lower triangle of A. The product \( \tilde{L} \tilde{D}^T \) is then formed and the elements of \( \tilde{L} \) and D computed by equating only the non-zero elements of A to those of \( \tilde{L} \tilde{D}^T \). For this reason, the method is referred to as 'incomplete factorisation'.

To illustrate the process we consider the Laplace two-dimensional model problem on a unit square with a grid of size \( h = 1/N \) (see Section 2.2). By using the usual row by row ordering of grid points we obtain the five-point difference formula (2.2.6) and a square matrix A of order \( (N-1)^2 \).

The structure of A is shown in Figure 4.5.1. (Mitchell and Griffiths, 1980).

![Figure 4.5.1](image)

The kth component of \( Au \) may be written in the form,

\[ (Au)_k = a_{k,k}u_k + a_{k,k+1}u_{k+1} + a_{k,k+N}u_{k+N} + a_{k-1,k}u_{k-1} + a_{k-N,k}u_{k-N} \]

for \( k = 1, 2, \ldots, n \) \( (= (N-1)^2) \) (4.5.26)

where \( a_{j,k} = 0 \) for \( j < 0 \) and \( a_{k,j} = 0 \) for \( j > n \).

The form of (4.5.26) is depicted by the computational molecule in Figure 4.5.2.
Suppose that D has diagonal elements $d_1, d_2, \ldots, d_n$ and $\tilde{L}$ is the unit lower triangular as shown in Figure 4.5.3. Then, the $k$th component of $\tilde{L}u$ will have the form,

$$ (\tilde{L}u)_k = u_k + \ell_{k-1}u_{k-1} + \ell_{k-N}u_{k-N} $$

for $k = 1, 2, \ldots, n$. (4.5.27)

Similarly,

$$ (\tilde{L}^T u)_k = u_k + \ell_{k,k+1}u_{k+1} + \ell_{k,k+N}u_{k+N} $$

and therefore,

$$ (DL^T u)_k = \frac{1}{n} (u_k + \ell_{k,k+1}u_{k+1} + \ell_{k,k+N}u_{k+N}) $$

(4.5.29)

From (4.5.27),
\[ (\tilde{L}^{T}LD_{k})_{k} = (\tilde{L}^{T}u)_{k} + \ell_{k-l,k}^{2} (\tilde{L}^{T}u)_{k-1} + \ell_{k-N,k}^{2} (\tilde{L}^{T}u)_{k-N} \]  

which becomes,

\[ (LD_{k})_{k} = (a_{k} + d_{k-l}^{2} \ell_{k-l,k}^{2} + d_{k-N}^{2} \ell_{k-N,k}^{2})_{k} \]

\[ + d_{k} (\ell_{k}^{2} + \ell_{k}^{2} u_{k} + \ell_{k}^{2} u_{k+1}) \]

\[ + d_{k-l} \ell_{k-l,k}^{2} (u_{k-l} + \ell_{k-l} u_{k+N-1}) \]

\[ + d_{k-N} \ell_{k-N,k}^{2} (u_{k-N} + \ell_{k-N} u_{k-N+1}). \]  

Equation (4.5.31) shows that the matrix \( LDL^{T} \) has up to seven non-zero entries in each row (see Figure 4.5.4).

By equating the coefficients of \( u_{k}, u_{k+1}, u_{k+N} \) in (4.5.27) and (4.5.31) we obtain,

\[ d_{k} = a_{k} - d_{k-l}^{2} \ell_{k-l,k}^{2} - d_{k-N}^{2} \ell_{k-N,k}^{2}, \quad \text{for } k=1,2,\ldots,n, \]  

\[ \ell_{k}^{2} = a_{k} + \ell_{k}^{2} / d_{k}, \quad \text{for } k=1,2,\ldots,n-1, \]  

\[ \ell_{k}^{2} = a_{k} + \ell_{k}^{2} / d_{k}, \quad \text{for } k=1,2,\ldots,n-N, \]  

with \( d_{j} = \ell_{j,k} = 0 \) for \( j \leq 0 \) and \( a_{k} = \ell_{k} = 0 \), for \( j > n \).

Mitchell and Griffiths (1980) used the incomplete factorisation above in conjunction with the conjugate gradient method (as described in Chapter 3). Indeed, various authors used the variant of incomplete factorisation in conjunction with the conjugate gradient method where for a system of algebraic equations of order \( n \), the conjugate gradient method using exact arithmetic would involve at most \( n \) steps. What is of considerable importance
is that at each step the solution is progressively refined. Although, with finite precision arithmetic the finite termination no longer holds and the solution attainable to the machine precision is obtained only after a very large number of steps, however, it is generally found that each step does progressively improve the solution and in real applications, generally a solution is required to only a finite precision, typically three or four decimal places. Thus, if a reasonable rate of convergence can be achieved, only a small number of iterations of a preconditioned iterative method need be used.

When using the conjugate gradient method on a symmetric, positive definite system it can be shown [Jennings, 1977] that each of the iteration steps successively refines the solution, namely the residuals are strictly monotonically decreasing. The rate at which the solution is refined depends on the matrix coefficient $R^{-1}A$ in (4.5.2). To be precise it depends on the number of distinct eigenvalues and the spread of their values. In the extreme case of a unit matrix convergence is obtained in only one step. The fewer distinct eigenvalues, the fewer steps are needed to obtain the exact solution (Jacobs, 1983). It is not only the number of distinct eigenvalues that is relevant, but also the clustering of the eigenvalues into groups [Mitchell & Wait, 1980]. These eigenvalues have been computed and found to be so by Meijerink and Van der Vorst (1977) and Kershaw (1978) for a wide variety of problems.

Meijerink and Van der Vorst used the Incomplete Cholesky (IC) in conjunction with conjugate gradient method (ICCG). In their paper it is shown that if $A$ is an M-matrix\textsuperscript{†}, then it can always be split into

$$A = R - C,$$  \hspace{1cm} (4.5.35)

where

$$R = L \Sigma U.$$  

\textsuperscript{†} A matrix $A=(a_{ij})$ is an M-matrix if $a_{ij} \geq 0$ for $i \neq j$, $A \neq 0$, and $A^{-1} \geq 0$. 
and L and U are sparse unit lower and upper triangular matrices respectively and E a positive diagonal matrix, and C is a correction matrix. If A is also symmetric, then R will be symmetric too. For this special case, of symmetric M-matrices, they proved that these approximate factorisations produced non-singular factor matrices L and that the factorisation process was at least as stable as a complete Cholesky factorisation.

Their method is restricted to systems characterised by M-matrices. Kershaw (1978) very successfully used the same basic method with a minor modification for solving a wider class of systems for which the matrix is symmetric positive definite, but not necessarily an M-matrix. With an M-matrix Meijerink and Van der Vorst had proved that the main diagonal terms of the approximate factors are always positive for the inexact factorisation they used, as they are for the exact factorisation.

A similar result does not hold for non-M-matrices. It is therefore necessary to ensure that the matrix factor is non-singular in order that the preconditioning is non-singular. This is satisfied if the main diagonal elements of the approximate factors are all non-zero. To achieve this requirement, Kershaw (1978) recommended forcing all the values of the main diagonal elements to be positive. If any main diagonal element was calculated to be non-positive during the approximate factorisation, he set it to a 'suitable' positive value. Such a modification, would only have a very limited effect on the eigenvalue spectrum of the preconditioning matrix, provided the number of such 'corrections' was small. He further shows that for the five-point difference scheme the storage and the amount of work required are very much less for the IC method compared to the complete Cholesky method.

Evans (1981) considers the splitting of a symmetric positive definite matrix A into the form,

\[ A = D^TV^T \]
where $D$ is a positive diagonal matrix and $V$ is a real upper-triangular matrix with unit diagonal. For the model problem (Section 2.2) he forms the approximate normalised factorisation.

$$A = D V^T V D,$$

where $V_r$ is a sparse triangular matrix in which only $r$ outermost off-diagonal (fill-in terms) of the matrix $V$ are retained and $D_r$ is a diagonal matrix as before. He later formulates a normalised implicit gradient method (NICG) which gives a substantial reduction in the number of iterations when compared to CG method as the number of $r$ increases, especially for large $n$ (the size of the matrix $A$). For example, for $n=3481$ and $r=4$, 100 iterations are required for CG to converge whereas only 20 iterations are required for NICG. A similar advantage has been experienced by Evans and Lipitakis (1983) when they introduced the SICG method.

Various difficulties, however may arise in the construction of an IC method and it is often not easy to tackle a large number of practical problems without deteriorating the efficiency of the preconditioned process. Henk van der Vorst (1983) attempts to overcome the stability problems that may occur in the construction of the IC above by forming what he calls a simple decomposition. This decomposition is denoted by,

$$L_1 D_1^{-1} U_1.$$

The factors $L_1$, $D_1$ and $U_1$ are defined by the following relations:

1) $D_1^{-1} = \text{diag}(L_1) = \text{diag}(U_1)$

2) If the element of $A$, $L_1$ and $U_1$ are denoted by $a_{ij}$, $\ell_{ij}$ and $u_{ij}$ respectively, then,

$$\ell_{ij} = 0 \text{, for } i<j, \quad \ell_{ij} = a_{ij} \text{, for } i>j,$$

$$u_{ij} = a_{ij} \text{, for } i<j, \quad u_{ij} = 0 \text{, for } i>j.$$

3) The elements of the diagonal matrix $D_1$ are defined by the relation

$$\text{diag}(L_1 D_1^{-1} U_1) = \text{diag}(A).$$

For the model problem (elliptic 2-dimension), he found that it seems
to be of little gain to construct a more complicated decomposition, that is decompositions which allow for more non-zero fill-in.

A variant of ICCG method has also been studied by Manteuffel (1980). He described the method called the shifted incomplete Cholesky factorisation (SIC) for general symmetric positive definite matrices and discussed its efficiency as well. The method produced a splitting of a matrix $A$ that is dependent upon a parameter $\alpha$ and it is shown to be more efficient (for a set of test problems) than either of the direct methods for explicit iteration schemes.

To show that the preconditioning strategy improves the rate of convergence of iteration methods we will apply the Jacobi (Gauss) iteration method to the preconditioned system given by,

$$B_{\omega}v = b'_{\omega}$$

given by,

$$v^{(n+1)} = c_{\omega}v^{(n)} + b'_{\omega},$$

where $c_{\omega} = I - B_{\omega}$.

The iteration is continued to convergence in the variable $v$ before the transformation,

$$u = (I - \omega I_{\omega})v,$$

(4.5.38)

to the required final solution $u$ is carried out.

**Lemma 4.5.1**

A necessary condition for convergence of the Jacobi iterative scheme is that there exists some norm $M$ of matrix $B$ such that,

$$||M|| = ||L_{\omega} + U_{\omega}|| < 1.$$

(4.5.39)

**Proof:**

We shall assume a splitting of $B_{\omega}$ of the form,

$$B_{\omega} = I - L_{\omega} - U_{\omega},$$

(4.5.40)

where $I$ is an identity matrix and $L_{\omega}$ and $U_{\omega}$ are lower and upper triangular matrices respectively. We can now express (4.5.36) in the form,

$$v^{(n+1)} = c_{\omega}v^{(n)} + b'_{\omega},$$

(4.5.41)
where \( C_\omega = L_\omega + U_\omega \) .

(4.5.42)

It is known that a necessary condition for the effective convergence
of iterative processes of form (4.5.39) is that,

\[
||M|| = ||L_\omega + U_\omega|| = \sum_{i=1}^{n} |b_{ij}| < 1
\]

(4.5.43)

Now, the relation (4.5.42) is always satisfied if \( B_\omega \) has strong diagonal
dominance. Since the transformation (4.5.36) leaves the positive
definiteness of \( B_\omega \) invariant (Evans, 1974) then (4.5.42) is satisfied if
the original matrix \( A \) has strong diagonal dominance.

From the tridiagonal matrix defined as,

\[
A = (a_{ij}) = \begin{cases} 
4, & \text{if } i=j \\
-1, & \text{if } i=j+1 \text{ or } i=j-1 \\
0, & \text{elsewhere}
\end{cases}
\]

we consider the 20x20 linear system:

\[
\begin{bmatrix}
4 & -1 & -1 \\
-1 & 4 & -1 \\
-1 & 4 & -1 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\end{bmatrix}
= 
\begin{bmatrix}
3 \\
-2 \\
\end{bmatrix}
\]

(4.5.44)

The performance of the preconditioned iterative method (4.5.36) on the
coefficient matrix above is shown in the accompanying tables and diagrams/graphs. Table 4.5.1 gives the largest and smallest eigenvalues, P-condition
numbers and the number of iterations required to achieve the accuracy of
\( 5 \times 10^{-6} \) for the matrix \( A \). Table 4.5.4 gives the largest and smallest
<table>
<thead>
<tr>
<th>Preconditioning parameter $\omega$</th>
<th>Smallest Eigenvalue $\lambda_n$</th>
<th>Largest Eigenvalue $\lambda_1$</th>
<th>P-Condition Number</th>
<th>Number of Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.5055846E+00</td>
<td>0.1494416E+01</td>
<td>0.2955817E+01</td>
<td>22</td>
</tr>
<tr>
<td>0.2</td>
<td>0.5595154E+00</td>
<td>0.1356860E+01</td>
<td>0.2425063E+01</td>
<td>19</td>
</tr>
<tr>
<td>0.4</td>
<td>0.6224777E+00</td>
<td>0.1237362E+00</td>
<td>0.1987802E+01</td>
<td>17</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6965823E+00</td>
<td>0.1132921E+01</td>
<td>0.1626399E+01</td>
<td>14</td>
</tr>
<tr>
<td>0.8</td>
<td>0.7845782E+00</td>
<td>0.1041494E+01</td>
<td>0.1327457E+01</td>
<td>11</td>
</tr>
<tr>
<td>0.9</td>
<td>0.8348694E+00</td>
<td>0.1010101E+01</td>
<td>0.1209891E+01</td>
<td>10</td>
</tr>
<tr>
<td>1.0</td>
<td>0.8900534E+00</td>
<td>0.1000000E+01</td>
<td>0.1123528E+01</td>
<td>8</td>
</tr>
<tr>
<td>1.1</td>
<td>0.9227787E+00</td>
<td>0.1010101E+01</td>
<td>0.1094630E+01</td>
<td>7</td>
</tr>
<tr>
<td>1.2</td>
<td>0.8878081E+00</td>
<td>0.1041667E+01</td>
<td>0.1173302E+01</td>
<td>7</td>
</tr>
<tr>
<td>1.3</td>
<td>0.8547776E+00</td>
<td>0.1098641E+01</td>
<td>0.1285295E+01</td>
<td>8</td>
</tr>
<tr>
<td>1.4</td>
<td>0.8235481E+00</td>
<td>0.1178318E+01</td>
<td>0.1430782E+01</td>
<td>10</td>
</tr>
<tr>
<td>1.6</td>
<td>0.7659929E+00</td>
<td>0.1374130E+01</td>
<td>0.1793921E+01</td>
<td>17</td>
</tr>
<tr>
<td>1.8</td>
<td>0.7142456E+00</td>
<td>0.1623455E+01</td>
<td>0.2272964E+01</td>
<td>34</td>
</tr>
<tr>
<td>1.9</td>
<td>0.6903101E+00</td>
<td>0.1773952E+01</td>
<td>0.2569790E+01</td>
<td>61</td>
</tr>
</tbody>
</table>

TABLE 4.5.4
FIG. 4.5.5: EIGENVALUES Vs ACC. PARAMETER W

FIG. 4.5.6: P-CON Vs ACC. PARAMETER W
eigenvalues, P-condition number and the number of iterations required for various values of \( \omega \). A graph of the largest and smallest eigenvalues is plotted against the values of \( \omega \) and is given in Figure 4.5.5. Finally, Figure 4.5.6 shows the P-conditioned numbers plotted for various values of \( \omega \).

Evans (1973) has related the P-condition number of \( B_\omega \) at the optimal value of \( \omega = \omega_0 \) to P-condition number of \( B_\omega \) at \( \omega = 0 \), \( P_0 \), by

\[
P_\omega = O(\sqrt{P_0})
\]

The rate of convergence of the Jacobi iteration is given by,

\[
R = \frac{2}{P}
\]

for \( P >> 1 \).

From (4.5.46) we have,

\[
R_0 = \frac{2}{P_0}
\]

and

\[
R_\omega = \frac{2}{P_\omega}
\]

\[
\begin{align*}
   & = \frac{2}{O(\sqrt{P_0})} \\
   & = \frac{\sqrt{2}}{\sqrt{2}} \frac{\sqrt{2}}{O(P_0)} \\
   & = O(\sqrt{P_0}) \\
   & = O(\sqrt{R_0}) \\
\end{align*}
\]

From the definition of the P-condition number and the rate of convergence it can be shown that the number of iterations at the optimal \( \omega \) is roughly of the same order as the square root of the number of iterations at \( \omega = 0 \).

A convincing experimental proof that the P-condition number is minimised to a value proportional to the square root of its original value at \( \omega = 0 \) is given by the earlier table and graphs.
CHAPTER 5

ANALYSIS OF VARIATIONAL METHODS
5.1 INTRODUCTION

The fundamental problems of mechanics are known to be governed both by differential equations and by so-called minimum energy principles. Thus, for instance, the equilibrium position of a mechanical system is the position corresponding to the minimum of its potential energy. In view of this, the problem of solving the boundary value problem for the differential equation of the given mechanical system, generally turns out to be equivalent to the problem of finding the function giving the minimum of the integral by which the potential energy is expressed. Mathematically, the problem of solving a boundary value problem for a differential equation is equivalent to that in variational calculus of minimizing the integral for which the given differential equation is the Euler-Lagrange equation. Hence, the methods which are based on the substitution of an equivalent variational method in the boundary value problem for the differential equation may be employed. The approximate solution of the boundary value problem is then the approximate solution of the corresponding variational problem. The exposition of such methods, the most familiar of which are the methods of Ritz, will be the object of this chapter. We shall consider the application of these methods to P.D.E.'s.

Definition 5.1.1

A functional is "an operator that converts functions into numbers (or scalars)". The typical expression of a functional is,

\[ J(U) = \int_0^L \phi(U, U', x) \, dx \]  

(5.1.1)

where \( U(x) \) plays the role of the independent variable (it is a function of \( x \in [0, L] \)) and \( J \) is the dependent variable (a scalar).

Example 5.1.1

Consider the path joining two points A and B whose horizontal distance from one another is \( L \) (Figure 5.1.1). The length of the path is given by
length = \int_{0}^{\ell} [1+(U')^2]^{\frac{1}{2}} \, dx \quad (5.1.2)

which is a simple functional of the form (5.1.1)

Consider the functional

$$J(U) = \int_{0}^{\ell} \phi(U, U', x) \, dx \quad (5.1.3)$$

It takes a definite value for any function (curve) \( U = U(x) \) given for \( 0 < x < \ell \). We propose to seek the difference

$$\delta J \equiv J(U+\delta U) - J(U) \quad (5.1.4)$$

corresponding to a "small" variation \( \delta U(x) \) in the argument \( U \) (Figure 5.1.2).

\[ \delta J \text{ is called the variation of } J. \text{ Using Taylor's Theorem, we may write,} \]

$$J(U+\delta U) = \int_{0}^{\ell} \phi(U+\delta U, U'+\delta U', x) \, dx$$

$$= \int_{0}^{\ell} \left( \phi(U, U', x) + \frac{\partial \phi}{\partial U} \delta U + \frac{\partial \phi}{\partial U'} \delta U' \right) dx + O(\delta U^2) \quad (5.1.5)$$

Variation of a Functional and Extremal Curves
(We shall neglect the term $O(\delta U^2)$ since we are interested in $\delta J$ when $\delta U \to 0$).

Therefore,

$$J(U+\delta U) = \int_0^L \left[ \phi(U,U(x)) + \frac{3\phi}{\delta U} \delta U + \frac{3\phi}{\delta U} \frac{d}{dx}(\frac{\delta U}{U}) \right] dx,$$

(5.1.6)

The last term of (5.1.6) may be integrated by parts, i.e.,

$$\int_0^L \frac{3\phi}{\delta U} \frac{d}{dx}(\frac{\delta U}{U}) dx = \left[ \frac{3\phi}{\delta U} \right]_0^L - \int_0^L \frac{d}{dx} \left( \frac{3\phi}{\delta U} \right) \delta U dx.$$

(5.1.7)

Then, from (5.1.4) we obtain,

$$\delta J = \int_0^L \left[ \frac{3\phi}{\delta U} - \frac{d}{dx} \left( \frac{3\phi}{\delta U} \right) \right] \delta U(x) dx + \left[ \frac{3\phi}{\delta U} \right]_0^L \delta U dx.$$

(5.1.8)

$$= \int_0^L \left[ \frac{3\phi}{\delta U} - \frac{d}{dx} \left( \frac{3\phi}{\delta U} \right) \right] \delta U(x) dx.$$

(5.1.9)

**Definition 5.1.1**

When $U(x)$ is such that $J(U)$ reaches a maximum or minimum, then $U(x)$ is called an *extremal function* or *extremal curve* of the functional $J(U)$.

A necessary condition for $U(x)$ to be extremal is that $\delta J=0$ for *any* small, permissible displacement $\delta U(x)$. Returning to (5.1.9), we see that this condition will require that,

$$\frac{3\phi}{\delta U} - \frac{d}{dx} \left( \frac{3\phi}{\delta U} \right) = 0,$$

(5.1.10)

everywhere in $x \in [0,L]$.

Equation (5.1.10) is called the Euler-Lagrange equation for the given extremal problem.

Thus, the minimum distance of the Example 5.1.1 may be formulated as a variational problem where,

$$\phi = [1+(U')^2]^{-\frac{3}{2}}.$$  

(5.1.11)

The corresponding Euler-Lagrange equation is,

$$\frac{d}{dx} \left( \frac{3\phi}{2U} \right) = U''[1+(U')^2]^{-3/2} = 0,$$

(5.1.12)

of which one solution is

$$U'' = 0,$$

i.e.

$$U' = \text{constant},$$

or

$$U(x) = a+bx \equiv \text{straight line}. $$

(5.1.13)
The other solution of (5.1.12) is

\[ [1+(u')^2]^{-3/2} = 0, \quad (5.1.14) \]

which does not have a finite solution and corresponds to the maximum of distance.

We shall now generalise this result to cover the following cases:

1) In two independent variables,

\[ J = \int_{\mathcal{R}} \phi(U, \frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, x, y) \, dx \, dy, \quad (5.1.15) \]

where \( \mathcal{J} \) is a functional of the function \( U(x, y) \). The domain \( \mathcal{R} \) of \( U \) in the \( X \times Y \) plane is given in Section (1.1). The necessary condition is

\[ \frac{\partial \phi}{\partial U} - \frac{\partial^2 \phi}{\partial x \partial U} - \frac{\partial^2 \phi}{\partial y \partial U} = 0. \quad (5.1.16) \]

2) Higher derivatives. Suppose the variational problem,

\[ J(U) = \int_{\mathcal{R}} \phi(x, U, U', U'', \ldots, U^{(n)}) \, dx, \quad (5.1.17) \]

under the conditions,

\[ U'(0) = U'_o, \quad U'(0) = U'_o, \quad \ldots, U^{(n-1)}(0) = U^{(n-1)}_o, \quad (5.1.18) \]

\[ U(t) = U_e, \quad U'(t) = U'_e, \quad \ldots, U^{(n-1)}(t) = U^{(n-1)}_e, \quad (5.1.19) \]

is posed, then the corresponding Euler-Lagrange equation will be of order \( 2n \), having the form,

\[ \frac{\partial \phi}{\partial U} - \frac{\partial^2 \phi}{\partial x \partial U} + \frac{\partial^2 \phi}{\partial x^2 \partial U} - \ldots + (-)^n \frac{\partial^n \phi}{\partial x^n \partial U} = 0. \quad (5.1.20) \]

Before we proceed with the approximate methods intended to be discussed in this Chapter, we shall require some important definitions.

**Definition 5.1.2**

A linear space (or vector space) \( S \) is a set of elements (or vectors) such that if \( x, y \in S \) and \( a, b \) are scalars (complex, in general), then \( ax + by \in S \); moreover, these operations obey the usual rules of algebra, i.e.,

\[ x + y = y + x, \quad x + (y + z) = (x + y) + z, \quad (z \in S) \]

\[ a(x + y) = ax + ay, \quad (a + b)x = ax + bx, \]

\[ a(by) = (a\beta)x, \quad 1 \cdot x = x \]
if \( x + y = x + z \), then \( y = z \)

there exists an unique element \( 0 \in S \), such that

\[ 0.x = 0.y = 0 \] (zero element).

**Definition 5.1.3 (Metric Spaces)**

A metric space \( X \) (not necessarily a linear space) is a set of elements, such that for any pair \( x,y \in X \) a real non-negative "distance function" \( d(x,y) \) is defined, satisfying the axioms:

i. \( d(x,y) \geq 0 \) 

(5.1.2la)

ii. \( d(x,y) = 0 \) iff \( x = y \) 

(5.1.2lb)

iii. \( d(x,y) = d(y,x) \) (symmetric) 

(5.1.2lc)

iv. \( d(x,y) \leq d(x,z) + d(z,y) \) 

(5.1.2ld)

**Definition 5.1.4**

A metric space \( X \) is called complete if any fundamental sequence \( \{x_n\} \) (of element from \( X \)) has a limit \( x \) belonging to the space \( X \).

**Definition 5.1.5**

A metric space \( X \) is called separable if it contains a countable set \( C \) which is dense in \( X \) (i.e. if it contains a sequence \( P \) such that each element \( x \in X \) can be considered as a limit of a subsequence suitably selected from \( P \)).

**Definition 5.1.6**

A set \( H \) is called a Hilbert space if:

1. \( H \) is a linear space (vector space)

2. To each pair of elements \( x,y \in H \), there is a uniquely assigned complex number \( (x,y) \) called inner (scalar) product of the elements \( x \) and \( y \) such that

\[ (x,y) = \overline{(y,x)} \] (complex conjugate) 

(5.1.22a)

\[ (x_1 + x_2, y) = (x_1, y) + (x_2, y) , \]

(5.1.22b)

\[ (\alpha x, y) = \alpha (x, y) , \]

(5.1.22c)
3. The norm of an element \( x \) is defined by the formula \( ||x|| = \sqrt{(x,x)} \), while \( (x,x) = 0 \) iff \( x = 0 \); 

4. \( H \) is a complete space in the sense of the metric \( d(x,y) = ||x-y|| \),

5. \( H \) is separable.

Spaces \( E_n \) and \( L_2(a,b) \) are Hilbert spaces of the scalar (inner) product is defined as follows,

\[
\text{in } E_n (x,y) = \sum_{i=1}^{\infty} x_i \overline{y_i} , \quad \text{(5.1.23)}
\]

\[
\text{in } L_2 (x,y) = \int_a^b x(t) \overline{y(t)} \, dt \quad \text{(5.1.24)}
\]

In this thesis, we shall only consider the scalar product as defined by (5.1.24), where the vector functions are real.

We will be applying (5.1.24) to the boundary value problems of second order P.D.E.'s (of elliptic type)

\[
AU = f , \quad \text{(5.1.25)}
\]

where \( A \) is the operator defined by,

\[
AU = - \frac{\partial}{\partial x} \left( p \frac{\partial U}{\partial x} \right) - \frac{\partial}{\partial y} \left( p \frac{\partial U}{\partial y} \right) + qU , \quad \text{(5.1.25a)}
\]

where \( p \) is a positive continuously-differentiable function in the region \( \mathbb{R}+C \) and \( q \) and \( f \) continuous functions in \( \mathbb{R}+C \) (\( q>0 \)). We will also consider the Dirichlet boundary condition,

\[
U|_C = 0 . \quad \text{(5.1.25b)}
\]

We need only consider these zero boundary conditions because the general case will only alter the right hand side of (5.1.25) to give us the boundary value problem of an equation with zero initial conditions [Mikhlin, 1964, p.622].

By using (5.1.24) we will show that the operator \( A \) is positive when \( q(x,y)\neq0 \) in the set of functions \( M \) whose elements are twice continuously-differentiable functions in \( \mathbb{R}+C \) which satisfy the boundary conditions (5.1.24a).

In fact,
\[(AU, U) = \int\int_R U U \, dxdy\]

\[= - \int\int_R \left[ \frac{\partial}{\partial y} \left( p \frac{\partial U}{\partial x} \right) + \frac{\partial}{\partial x} \left( p \frac{\partial U}{\partial y} \right) \right] dxdy + \int\int_R q(x, y) U^2 \, dxdy\]

\[= \int\int_R \left[ p \left( \frac{\partial U}{\partial x} \right)^2 + \left( \frac{\partial U}{\partial y} \right)^2 \right] - \frac{\partial}{\partial x} (pU \frac{\partial U}{\partial x}) - \frac{\partial}{\partial y} (pU \frac{\partial U}{\partial y}) dxdy + \int\int_R qU^2 \, dxdy\]

\[= \int\int_R \left[ p \left( \frac{\partial U}{\partial x} \right)^2 + \left( \frac{\partial U}{\partial y} \right)^2 \right] + qU^2 \, dxdy - \int_C pU \frac{\partial U}{\partial n} \, dc , \quad (5.1.26)\]

where \( n \) is an outward normal to \( C \).

But \( U|_C = 0 \), and

\[(AU, U) = \int\int_R \left[ p \left( \frac{\partial U}{\partial x} \right)^2 + \left( \frac{\partial U}{\partial y} \right)^2 \right] + qU^2 \, dxdy \geq 0 . \quad (5.1.27)\]

**Definition 5.1.7**

An operator \( A \) is called symmetric if for any element \( U \) and \( V \) from the field of definition of this operator the identity,

\[(AU, V) = (U, AV) , \quad (5.1.28)\]

is valid.

**Definition 5.1.8**

A symmetric operator \( A^+ \) defined on some Hilbert space is said to be **positive** if for any element in the domain of existence of this operator we have the inequality

\[(AU, U) \geq 0 , \quad (5.1.29)\]

where equality occurs, iff \( U = 0 \).

If \( A \) is a positive operator, the scalar product \( (AU, U) \) is called the energy of the element \( U \) relative to the operator \( A \).

**Definition 5.1.9**

A symmetric operator \( A \) is said to be **positive definite** if there exists

---

\(^+\) Example of \( A \) is Laplace operator denoted by \( \Delta \)
a positive constant \( \gamma^2 \) such that for any element \( U \) in the domain of existence of the operator \( A \) we have the inequality,

\[
(AU, U) \geq \gamma^2 |U|^2.
\]  

(5.1.30)

**Energy Space**

With any positive operator, we can associated a special Hilbert space, which is called the *energy space*. Let \( M \) be the domain of some Hilbert space. If \( U \) and \( V \) are elements of \( M \), we set

\[
[U,V] = (AU,V) ,
\]  

(5.1.31)

The quantity \([U,V]\) is called the *energy product* of the element \( U \) and \( V \).

We denote the new Hilbert space constructed in this manner the *energy space* and denote it by \( H_A \). The *energy norm* is given by the formula,

\[
|U| = \sqrt{(AU,U)} .
\]  

(5.1.32)

**Energy Method**

If the operator \( A \) is positive, solution of (5.1.25) can be reduced to the solution of some variational problem as the following theorem implies.

**Theorem 5.1.2**

Let \( A \) be a positive operator. If the equation \( AU = f \) has a solution, this solution minimises the functional,

\[
J(U) = (U, AU) - 2(U,f) .
\]  

(5.1.33)

Conversely, if there exists an element that minimises function (5.1.33) this element satisfies the equation \( AU = f \).

The method of solving boundary-value problems consists of replacing (5.1.25) by the problem of minimizing the functional in (5.1.33) and is called the energy method [Mikhlin, 1965].

**Definition 5.1.10:**

The sequence \( \{u_n\} \), \( n=1,2,\ldots \) of functions belonging to the field of definition of the functional \( J(U) \) is known as the *minimising sequence* if,
If equation (5.1.25) has a solution then any sequence which is minimizing for the functional (5.1.34) converges in energy to this solution.

**Proof**

See [Mikhlin, 1964].
5.2 THE RITZ METHOD

The Ritz method is probably the most well known of the methods of constructing a minimizing sequence. Theorem 5.1.2 gives that the solution of $Au = f$ minimizes the functional (5.1.34).

For the actual determination of the sequence $\{u_n\}$, Ritz proposed the following: In the space $H_A$ we select a sequence of elements,

$$Q_1, Q_2, \ldots, Q_n, \ldots$$

(5.2.1)

that satisfy the following two conditions:

1) for any $n$, the elements $Q_1, Q_2, \ldots, Q_n$ are linearly independent.

2) the sequence (5.2.1) is complete in energy: i.e.,

For any $u_n \in H_A$ and any $\varepsilon > 0$, there exist a natural number $N$ and constants $a_1, a_2, \ldots, a_n$ such that

$$|u_n - \sum_{k=1}^{N} a_k Q_k| < \varepsilon$$

The elements (5.2.1) are called coordinate elements. We construct a linear combination of the first $n$ coordinate functions,

$$u_n = \sum_{k=1}^{n} a_k Q_k$$

(5.2.2)

where $a_k$ are constants to be determined so that $J(u_n)$ in (5.1.34) is a minimum.

When boundary condition (5.1.25a) is not homogeneous and is of the form,

$$u|_C = g(x,y)$$

(5.2.2a)

we represent the approximate solution in the form,

$$u_n = \psi + \sum_{k=1}^{n} a_k Q_k$$

(5.2.2b)

where $\psi$ satisfies (5.2.2a) and the $Q_k$ are functions in $H_A$ that vanish on the boundary $C$ and satisfy the boundary condition (5.1.25b).

We will however derive the Ritz system for the homogeneous boundary condition only. The non-homogeneous boundary condition follows essentially the same procedure.
We substitute \( u_n \) for \( U \) in the functional (5.1.33) which makes \( J(U) \) into a function of \( n \) independent variables \( a_1, a_2, \ldots, a_n \),

\[
J(u_n) = \left[ \sum_{j=1}^{n} a_j Q_j \right] - 2 \left[ \sum_{k=1}^{n} a_k R_k \right] - 2 \left[ \sum_{j=1}^{n} a_j f_j \right]
\]

\[
= \sum_{j,k=1}^{n} (Q_j R_k) a_j a_k - 2 \sum_{j=1}^{n} (f_j R_j) a_j . \tag{5.2.3}
\]

The coefficient \( a_j \) are selected so that the function (5.2.3) is a minimum.

The necessary conditions are

\[
\frac{\partial J(u_n)}{\partial a_i} = 0 ; \quad i = 1, 2, \ldots, n. \tag{5.2.4}
\]

We rewrite (5.2.3) as a three-termed quadratic expression in \( a_i \) and that by separating the terms which contain \( a_i \) to the first or second power, we can reduce \( J(u_n) \) to the form,

\[
J(u_n) = (Q_1 R_1) a_1 + \sum_{k \neq 1} (Q_k R_k) a_k a_k + \sum_{j \neq 1} (Q_j R_j) a_j a_1 - 2(f_1 R_1) a_1 + \ldots \tag{5.2.5}
\]

the dots denoting the group of terms not containing \( a_1 \). Since the positive definite operator is symmetric \( (Q_k R_1) = (Q_1 R_k) = (Q_1 R_k) \), and in the second summation we may write \( k \) instead of \( j \). Thus

\[
J(u_n) = (Q_1 R_1) a_1 + 2 \sum_{k \neq 1} (Q_k R_k) a_k a_k - 2(f_1 R_1) a_1 + \ldots \tag{5.2.6}
\]

Now,

\[
\frac{\partial J(u_n)}{\partial a_1} = 2(Q_1 R_1) a_1 + 2 \sum_{k \neq 1} (Q_k R_k) a_k - 2(f_1 R_1) , \tag{5.2.7}
\]

and combining the first term with the summation, we obtain,

\[
\frac{\partial J(u_n)}{\partial a_1} = 2 \sum_{k=1}^{n} (Q_k R_k) a_k - 2(f_1 R_1) . \tag{5.2.8}
\]

Finally, equating the derivatives to zero, we arrive at the system of Ritz equations,

\[
\sum_{k=1}^{n} (Q_k R_k) a_k = (f_1 R_1) ; \quad i = 1, 2, \ldots, n \tag{5.2.9}
\]
or
\begin{align*}
\begin{bmatrix}
\mathcal{A}_{Q_1, Q_1} a_1 + \mathcal{A}_{Q_1, Q_2} a_2 + \ldots + \mathcal{A}_{Q_1, Q_n} a_n &= (f, \mathcal{Q}_1), \\
\mathcal{A}_{Q_2, Q_1} a_1 + \mathcal{A}_{Q_2, Q_2} a_2 + \ldots + \mathcal{A}_{Q_2, Q_n} a_n &= (f, \mathcal{Q}_2), \\
\cdots \\
\mathcal{A}_{Q_n, Q_1} a_1 + \mathcal{A}_{Q_n, Q_2} a_2 + \ldots + \mathcal{A}_{Q_n, Q_n} a_n &= (f, \mathcal{Q}_n).
\end{bmatrix} \\
\text{(5.2.9a)}
\end{align*}

Using the concept of the energy product, we have \((\mathcal{A}_{Q_i, Q_k}) = [\mathcal{Q}_i, \mathcal{Q}_k]\) and the system (5.2.9a) assumes the form,
\begin{align*}
\begin{bmatrix}
[\mathcal{Q}_1, \mathcal{Q}_1] a_1 + [\mathcal{Q}_2, \mathcal{Q}_1] a_2 + \ldots + [\mathcal{Q}_n, \mathcal{Q}_1] a_n &= (f, \mathcal{Q}_1) \\
[\mathcal{Q}_1, \mathcal{Q}_2] a_1 + [\mathcal{Q}_2, \mathcal{Q}_2] a_2 + \ldots + [\mathcal{Q}_n, \mathcal{Q}_2] a_n &= (f, \mathcal{Q}_2) \\
\cdots \\
[\mathcal{Q}_1, \mathcal{Q}_n] a_1 + [\mathcal{Q}_2, \mathcal{Q}_n] a_2 + \ldots + [\mathcal{Q}_n, \mathcal{Q}_n] a_n &= (f, \mathcal{Q}_n)
\end{bmatrix} \\
\text{(5.2.9b)}
\end{align*}

Because \(\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_n\) are linearly independent functions, the determinant of the Ritz system is non-zero and the system has a unique system. A sequence of approximate solutions constructed with the Ritz method is a minimizing sequence (Mikhlin, 1964). As \(n\) increases, the energy norms of the approximate solutions given by the Ritz method increase (more accurately, do not decrease) and approach the energy form of the exact solution
\[ |\bar{u}_k| \leq |\bar{u}_n|, \quad k < n \lim_{n \to \infty} |u_n| = |u_0| \]
Moreover, we have the identity
\[ |u_0 - \bar{u}_n|^2 = |u_0|^2 - |u_n|^2. \] 

(5.2.10)

We agree to estimate the error of the approximate solution \(u_n\) by the quantity \(|u_0 - \bar{u}_n|\). It follows from (5.2.10) that the error of the approximate solution given by the Ritz method decreases (or more accurately, does not increase) as new coordinate elements are added.

Example 5.2.1

Asymmetric viscous flow in a channel
This example concerns the viscous flow between two parallel plates AB and CD with prescribed velocity distributions on AD and BC (Figure 5.2.1).

The governing equation is,
\[ \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0, \tag{5.2.11} \]
and the boundary conditions are,
\[ U(0,y) = \cos \frac{\pi}{2} y, \quad U(1,y) = e^{-\pi/2} \cos \frac{\pi}{2} y, \quad U(x, \pm 1) = 0. \tag{5.2.12} \]

The solution is simplified if a new velocity, \( u_d \) is introduced as
\[ U = u_d + e^{-\pi/2} \cos \frac{\pi}{2} y. \tag{5.2.13} \]

The governing equation for \( u_d \) is,
\[ \frac{\partial^2 u_d}{\partial x^2} + \frac{\partial^2 u_d}{\partial y^2} - e^{-\pi/2} \frac{\pi^2}{4} \cos \frac{\pi}{2} y = 0, \tag{5.2.14} \]
and the boundary conditions are,
\[ u_d(0,y) = (1-e^{-\pi/2}) \cos \frac{\pi}{2} y, \quad u_d(1,y) = 0, \quad u_d(x, \pm 1) = 0. \tag{5.2.15} \]

We represent an approximation \( u_6(x,y) \) for \( u_d \) by
\[ u_6(x,y) = (1-e^{-\pi/2}) \cos \frac{\pi}{2} y \cdot (1-x) + x(1-x)(1-y^2) \cdot \sum_{k,l=0,1,2}^{6} a_{k,l} x^k y^l, \tag{5.2.16} \]

The first term in (5.2.16) is chosen to satisfy the non-zero boundary condition; consequently the second term must be chosen to satisfy the homogeneous boundary conditions. Equation (5.2.16) may be rewritten as,
\[ u_6(x,y) = \psi(x,y) + \sum_{k=1}^{6} a_k Q_k(x,y), \tag{5.2.17} \]
where $\psi = (1-e^{-\pi/2}) \cdot \cos \frac{\pi}{2} y(1-x)$

$$Q_1 = x(1-x)(1-y)^2, \quad Q_2 = x^2(1-x)(1-y^2), \quad Q_3 = x(1-x)y^2(1-y^2)$$

$$Q_4 = x^3(1-x)(1-y^2), \quad Q_5 = x^2(1-x)y^2(1-y^2), \quad Q_6 = x^2(1-x)y^4(1-y^2)$$  \hspace{1cm} (5.2.17a)

In our case the energy product is given by the formula

$$[u,v] = \int_{-1}^{1} \int_{0}^{1} v(x,y) \Delta U(x,y) \, dx \, dy,$$  \hspace{1cm} (5.2.18)

then integration by parts gives the formula a symmetric form:

$$[u,v] = \int_{-1}^{1} \int_{0}^{1} \left( \frac{\partial U}{\partial x} \frac{\partial v}{\partial x} + \frac{\partial U}{\partial y} \frac{\partial v}{\partial y} \right) \, dx \, dy,$$  \hspace{1cm} (5.2.19)

and $(f,U)$ is given by,

$$\int_{-1}^{1} \int_{0}^{1} f \, U \, dx \, dy,$$

where $f = -\Delta \psi = -\frac{\pi}{4} \cdot \cos \frac{\pi}{2} y \cdot [1-(1-\exp^{-\pi/2})x]$  \hspace{1cm} (5.2.20)

Then, using (5.2.9), we obtain the following Ritz system:

<table>
<thead>
<tr>
<th></th>
<th>0.4444</th>
<th>0.2222</th>
<th>0.0686</th>
<th>0.1321</th>
<th>0.0343</th>
<th>0.0246</th>
<th>a₁</th>
<th>-0.2563</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.2222</td>
<td>0.1676</td>
<td>0.0343</td>
<td>0.1225</td>
<td>0.0254</td>
<td>0.0123</td>
<td>a₂</td>
<td>-0.1114</td>
</tr>
<tr>
<td></td>
<td>0.0686</td>
<td>0.0343</td>
<td>0.0449</td>
<td>0.0203</td>
<td>0.0224</td>
<td>0.0255</td>
<td>a₃</td>
<td>-0.0350</td>
</tr>
<tr>
<td></td>
<td>0.1321</td>
<td>0.1225</td>
<td>0.0203</td>
<td>0.1020</td>
<td>0.0184</td>
<td>0.0073</td>
<td>a₄</td>
<td>-0.0601</td>
</tr>
<tr>
<td></td>
<td>0.0343</td>
<td>0.0254</td>
<td>0.0224</td>
<td>0.0184</td>
<td>0.0148</td>
<td>0.0127</td>
<td>a₅</td>
<td>-0.0152</td>
</tr>
<tr>
<td></td>
<td>0.0246</td>
<td>0.0123</td>
<td>0.0255</td>
<td>0.0073</td>
<td>0.0127</td>
<td>0.0192</td>
<td>a₆</td>
<td>-0.0113</td>
</tr>
</tbody>
</table>

Solving these simultaneous equations by Crout's factorisation, we obtain,

$$a_1 = -0.7743, a_2 = 0.4232, a_3 = 0.1735, a_4 = -0.1157, a_5 = 0.0684 \text{ and } a_6 = -0.0114.$$  \hspace{1cm} (5.2.21)

Substitution of these values of $a_k$ into equations (5.2.16) and (5.2.13) gives an approximate solution. A comparison of this solution with the exact solution,

$$w = e^{-\pi/2} x \cdot \cos \pi/2 y,$$  \hspace{1cm} (5.2.22)

and the weighted residual solution is shown in Table 5.7.1.

For details of application of the Ritz method to problems in mathematical physics, see for example (Mikhlin, 1965) and (Mikhlin, 1971).
5.3 METHODS FOR SOLVING THE RITZ SYSTEM

We write (5.2.9) as a single vector equation,

\[ Ru = b, \quad (5.3.1) \]

where \( R \) is positive definite. Many methods can be used to solve systems of linear algebraic equations with positive definite matrices. We will consider only an iteration method. It is convenient to write equation (5.3.1) in the equivalent form,

\[ u = (I - \alpha R)u + \alpha b, \quad (5.3.2) \]

where \( I \) is the \( n \)th-order identity matrix and \( \alpha \) is any non-zero number.

The eigenvalues \( \lambda_1^{(n)}, \lambda_2^{(n)}, ..., \lambda_n^{(n)} \) of the matrix \( R \) are all positive, and we arrange them in increasing order:

\[ 0 < \lambda_1^{(n)} < \lambda_2^{(n)} < ... < \lambda_n^{(n)}. \]

The eigenvalues of the matrix \( I - \alpha R \) are equal to \( 1 - \alpha \lambda_k^{(n)} \) \((k=1,2,\ldots,n)\); if we set [Mikhlin, 1971]

\[ \alpha = \frac{2}{\lambda_1^{(n)} + \lambda_n^{(n)}}, \quad (5.3.3) \]

all eigenvalues of the matrix \( I - \alpha R \) will lie between the numbers,

\[ \frac{\lambda_n^{(n)} - \lambda_1^{(n)}}{\lambda_n^{(n)} + \lambda_1^{(n)}} \quad \text{and} \quad \frac{\lambda_n^{(n)} - \lambda_1^{(n)}}{\lambda_n^{(n)} + \lambda_1^{(n)}} \]

and consequently, will all have an absolute value less than one. Thus, the iteration with formula

\[ u^{(k+1)} = (I - \frac{2}{\lambda_1^{(n)} + \lambda_n^{(n)}} R)u^{(k)} + \frac{2}{\lambda_1^{(n)} + \lambda_n^{(n)}} b \quad (5.3.4) \]

converges to the solution of the Ritz system with the speed of a geometric mean,

\[ \frac{\lambda_n^{(n)} - \lambda_1^{(n)}}{\lambda_n^{(n)} + \lambda_1^{(n)}} = \frac{P-1}{P+1}, \quad (5.3.5) \]

where \( \lambda_n^{(n)}/\lambda_1^{(n)} \) is the \( P \)-condition number of the matrix \( R \).

If \( n \) increases, \( \lambda_1^{(n)} \) does not increase, and \( \lambda_n^{(n)} \) does not decrease.

Thus, it can be assumed that at least one of the relations,
holds, then (5.3.5) approaches 1 as $n \to \infty$. Hence, at the large values of $n$ required for the Ritz approximation to be sufficiently accurate, the iteration process converges very slowly.

A very important practical case is the one in which all of the eigenvalues of the matrix $A$ lie between two positive constants that do not depend on $n$:

$$0 < c_1 \leq \lambda_{(n)}^{(n)} \leq c_2 < \infty \quad (5.3.6)$$

In this case we can take,

$$\alpha = \frac{2}{c_1 + c_2} \quad (5.3.7)$$

and the iteration proceeds according to the formula

$$u^{(k+1)} = (I - \frac{2}{c_1 + c_2} R)u^{(k)} + \frac{2}{c_1 + c_2} \frac{c_2 - c_1}{c_2 + c_1}$$

with the speed of a geometric mean

$$\frac{c_2 - c_1}{c_2 + c_1} \quad (5.3.8)$$

and which converges for any value of $n$.

To accelerate the convergence rate we use the preconditioning technique as discussed in Chapter 4.
5.4 THE BUBNOV-GALERKIN METHOD

Assume that we have an equation

\[ AU = f, \quad (5.4.1) \]

where \( A \) is a linear operator defined in some Hilbert space \( H \), where no assumption is made about \( A \) except that it is linear. We select a sequence of coordinate functions \( \{ Q_n \}, Q_n \in D_A \) which are continuously differentiable in the closed domain \( \overline{\Omega} = \Omega + \Gamma \) as many times as are required for the specified problem and which satisfy all boundary conditions of our problem. As usual we denote the boundary of the domain by \( \Gamma \). We attempt to find an approximate solution in the form,

\[ u_n = \sum_{k=1}^{n} a_k Q_k. \quad (5.4.2) \]

According to the Bubnov-Galerkin method the coefficients \( a_k \) are determined from the requirement that the left hand side of (5.4.1) after substituting \( u_n \) for \( u \), is orthogonal to the functions \( Q_1, Q_2, \ldots, Q_n \) [Mikhlin, 1964]. This leads to the following systems of linear equations in the unknowns \( a_k \),

\[ \sum_{k=1}^{n} (A Q_k, Q_j) a_k = (f, Q_j), \quad j=1, 2, \ldots, n \quad (5.4.3) \]

System (5.4.3) is identical in form to the system (5.2.9) to which the Ritz method leads. Hence it is readily concluded that the Bubnov-Galerkin and Ritz method are identical if the operator \( A \) is symmetric (self-adjoint).

For non-symmetric and non-linear operators one can apply the Bubnov-Galerkin method whereas the Ritz method is inapplicable (Pinder G.F. and Gray W.G., 1977).

Here we shall discuss the Bubnov-Galerkin method upon the assumption that the operator \( A \) in the equation (5.4.1) has the form

\[ A = A_0 + K, \quad (5.4.4) \]

where \( A_0 \) is an operator which is symmetric and positive-bounded below in some Hilbert space \( H \). We shall assume the field of definition of the operator \( K \) is wider than the field of definition of the
operator $A_0$ (i.e. $D_\mathbb{K} \supset D_{A_0}$), so that the expression $Ku$ is meaningful whenever the expression $A_0 u$ is meaningful.

We introduce the space $H_0$ in which the scalar product equals

$$[U, V] = (A_0 U, V).$$

We shall select a sequence of coordinate elements $\{Q_n\}$ within the field of definition of the operator $A_0$ and is complete in $H_0$.

Equation (5.4.3) in our case has the form

$$\sum_{k=1}^{n} \{(A_0 Q_k, Q_j) + (KQ_k, Q_j)\}a_k = (f, Q_j), \quad j=1,2,\ldots,n$$

(5.4.6)

The expression $A_0 U = f$ can be represented in the form,

$$A_0 U + KU = f.$$

(5.4.7)

By applying the operator $A_0^{-1} = G$ to both sides of equation (5.4.7) we obtain the new equation,

$$U + TU = f', \quad f' = GF,$$

(5.4.8)

where $T = A_0^{-1} K$.

**Theorem 5.4.1**

The approximate solutions of equations (5.4.7) constructed by the Bubnov-Galerkin method converge in the energy of operator $A_0$ (in the sense of convergence in the space $H_0$) to the exact solution of this equation if the following conditions are satisfied:

1. Equation (5.4.7) has not more than one solution in $H_0$.
2. The operator $T$ is completely continuous in $H_0$.

**Proof:** see (Mikhlin, 1964)

We shall now consider the Bubnov-Galerkin method when applied to the Dirichlet problem for a second-order elliptic equation. Consider the equation with (in general) variable coefficients,

$$- \sum_{i,j=1}^{m} \frac{\partial}{\partial x_i} (A_{ik} \frac{\partial U}{\partial x_k}) + \sum_{i=1}^{m} B_i \frac{\partial U}{\partial x_i} + DU = f(x)$$

(5.4.9)

with the boundary conditions of the Dirichlet problem,

$$U|_{\Gamma} = 0.$$
We shall consider that equation (5.4.9) is elliptic non-degenerate. This implies the existence of a constant $u_0$ such that, for any point $P \in \mathbb{R}$ and for any real numbers $t_1, t_2, \ldots, t_m$ the following inequality holds:

$$
\sum_{i,k=1}^{m} A_{ik} t_i t_k \geq u_0 \sum_{i=1}^{m} t_i^2.
$$

We suppose further that,

1. the coefficients $A_{ik}$ and their first derivatives are continuous in $\mathbb{R}$ and that the coefficients $B_i$ and $D$ are continuous in $\mathbb{R}$.

2. the function $f(P)$ has a finite norm.

3. the problem posed has a unique solution.

We put,

$$
A_0 U = -\sum_{i,k=1}^{m} \frac{\partial}{\partial x_i} (A_{ik} \frac{\partial U}{\partial x_k}) ; KU = \sum_{i=1}^{m} B_i \frac{\partial U}{\partial x_i} + DU.
$$

It has been established (Mikhlin, 1964, p.24) that the operator $A_0$ is positive-bounded below for the set of functions which vanish on $\mathbb{R}$. For such functions the inequality,

$$
(A_0 U, U) \geq \gamma \|U\|^2, \quad \gamma > 0
$$

holds. We now introduce the space $H_0$ by putting,

$$
[U,V] = (A_0 U, V) = \int_{\mathbb{R}} \sum_{i,k=1}^{n} A_{ik} \frac{\partial U}{\partial x_i} \frac{\partial V}{\partial x_k} dR
$$

For this problem, the Bubnov-Galerkin method converges [see Mikhlin, 1964, p.482].

**Example 5.4.1**

Consider the equation,

$$
\nabla^2 U + x \frac{\partial U}{\partial x} = 2x^2 + 2y^2 + 2x^2 y^2 - 2x^2 y^2 + xy^2 + xy - 2x - 2y.
$$

It has the solution,

$$
U = xy(x-1)(y-1),
$$

which vanishes on the boundary of the square,

$$
x = 0, \quad x = 1, \quad y = 0, \quad y = 1,
$$

As the coordinate functions, we take,

$$
\sin k\pi x \sin \ell \pi y ; \quad k, \ell = 1, 2, 3, \ldots
$$
which satisfies (5.4.14). We restrict ourselves to terms for which \( k+2 \geq 4 \)
and put,

\[
u_6 = a_1 \sin \pi x \sin \pi y + a_2 \sin 2\pi x \sin \pi y + a_3 \sin \pi x \sin 2\pi y + a_4 \sin 3\pi x \sin \pi y + a_5 \sin 2\pi x \sin 2\pi y + a_6 \sin \pi x \sin 3\pi y
\]

(5.4.16)

The Bubnov-Galerkin equations in this case are as follows:

\[
\begin{bmatrix}
\frac{1}{2} + \frac{1}{8} & \frac{1}{3} & 0 & -\frac{3}{16} & 0 & 0 \\
-\frac{1}{3} & \left(\frac{5\pi^2}{4} + \frac{1}{8}\right) & 0 & \frac{3}{5} & 0 & 0 \\
0 & 0 & \left(\frac{5\pi^2}{4} + \frac{1}{8}\right) & 0 & \frac{1}{3} & 0 \\
\frac{3}{16} & -\frac{3}{5} & 0 & \left(\frac{5\pi^2}{2} + \frac{1}{8}\right) & 0 & 0 \\
0 & 0 & -\frac{1}{3} & 0 & \left(2\pi^2 + \frac{1}{8}\right) & 0 \\
0 & 0 & 0 & 0 & 0 & \left(\frac{5\pi^2}{2} + \frac{1}{8}\right)
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6
\end{bmatrix}
= \begin{bmatrix}
\frac{4}{6}(9\pi^2 - 8) \\
\frac{2}{\pi} \\
0 \\
\frac{4}{27\pi}(49\pi^2 - 8) \\
\frac{4}{27\pi}(41\pi^2 - 8)
\end{bmatrix}
\]

The solution of this system by Crout's factorization method is,

\[
a_1 = 0.066653; \quad a_2 = 0.000014499; \quad a_3 = 0;
\]

\[
a_4 = 0.0024528; \quad a_5 = 0; \quad a_6 = 0.0024648.
\]

Substituting these values in (5.4.16), we obtain an approximation for \( u_6 \)
as shown in the Table 5.4.1

<table>
<thead>
<tr>
<th>(x,y)</th>
<th>(0.0,7)</th>
<th>(0.25,.5)</th>
<th>(0.25,0.75)</th>
<th>(0.5,.25)</th>
<th>(0.5,.75)</th>
<th>(.75,.25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-Galerkin</td>
<td>0.0000</td>
<td>0.0471</td>
<td>0.0357</td>
<td>0.0471</td>
<td>0.0471</td>
<td>0.0357</td>
</tr>
<tr>
<td>Exact solution</td>
<td>0.0000</td>
<td>0.0469</td>
<td>0.0352</td>
<td>0.0469</td>
<td>0.0469</td>
<td>0.0352</td>
</tr>
</tbody>
</table>

**TABLE 5.4.1**

The problem of integrating (5.4.9) for the Neumann and the mixed boundary conditions of a second order elliptic equation is discussed in Mikhlin, 1964.
5.5 TREFFTZ METHOD

The idea of Trefftz's method consists in looking for approximate solutions, from among the appropriate class of functions that satisfy exactly the differential equation, but not necessarily satisfy the prescribed boundary conditions. It is in contrast to the Ritz method, where approximate solutions satisfy the boundary conditions but not necessarily the differential equation. We will restrict ourselves here to the case of Dirichlet problem for the following equation,

\[ AU = - \frac{\partial}{\partial x} \left( p \frac{\partial U}{\partial x} \right) - \frac{\partial}{\partial y} \left( p \frac{\partial U}{\partial y} \right) + qU = 0 , \quad (5.5.1) \]

in the finite region \( R \) bounded by boundary \( C \) and where \( p \) is a positive continuously-differentiable function in the region \( R+C \), and \( q \) continuous functions in \( R+C \) \((q \geq 0)\), although the Trefftz method can also be applied to other problems.

**Definition 5.5.1**

A function \( U(x,y) \) which satisfies the equation

\[ \Delta U = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0 , \quad (5.5.2) \]

on a region \( R \) of the \( XOY \) plane is said to be harmonic.

We have to find, in the region \( R \), a harmonic function satisfying the boundary condition,

\[ U \big|_C = g . \quad (5.5.2a) \]

Let us consider a sequence of linearly independent functions \( Q_n \), harmonic in \( R \), which is complete in the following sense. For an arbitrary function \( Q \), harmonic in \( R \) and square integrable together with its first derivatives, and for an arbitrary \( c>0 \), there exists a linear combination

\[ \sum_{k=1}^{n} a_k Q_k \]

of function \( Q_k \), such that

\[ \int_{R} \left[ \left( \frac{\partial}{\partial x} \left( \sum_{k=1}^{n} a_k Q_k \right) \right)^2 + \left( \frac{\partial}{\partial y} \left( \sum_{k=1}^{n} a_k Q_k \right) \right)^2 \right] dx dy < \epsilon \]

As before, we represent an approximate solution in the form,
\[ u_n(x,y) = \sum_{k=1}^{n} a_k Q_k \]  

(5.5.3)

We have to make \( a_k \) \((k=1, 2, \ldots, n)\) such that the functions \( u_n(x,y) \) in some sense satisfy the boundary conditions (5.5.2a) most closely. For example, we can make \( a_k \) \((k=1, 2, \ldots, n)\) such that the integral,

\[ J_n(u_n) = \int_C u_n^2(x,y) \, ds \]

takes the lowest value. Therefore we would get a set of linear algebraic equations to find \( a_1, a_2, \ldots, a_n \),

\[ \frac{\partial J_n^2}{\partial a_k} = 2 \int_C u_n a_k \, ds = 0, \quad (k=1, 2, \ldots, n), \]

(5.5.4)

In Trefftz's method it is required that the difference between \( u_n \) and the exact solution \( u \) should minimize the functional,

\[ J_3(u) = \iint_R \left\{ p \left[ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right] + qu^2 \right\} \, dxdy \]

(5.5.5)

i.e. the coefficient \( a_k \) must be made such that the integral

\[ \phi = \iint_R \left\{ p \left[ \frac{\partial (u_n - U)}{\partial x} \right]^2 + \frac{\partial (u_n - U)}{\partial y} \right\} \, dxdy \]

(5.5.6)

be a minimum.

In this case, \( a_k \) \((k=1, 2, \ldots, n)\) must be the solution of the set,

\[ \frac{\partial \phi}{\partial a_k} = 2 \iint_R \left\{ p \left[ \frac{\partial (u_n - U)}{\partial x} \right] + \frac{\partial (u_n - U)}{\partial y} \right\} + q(u_n - U)Q_k \] \, dxdy \]

(5.5.7)

Using Ostrogradski's formula [Berezin I.S. & Zhidicov N.P., 1965] (5.5.7) can be transformed into

\[
\iint_R \left\{ - \frac{\partial}{\partial x} \left( p \frac{\partial Q_k}{\partial x} \right) - \frac{\partial}{\partial y} \left( p \frac{\partial Q_k}{\partial y} \right) + qQ_k \right\} (u_n - U) \, dxdy + \\
\iint_R \frac{\partial}{\partial x} \left[ p \frac{\partial Q_k}{\partial x} (u_n - U) \right] + \frac{\partial}{\partial y} \left[ p \frac{\partial Q_k}{\partial y} (u_n - U) \right] \, dxdy
\]

\[
= \iint_R A_k (u_n - U) \, dxdy + \int_C \left[ \frac{\partial Q_k}{\partial x} \cos nx + p \frac{\partial Q_k}{\partial y} \cos ny \right] (u_n - U) \, dC. \quad (5.5.8)
\]

Since \( A_k = 0 \) and \( U \mid_C = g \), we can rewrite the set (5.5.7) in the form,
\[
\int_C p \frac{\partial Q_k}{\partial n} u_n \, dc = \int_C \frac{\partial Q_k}{\partial n} g \, dc, \quad k=1,2,\ldots,n
\]  
(5.5.9)

where \( n \) is the outward normal to \( C \). This set does not contain \( U(x,y) \).

Substituting (5.5.3) into (5.5.9) we obtain,
\[
\sum_{j=1}^{n} a_j \int_C \frac{\partial Q_j}{\partial n} \, dc = \int_C \frac{\partial Q_k}{\partial n} \, dc, \quad (k=1,2,\ldots,n)
\]  
(5.5.10)

By solving we find \( a_1, a_2, \ldots, a_n \) and consequently \( u_n(x,y) \).

The difficulty of finding a complete system of harmonic functions in concrete cases is a major defect of the Trefftz method. For a two-dimensional simply connected region \( R \) with a sufficiently smooth boundary, harmonic polynomials form a complete system. In the case of multiply connected regions and in higher dimensions, the problem is difficult and there are still many unsolved problems in this field [Rektorys, 1969].

**Example 5.5.1**

Let us solve the boundary value problem
\[
\Delta U = -1, \\
\text{in the rectangle } R(|x|<2,|y|<1) \text{ and } \\
U = 0, \\
\text{on the boundary.}
\]

We transform the given problem by the substitution \( U=U-\frac{1}{4}(x^2+y^2) \) to the problem,
\[
\Delta \tilde{U} = 0, \\
\text{with the boundary } \tilde{U} = \frac{1}{4}(x^2+y^2).
\]

We choose two even harmonic polynomials (since \( \tilde{U} \) is an even function in both arguments) as coordinate functions to give,
\[
\tilde{U}_2(x,y) = a_1(x^2-y^2) + a_2(x^4-xy^2+y^4)
\]  
(5.5.11)

The system (5.5.8) yields, in this case,
\[
\begin{bmatrix}
160 \\
192 \\
192
\end{bmatrix}
= 
\begin{bmatrix}
11904 \\
176 \\
15
\end{bmatrix}
\]
The solution is $a_1 = 0.211$, $a_2 = -0.017$; hence,
\[
\tilde{u}_2 = 0.211(x^2 - y^2) - 0.017(x^4 - 6x^2y^2 + y^4)
\]
and
\[
\tilde{u} = 0.211(x^2 - y^2) - 0.017(x^4 - 6x^2y^2 + y^4) - \frac{1}{2}(x^2 + y^2) + k
\]
The constant $k$ must be chosen in such a way that the mean value of $U$ on the boundary is zero, i.e. $k = -\frac{1}{C} \int_C UdC$. We find that $p = \frac{924}{2070} = 0.449$. Hence
\[
U = 0.449 + 0.211(x^2 - y^2) - 0.25(x^2 + y^2) - 0.017(x^4 - 6x^2y^2 + y^4)
\]
Better results are obtained if we use harmonic trigonometric functions, i.e. if we put even with one term approximation
\[
u_1(x,y) = a_1 \cos \frac{\pi x}{4} \cosh \frac{\pi y}{2}
\]  \hspace{1cm} (5.5.12)
By this choice of coordinate function Mikhlin (1964, p.412) shows that the result is also considerably more precise than that from the Ritz method. This is due to the selection of the coordinate function which approximates closely to the required harmonic function.
5.6 **KANTOROVICH'S METHOD**

This method is discussed in detail in the monograph by L.V. Kantorovich and V.I. Krylov, 19

For simplicity we investigate only the two-dimensional problem. The Kantorovich method is formally similar to the Rayleigh-Ritz method. The approximate solution is assumed to be of the form,

\[ u_n(x,y) = \sum Q_n(x,y)f_k(x) \quad (5.6.1) \]

where \( Q_n(x,y) \) are known functions satisfying the prescribed boundary conditions. The unknown functions \( f_k(x) \) are determined such that the functional \( J(u_n) \) attains its minimum value. Following a similar procedure as in the Rayleigh-Ritz method, we obtain for the unknown functions \( f_k \) a boundary value problem for a system of ordinarily differential equations in a certain interval \([a,b]\).

By this method, the solution of boundary value problem for P.D.E.'s in two dimensions is reduced to that of the solution of ordinary differential equations (O.D.E.'s). For this reason this method is not relevant within the content of our discussion. We will however give an example to illustrate the method.

**Example 5.6.1**

Let us solve the boundary value problem of Example 5.5.1. We look for a solution of the form,

\[ u_1 = (1-y^2)f(x) \quad (5.6.2) \]

This solution satisfies the boundary condition for \( y=\pm1 \). Substituting in the corresponding functional, we obtain,

\[ J(u_1) = \int \int_R \left[ \left( \frac{\partial u_1}{\partial x} \right)^2 + \left( \frac{\partial u_1}{\partial y} \right)^2 - 2u_1 \right] dx dy \]

\[ = \int_{-2}^{2} dx \int_{-1}^{1} \left[ (1-y^2)^2 f_x^2 + 4y^2 f_x^2 - 2f(1-y^2) \right] dy. \]

Integrating with respect to \( y \), we obtain,
The corresponding Euler equation is

\[ f'' - \frac{5}{2} f = -\frac{5}{4} \]  

with boundary conditions \( f(\pm 2) = 0 \).

The solution of this problem is

\[ f(x) = \frac{1}{2}(1 - \frac{\cosh(\frac{5}{2}x)}{\cosh(10)}) \]

Hence, the approximation \( u_1 \) is given by,

\[ u_1(x,y) = \frac{1}{2}(1-y^2) \left[ 1 - \frac{\cosh(\frac{5}{2}x)}{\cosh(10)} \right] \]

Remarks

If the given boundary conditions (of the two-dimensional problem) are homogeneous and if the boundary of the region considered includes the segments \( x = a, x = b \), then it is not necessary to satisfy the boundary conditions on these segments by the proper choice of function \( Q_k \), since we may satisfy them instead by prescribing them as boundary conditions for the functions \( f_k \) (equation 5.6.5a).

The advantage of the reduction of the original P.D.E.'s to a set of O.D.E.'s rather than to a set of algebraic equations is that the O.D.E.'s can often be solved analytically or are amenable to very efficient numerical algorithms (Fletcher, 1978). A comparable idea of reducing the original P.D.E.'s by finite difference methods has been used by Kaplan et al (1964).
5.7 THE METHOD OF WEIGHTED RESIDUALS

We have seen that the solution $u$ of a P.D.E. may be approximated by an expression of the form,

$$u_n = \sum_{k=1}^{n} a_k Q_k,$$  \hspace{1cm} \text{(5.7.1)}

when the $Q_k$, $k=1,2,\ldots$ are basis functions satisfying the prescribed boundary condition.

We now seek approximations of the form (5.7.1) by the use of principles other than those based on the calculus of variations.

As usual consider the Dirichlet boundary-value problem,

$$AU = f, \quad \text{in } \mathbb{R},$$ \hspace{1cm} \text{(5.7.2)}

subject to the boundary conditions,

$$u|_C = g,$$ \hspace{1cm} \text{(5.7.2a)}

on the boundary $C$.

The degree to which $u$ fails to satisfy (5.7.2) is expressed by the residual equation, defined by,

$$r = Au - f,$$ \hspace{1cm} \text{(5.7.3)}

where $u$ is a trial solution expressed in (5.7.1) with the subscript $n$ omitted. The smaller $r$, the better the approximation.

The essence of the weighted residual methods is to require that the unknown parameters, $a_k$ in equation (5.7.1) are chosen to minimize the residual $r$ in some sense. Different methods of minimizing the residual yield different approximate solutions, i.e. we require that $r$ be orthogonal to the weighting functions $\{w_j\}$. The $a_j$ are calculated by satisfying the constraints which arise when setting the weighted integrals of the residual to zero,

$$\int_{\mathbb{R}} w_j r \, d\mathbb{R} = 0, \quad (j=1,2,\ldots,n),$$ \hspace{1cm} \text{(5.7.4)}

\textbf{Collocation Method}

The trial function (5.7.1) is chosen to satisfy the boundary conditions.
Let \( \{P_j: j=1,2,\ldots,n\} \) be points in \( \mathbb{R} \) called collocation points. The weight functions are then chosen as,
\[
w_j = \delta(P-P_j)
\]
where \( \delta(P-P_j) \) are Dirac delta functions which have the property that
\[
\int_{\mathbb{R}} w_j x - x(P_j) = 0, \quad j=1,2,\ldots,n
\]
For an equation,
\[
A\bar{u}(P) = f(P)
\]
the residual is,
\[
\underline{r}(u) = A\left[ \sum_{k=1}^{n} Q_k(P) a_k \right] - f(P) - f(P_j)
\]
and the collocation conditions (5.7.7) here are,
\[
A\left[ \sum_{k=1}^{n} Q_k(P) a_k \right] - f(P_j) = 0, \quad j=1,2,\ldots,n.
\]
If the problem is linear, then (5.7.10) becomes,
\[
\sum_{k=1}^{n} A Q_k(P_j) a_k = f(P_j), \quad j=1,2,\ldots,n.
\]

**Example 5.7.1**

Again consider the problem of Example 5.2.1 with the same basis functions, i.e. \( \psi = (1-e^{-\pi/2}) \cdot \cos \frac{\pi}{2} y \cdot (1-x) \)
\[
Q_1 = x(1-x)(1-y^2), \quad Q_2 = x^2(1-x)(1-y^2), \quad Q_3 = x(1-x)y^2(1-y^2)
\]
\[
Q_4 = x^3(1-x)(1-y^2), \quad Q_5 = x^2(1-x)y^2(1-y^2), \quad Q_6 = x^2(1-x)y^4(1-y^2)
\]
Thus,
\[
A Q_1 = 2(x-x^2)-2(1-y^2), \quad A Q_2 = 2(1-3x)(1-y^2)-2x^2(1-x),
\]
\[
A Q_3 = -2(y^2-y^4)+2x(1-x)(1-y^2), \quad A Q_4 = (6x-12x^2)(1-y^2)-2x^3(1-x),
\]
\[
A Q_5 = (1-3x)(2y^2-2y^4)+2x^2(1-x)(1-6y^2), \quad A Q_6 = (1-3x)(2y^4-2y^6)+2x^2(1-x)(6y^2-15y^4)
\]
and \( f = -A\dot{\psi} = -\frac{\pi^2}{4} \cdot \cos \frac{\pi}{2} y \cdot [1-(1-e^{-\pi/2})x] \).
If we choose the collocation points to be distributed uniformly they are as follows:

\[ p_1 = \left( \frac{1}{7}, \frac{5}{7} \right), \quad p_2 = \left( \frac{2}{7}, \frac{3}{7} \right), \quad p_3 = \left( \frac{3}{7}, \frac{1}{7} \right), \quad p_4 = \left( \frac{4}{7}, \frac{1}{7} \right), \]
\[ p_5 = \left( \frac{5}{7}, \frac{3}{7} \right), \quad p_6 = \left( \frac{6}{7}, \frac{5}{7} \right), \]

then equation (5.7.11) leads to the following system of equations,

\[
\begin{bmatrix}
-1.2245 & 0.5248 & 0.2135 & 0.1427 & 0.0738 \\
-2.0408 & 0.1166 & 0.0309 & 0.1068 & 0.0401 \\
-2.4490 & 0.7697 & 0.1728 & 0.2699 & 0.0106 \\
-2.4490 & -1.6793 & 0.2171 & -0.6397 & 0.1306 & 0.0184 \\
-2.0408 & -2.1574 & -0.3725 & -1.7076 & -0.2906 & 0.0735 \\
-1.2245 & -1.7473 & -1.2181 & -1.9792 & -1.2889 & 0.0620
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6
\end{bmatrix} =
\begin{bmatrix}
0.9494 \\
1.4925 \\
1.5889 \\
1.3167 \\
0.8376 \\
0.3437
\end{bmatrix}
\tag{5.7.12}
\]

Again, using the method of Crout's factorization, the solution is

\[ a_1 = -0.6834, \quad a_2 = 0.1512, \quad a_3 = -0.0387, \quad a_4 = 0.1457 \]
\[ a_5 = -0.0111, \quad a_6 = 0.0028. \]

Thus the approximate solution for \( u_6 \),

\[
\begin{align*}
  u_6(x,y) &= (1-e^{-\pi/2}) \cos \frac{\pi y}{2} (1-x) - 0.6834 x(1-x)(1-y^2) \\
  &\quad + 0.1512 x^2(1-x)(1-y^2) - 0.0387 x(1-x) y^2 (1-y^2) \\
  &\quad + 0.1457 x^3 (1-x)(1-y^2) - 0.0111 x^2 (1-x) y^2 (1-y^2) + 0.028 x^2 \\
  &\quad (1-x^2) y^4 (1-y^2) .
\end{align*}
\]

From (5.2.13),

\[
U = u_6(x,y) + e^{-\pi/2} \cos \frac{\pi y}{2} . \tag{5.7.13}
\]

This approximation for \( U \) is shown in Table 5.7.1.

When \( m \) collocation points, where \( m > n \) are chosen, an overdetermined system of equations is obtained for the unknown parameters. System (5.7.11) then becomes,

\[
\sum_{k} A Q_k(P_j) a_k = f(P_j) , \quad j = 1, 2, \ldots, m \tag{5.7.14}
\]

These equations may then be solved by the method of least squares.

**Example 5.7.2**

Suppose in Example 5.7.1, the same approximation is used, but the
collocation points are taken to be \((x_i, y_i)\), where 
\[ x_i = \frac{0+1}{m+1}, \]
\[ y_i = \frac{-1+2}{m+1}, \quad i = 1, 2, \ldots, 9. \]
Then (5.7.14) yields the system

\[
\begin{bmatrix}
-0.9000 & 0.4860 & 0.2714 & 0.1710 & 0.1055 & 0.0666 \\
-1.6000 & 0.4480 & 0.1101 & 0.4480 & 0.1510 & 0.0625 \\
-2.1000 & 0.0420 & 0.0319 & 0.5670 & 0.0983 & 0.0373 \\
-2.4000 & -0.5760 & 0.1306 & 0.3840 & 0.0768 & 0.0173 \\
-2.5000 & -1.2500 & 0.2500 & -0.1250 & 0.1250 & 0.0000 \\
-2.4000 & -1.8240 & 0.1574 & -0.8640 & 0.1037 & 0.0362 \\
-2.1000 & -2.1420 & -0.2839 & -1.6170 & -0.2176 & 0.0824 \\
-1.6000 & -2.0480 & -0.9421 & -2.0480 & -0.9011 & -0.1946 \\
-0.9000 & -1.3860 & -1.2434 & -1.7010 & -1.4094 & -0.9729 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6 \\
\end{bmatrix}
= \begin{bmatrix}
0.7021 \\
1.2205 \\
1.5218 \\
1.6031 \\
1.4902 \\
1.2313 \\
0.8893 \\
0.5313 \\
0.2189 \\
\end{bmatrix}
\tag{5.7.15}
\]

This is an overdetermined set of algebraic equations of the form

\[
R u = h \tag{5.7.16}
\]

The usual method of least squares yields the following square set of equations

\[
R^t Ru = R^t h. \tag{5.7.17}
\]

Thus, (5.7.15) becomes,

\[
\begin{bmatrix}
33.3300 & 16.6650 & 1.4190 & 7.6065 & 1.8785 & 0.6473 \\
1.4190 & 3.7686 & 2.7054 & 4.5003 & 2.7692 & 1.4039 \\
7.6065 & 11.8341 & 4.5003 & 11.1634 & 4.6604 & 1.9562 \\
1.8785 & 3.9984 & 2.7692 & 4.6604 & 2.9216 & 1.5539 \\
0.6473 & 1.5565 & 1.4039 & 1.9562 & 1.5539 & 1.0026 \\
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6 \\
\end{bmatrix}
= \begin{bmatrix}
-19.2232 \\
-7.3765 \\
0.1240 \\
-2.0032 \\
-0.1357 \\
0.0090 \\
\end{bmatrix}
\tag{5.7.18}
\]

giving (again, by Crout's factorization method),

\[
a_1 = -0.6834, \quad a_2 = 0.1512, \quad a_3 = -0.0382, \quad a_4 = 0.1456, \quad a_5 = 0.0114, \quad a_6 = 0.0026.
\]
In this case the solution is the same as the solution by the collocation method (see Table 5.7.1).

From (5.7.7), we notice that the method consists of simply evaluating the residual at the collocation points and therefore involves a minimum of computational effort. As n increases the residual is zero at more and more points and presumably approaches zero everywhere. Unfortunately, the method depends greatly on the choice of the collocation points and consequently has not been widely used [Finlayson, 1972]. However Jains, 1962, has developed the method called "External point collocation method" for fluid flow problems which converges rapidly and gives sufficiently good results with low order approximations. Villadsen and Stewart (1967) have shown that a special case of this method known as orthogonal collocation permits rapid solution of many types of second-order boundary-value problems.

Least Squares

The method of least squares is applied directly to the residual. Again, the trial functions are chosen to satisfy the boundary conditions. The weighting function is $3r/3a_j$. This is equivalent to minimizing the integrated square residual,

$$J(u) = \int_R r(P)^2 \, dR,$$  \hspace{1cm} (5.7.19)

with respect to the trial parameters $a_k$, $k=1,2,\ldots$.

Thus,

$$\frac{\partial J}{\partial a_i} = 0, \; i=1,2,\ldots,n$$ \hspace{1cm} (5.7.20)

Now,

$$J(a_1,\ldots,a_n) = \int_R (A(\sum_{k=1}^n a_k Q_k - f))^2 \, dR$$

If $A$ is a linear operator, then,

$$\frac{\partial J}{\partial a_i} = 2 \int_R (A(\sum_{k=1}^n a_k Q_k) A Q_i - f A Q_i) \, dR,$$

so that from (5.7.20)
By writing it in the form of the Ritz matrix, (5.7.21) becomes,

\[
\begin{align*}
\sum_{k=1}^{n} a_k \int_{R} (A_{Q_k} A_{Q}) dR &= \int_{R} f dQ, \quad i=1, \ldots, n \\
(A_{Q_1} A_{Q_1}) a_1 + (A_{Q_2} A_{Q_1}) a_2 + \ldots + (A_{Q_n} A_{Q_1}) a_n &= (f, A_{Q_1}) \\
(A_{Q_1} A_{Q_2}) a_1 + (A_{Q_2} A_{Q_2}) a_2 + \ldots + (A_{Q_n} A_{Q_2}) a_n &= (f, A_{Q_2}) \\
&\vdots \\
(A_{Q_1} A_{Q_n}) a_1 + (A_{Q_2} A_{Q_n}) a_2 + \ldots + (A_{Q_n} A_{Q_n}) a_n &= (f, A_{Q_n}).
\end{align*}
\]  

(5.7.22)

We note that system (5.7.22) is symmetric. Mikhlin (1964, p.493) proves the following theorem.

**Theorem 5.7.1**

When the homogeneous equation \( A u = 0 \) has only a trivial solution (i.e., the inverse operator exists) approximate solution can be constructed by the method of least squares and they are unique. The method gives a sequence of approximate solutions which converges to the exact solution if the following conditions are satisfied:

1. The sequence of basis functions is \( A \)-complete (i.e. for every \( \varepsilon > 0 \), it is possible to find a positive integer \( n \) and constant \( \{a_k\} \) such that,

\[
||A u - A_n u|| < \varepsilon
\]  

(5.7.23)

2. Equation (5.7.8) is solvable,

3. There exists a constant \( K \) such that for any \( u \) in the field of definition of the operator \( A \) the following inequality holds:

\[
||u|| \leq K ||A u||
\]  

(5.7.24)

For a second order operator it must be possible to approximate second derivatives, whereas in the Rayleigh-Ritz method it was only necessary to approximate first derivatives.

**Example 5.7.1**

Again, the problem of Example 5.2.1 is considered with \( A_{Q_i} \), \( i=1,2,\ldots,6 \)
and \( f \) as given in Example 5.7.1.

Equation (5.7.22) thus gives,

\[
\begin{bmatrix}
6.3111 & 3.1556 & 2.3873 & 2.7429 & 1.1937 & 1.1556 \\
3.1556 & 5.6540 & 1.1937 & 4.8476 & 1.2597 & 0.5778 \\
2.3873 & 1.1937 & 3.0375 & 0.8305 & 1.5187 & 2.8251 \\
2.7429 & 4.8476 & 0.8305 & 5.6089 & 1.0781 & 0.3810 \\
1.1937 & 1.2597 & 1.0781 & 1.175 & 1.4125 \\
1.1556 & 0.5778 & 2.8251 & 0.3810 & 1.4125 & 3.4678 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
-3.7083 \\
-0.8041 \\
-1.1389 \\
-0.6776 \\
-0.3846 \\
-0.0156 \\
\end{bmatrix}
\]

(5.7.25)

giving, \( a_1 = -0.7982, \ a_2 = 0.4110, \ a_3 = 0.1648, \ a_4 = -0.0963 \)
\( a_5 = -0.0662 \) and \( a_6 = -0.0156. \)

Following a similar procedure, we substitute these values for \( u_6(x,y) \) and then for \( U \) as in Example 5.7.1. The solution obtained is shown in Table 5.7.1.

This method generally leads to complicated expressions but it is often used in error analysis since error bounds can be derived in terms of it.

Thus, minimization of (5.7.10) gives the best possible bounds for the error. Despite its cumbersome equations, the method has been applied to complicated problems arising in nuclear reactor engineering (Becker, 1964).

Mikhlin (1964, p.503) considers the problem

\[
\Delta U = f(x,y),
\]

and proves the inequality

\[
[U(x,y)-u_n(x,y)]^2 \leq \int_R G^2(x,y,\xi,\eta)d\xi d\eta \int_R (f-\Delta u_n)^2d\xi,
\]

(5.7.27)

where \( G \) is the Green's function. The first integral in (5.7.27) is bounded and the second integral is the quantity minimized by the least squares method. Thus, (5.7.27) provides uniform convergence of the approximate solution as well as pointwise error bounds. Furthermore, the error bounds are improved when more and more terms are included in the approximate
solution. Despite this advantage, the convergence of the least squares method is slower than the Rayleigh-Ritz method, especially when trigonometric polynomials are used as the basis functions.

**Galerkin (w.r.)-method**

It is to be noted here that the Bubnov-Galerkin method (Section 5.4) can be classified as the method of weighted residuals when the weighting functions are chosen from the same family as the trial functions, i.e.,

\[ w_j(P) = Q_j(P) . \]  

(5.7.28)

Since the trial functions are chosen from a linearly independent set of functions, equations (5.7.4) and (5.7.28) may be interpreted as requiring the residuals to be orthogonal to every member of the complete set up to order \( n \) (see Section 5.4), i.e.,

\[ \int_P Q_j \, dR = 0 , \quad j=1,2,\ldots,n . \]  

(5.7.29)

This yields the following \( n \) equations for the \( n \) parameters \( a_j \):

\[ \int_R (Au-f)Q_j \, dR = 0 , \quad j=1,2,\ldots,n . \]  

(5.7.30)

When \( A \) is linear, (5.7.30) becomes,

\[ \sum_{k=1}^{n} a_k \int_R Q_k A Q_j \, dR = \int_R f Q_j \, dR , \quad j=1,2,\ldots,n . \]  

(5.7.31)

Written in energy form, this last equation is identical to (5.4.3).

<table>
<thead>
<tr>
<th>( (x,y) )</th>
<th>(.3,1)</th>
<th>(.25,0)</th>
<th>(.5, -0.5)</th>
<th>(.5, 0)</th>
<th>(.5, 0.5)</th>
<th>(.75, 0)</th>
<th>(.0, 0.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rayleigh-Ritz Galerkin (w.r.)</td>
<td>0.0000</td>
<td>0.6752</td>
<td>0.3193</td>
<td>0.4559</td>
<td>0.3193</td>
<td>0.3079</td>
<td>0.7071</td>
</tr>
<tr>
<td>Collocation</td>
<td>0.0000</td>
<td>0.6826</td>
<td>0.3189</td>
<td>0.4611</td>
<td>0.3189</td>
<td>0.3144</td>
<td>0.7071</td>
</tr>
<tr>
<td>Overdetermined-collocation</td>
<td>0.0000</td>
<td>0.6825</td>
<td>0.3189</td>
<td>0.4611</td>
<td>0.3189</td>
<td>0.3144</td>
<td>0.7071</td>
</tr>
<tr>
<td>Least squares</td>
<td>0.0000</td>
<td>0.6716</td>
<td>0.3198</td>
<td>0.4558</td>
<td>0.3198</td>
<td>0.3141</td>
<td>0.7071</td>
</tr>
<tr>
<td>Exact solution</td>
<td>0.0000</td>
<td>0.6752</td>
<td>0.3224</td>
<td>0.4559</td>
<td>0.3224</td>
<td>0.3079</td>
<td>0.7071</td>
</tr>
</tbody>
</table>

**TABLE 5.7.1**: Approximation solution to the boundary-value problem of asymmetric viscous flow
ON THE CHOICE OF BASIS FUNCTIONS

It is obvious that when using the approximate methods, the basic problem is the choice of sequence of coordinate function \( \{ Q_n \} \) which is complete with respect to the operator \( A \). If such a choice is made, the approximate solution will approach the exact solution as the value of \( n \) increases. This is true in general. If the knowledge of the form of the solution is available, the coordinate functions can be chosen appropriately and an acceptable solution may be found by using only a few parameters. If prior knowledge of the form of the solution is unavailable, a poor choice of coordinate functions may occur. The result is slow convergence. We will not dwell on details of the theoretical discussion here, but we will give some examples of such complete systems, the proofs of which are given in the literature referenced later in the chapter.

Sequence of Trigonometric Functions

If the given region is a rectangle \((0,a) \times (0,b)\), we choose,

\[
Q_{kn}(x,y) = \sin \frac{k \pi x}{a} \sin \frac{\ell \pi y}{b}, \quad k, \ell = 1, 2, \ldots
\]  

(5.8.1)

All functions of this sequence satisfy the boundary conditions

\[
Q_{kn}(0,y) = Q_{kn}(a,y) = Q_{kn}(x,0) = Q_{kn}(x,b) = 0,
\]  

(5.8.2)

and the same holds for all their even derivatives:

\[
\frac{\partial^{2n} Q_{kn}}{\partial x^{2n}}(0,y) = \frac{\partial^{2n} Q_{kn}}{\partial x^{2n}}(a,y) = \frac{\partial^{2n} Q_{kn}}{\partial x^{2n}}(x,0) = \frac{\partial^{2n} Q_{kn}}{\partial x^{2n}}(x,b) = 0 \quad (5.8.3)
\]

It can be shown that the sequence chosen in such a manner is complete for differential operators (in two variables) of the second order as far as the boundary conditions are of the indicated form (Mikhlin, 1964, Ch.8). If the boundary conditions are of a different form, it is necessary to modify \( \{ Q_n \} \) or even to use functions other than trigonometric functions.

Example 5.8.1

To the operator of the second order with boundary conditions,
there corresponds the choice,
\[ Q_{k\ell}(x,y) = \sin \frac{(2k-1)\pi x}{2a} \sin \frac{\ell \pi y}{b}, \quad k, \ell = 1, 2, \ldots \] (5.8.4)

**Sequences using Polynomials**

We consider the Dirichlet problem in some domain \( R \) for Poisson’s equation under the condition that the required function vanishes on the boundary \( C \) of the domain \( R \).

Let \( w(x, y) \) be a continuous function which vanishes on the boundary \( C \) and is positive at points within the domain \( R \); we further assume that this function has bounded and continuous first derivatives within \( R \). Then the system of functions,
\[ Q_n(x, y) = w(x, y) x^k y^\ell, \quad k, \ell = 1, 2, \ldots \] (5.8.5)
is complete within \( R \) [Mikhlin, 1964].

The construction of the function \( w(x, y) \) usually encounters no difficulty. Thus, for example, for the rectangle referred to above, we can take \( w = xy(a-x)(b-y) \). [Kantorovich & Krylov, 1964, p.276].

If the contour bounding the region \( R \) admits an equation of the form \( F(x,y) = 0 \), where \( F \) is a function continuous together with its partial derivatives, then one can adopt \( w = \pm F \). Thus, for example, for a circle of radius \( r \) with the centre at the origin, we can take \( w = r^2 - x^2 - y^2 \). Kantarovich & Krylov (1964) indicate some general rules for the construction of the function \( w(x, y) \) for a more complex region \( R \).
CHAPTER 6

THE PRECONDITIONED VARIATIONAL METHOD AND

FINITE ELEMENT METHOD
From the discussion of Chapter 4, it is clear that the central issue in solving the ill-conditioned system,

\[ Au = b , \]  

is the need to improve the condition of the original system in order to ensure that the rounding errors involved will be minimised if a direct method of solution is used and to obtain the fastest rate of convergence if an iterative method of solution is used.

In Chapter 5 it was shown that the solution of certain partial differential equations, together with the prescribed boundary and/or initial conditions can be obtained using the variational methods. The Rayleigh-Ritz method, for example, provide an algorithm for minimizing a given functional and requires the choice of a suitable complete set of linearly independent basis functions \( \Phi_i, \ i=1,2,\ldots \). The exact solution, \( u \), is approximated by a sequence of trial functions,

\[ u_n = \sum_{i=1}^{n} a_i \Phi_i , \]  

or by (5.2.2b), where the constants \( a_i \) are chosen to minimize \( J[u_n] \). In the case where a suitable functional does not exist or cannot be found, we have chosen other methods of solution. A similar approximation to \( u \) is also made, where the \( a_i \) are chosen in the case of the methods of weighted residuals, to minimise the norm of the residual,

\[ r = AU - f , \]  

in some sense.

The desire to obtain accurate solutions using the variational methods poses an immediate problem. As the number of terms, \( n \), in the basis functions (6.1.2) is increased, the coefficient matrix \( A \) in (6.1.1) become large and the solution of (6.1.1) becomes inefficient and inaccurate. This is because, the matrix \( A \) is dense and becomes ill-conditioned for large \( n \).

The basic aim of this chapter is therefore to compare the condition
numbers for these various variational methods so that we can select the 'best' method to use based on the condition numbers of the coefficient matrix of the preconditioned matrix which is to be discussed later. Once the best method has been determined, different forms of basis functions are then chosen. To achieve this objective, the obtained system (6.1.1) is first transformed to its symmetric form with unit diagonal entries, i.e.,
\[ D^{-1/2} (D^L u) = D^{-1/2} b. \] (6.1.4)

If \( L \) and \( L^T \) are lower and upper triangular matrices respectively, then (6.1.4) becomes,
\[ \hat{A} \hat{v} = (I-L-L^T) \hat{v} = d, \] (6.1.5)
where \( \hat{L}, \hat{L}^T, \hat{d} \) and \( \hat{v} \) are defined to be \( D^{-1/2} L D^{-1}, D^{-1/2} L D^{-1}, D^{-1/2} d \) and \( D^{-1/2} u \) respectively. Following the procedure of Chapter 4, the preconditioned form of (6.1.5) is,
\[ (I-\omega L)^{-1} \hat{A} (I-\omega L^T)^{-1} z = b_\omega \] (6.1.6)
or
\[ B_\omega z = b_\omega, \] (6.1.7)
where, \( z = (I-\omega L)^T \hat{v} \) and \( b_\omega = (I-\omega L)^{-1} d \).

Since the preconditioned system (6.1.7) is symmetric and positive definite, we can apply the basic iterative methods of Chapter 3 to this preconditioned system. In this chapter, we use the Preconditioned Simultaneous Displacement method, given by,
\[ z^{(k+1)} = z^{(k)} + a (B_\omega - B) z^{(k)}. \] (6.1.8)
The optimum values of \( a \) is given by (3.3.35) except that this time (3.3.35) denotes the eigenvalue bounds for the eigenvalues of the positive definite preconditioned matrix \( B_\omega \). The iterations proceed in the \( z \)-variable until an accuracy of \( \varepsilon = 0.5 \times 10^{-6} \) is achieved for every component of the solution vector. From the discussion in Section 5.3, it was shown that (6.1.8) will converge to the solution of (6.1.1).

It was shown in Chapter 5, Section 7, that when the operator \( A \) is linear, then the algebraic equations obtained for the Galerkin method (5.7.25)
are identical with those obtained using the Rayleigh-Ritz method (5.4.3).
Since the Rayleigh-Ritz method does not require the second order derivative
of the approximate function as the Galerkin method does, we will therefore
in this chapter only need to consider the Rayleigh-Ritz method in obtaining
the algebraic equation (6.1.1).

A double precision arithmetic was used throughout this chapter to ensure
that the integration's were evaluated accurately (within the round-off error
of the computer). In this chapter the integrations were evaluated
numerically using Householder reduction and the maximum and minimum eigenvalues
of the matrix $B$ were determined by use of the Peterson methods using the
NAG library.
6.2 PROBLEM ONE

We consider a simple two-point boundary-value problem for which we know the exact solution so that the approximate solutions can be compared with the exact solution. Such an example is,

$$-\frac{\partial^2 u}{\partial x^2} = x^3, \quad 0 < x < 1,$$

with

$$u(0) = u(1) = 0,$$

which has the exact solution,

$$u_0(x) = \frac{x}{20} (1-x^4).$$

The set of independent basis functions in (6.1.2) is taken to be a set of polynomials. As the coordinate functions, we take,

$$x(l-x)x_k^k, \quad k = 0, 1, 2, \ldots$$

which satisfies (6.2.1a). We restrict ourselves to terms for \( k \leq 2 \), thus an approximation \( u_3(x) \) for \( u \) is given by,

$$u_3(x) = a_0 x(l-x) + a_1 x^2(l-x) + a_2 x^3(l-x).$$

In this case, \( \Omega_1 = x(l-x), \quad \Omega_2 = x^2(l-x), \) and \( \Omega_3 = x^3(l-x) \).

The Rayleigh-Ritz Method

From the theory presented earlier in Chapter 5 then using the one-dimensional form of (5.2.9) and (5.2.20), we have,

$$[\Omega_1, \Omega_j] = \int_0^1 \frac{\partial \Omega_1}{\partial x} \frac{\partial \Omega_j}{\partial x} \, dx, \quad i, j = 0, 1, 2.$$

and,

$$\langle f, \Omega_i \rangle = \int_0^1 x^3 \Omega_i^t \, dx.$$  

From (6.2.4), we obtain the entries of \( A = (a_{ij}) \) as follows:

$$a_{11} = \int_0^1 (1-2x)^2 \, dx = \frac{1}{3}, \quad a_{12} = a_{21} = \int_0^1 (1-2x)(2x-3x^2) \, dx = \frac{1}{6},$$

$$a_{22} = \int_0^1 (2x-3x^2)^2 \, dx = \frac{2}{15}, \quad a_{23} = a_{32} = \int_0^1 (2x-3x^2)(3x^2-4x^3) \, dx = \frac{1}{10},$$

$$a_{13} = a_{31} = \int_0^1 (1-2x)(3x^2-4x^3) \, dx = \frac{1}{10},$$

$$a_{33} = \int_0^1 (3x^2-4x^3)^2 \, dx = \frac{3}{35}, \quad b_1 = \int_0^1 x^3(x-x^2) \, dx = \frac{1}{30}, \quad b_2 = \int_0^1 x^3(x^2-x^3) \, dx = \frac{1}{42}.$$
Thus (6.1.1) becomes,
\[
\begin{bmatrix}
1/3 & 1/6 & 1/10 \\
1/6 & 2/15 & 1/10 \\
1/10 & 1/10 & 3/35 \\
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix}
= 
\begin{bmatrix}
1/30 \\
1/42 \\
1/56 \\
\end{bmatrix}
\]
(6.2.6)

When we solve for \(a_0\), \(a_1\) and \(a_2\) by Crout's factorisation method, we obtain,
\[a_0 = 0.0535714, \quad a_1 = 0.0178571, \quad a_2 = 0.125000.\]

Thus, the Rayleigh-Ritz approximation is given by,
\[a_3(x) = 0.0535714x(1-x) + 0.0178571x^2(1-x) + 0.125000x^3(1-x).\] (6.2.7)

A similar procedure is carried out for \(n=2\) and \(n=4\). A comparison between \(u_n(x), n=2,3,\) and \(4\) and \(u_0(x)\) for the Rayleigh-Ritz method is given in Table 6.2.1.

<table>
<thead>
<tr>
<th>(x)</th>
<th>Approximate solution, (u_n \times 10)</th>
<th>Exact solution (u_0 \times 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.2)</td>
<td>0.0914286</td>
<td>0.0994826</td>
</tr>
<tr>
<td>(0.4)</td>
<td>0.205714</td>
<td>0.193714</td>
</tr>
<tr>
<td>(0.6)</td>
<td>0.274286</td>
<td>0.262286</td>
</tr>
<tr>
<td>(0.8)</td>
<td>0.228571</td>
<td>0.236571</td>
</tr>
</tbody>
</table>

| TABLE 6.2.1: Comparisons of solutions by approximate Rayleigh-Ritz method with the exact solution |

Table 6.2.2 gives the largest and smallest eigenvalues, P-condition number, the number of iterations required and the N and M-condition numbers of the preconditioned Rayleigh-Ritz matrix for \(n=3\). Figure 6.2.1 shows the P-condition numbers plotted for various values of the preconditioning parameter \(\omega\) for the different methods of approximation. A comparison of the condition numbers and the number of iterations required to achieve an accuracy of \(0.5 \times 10^{-6}\) for the Simultaneous Displacement method for values of the preconditioning parameter \(\omega = 0\) (without preconditioning) and \(\omega = \omega_{opt}\) (optimal preconditioning) is given in Table 6.2.9. Finally in Figure 6.2.2 we compare the relative errors for the four approximations used.
### TABLE 6.2.2: Tabulation of the extreme eigenvalues, condition numbers for the coefficient matrix \( B_w \) and the number of iterations required (to achieve an accuracy of \( 5 \times 10^{-6} \)) for the Rayleigh-Ritz method with \( n=3 \).

<table>
<thead>
<tr>
<th>( \omega )</th>
<th>Smallest eigenvalues</th>
<th>Largest eigenvalues</th>
<th>P-Condition Number</th>
<th>Iteration</th>
<th>N-condition Numbers</th>
<th>M-condition Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.231232D-01</td>
<td>0.2553466D+01</td>
<td>0.1104287D+03</td>
<td>639</td>
<td>0.3737081D+02</td>
<td>0.7800000D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2823364D-01</td>
<td>0.1893175D+01</td>
<td>0.6705388D+02</td>
<td>394</td>
<td>0.2307019D+02</td>
<td>0.5857500D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.3467899D-01</td>
<td>0.1453266D+01</td>
<td>0.4190637D+02</td>
<td>252</td>
<td>0.1480413D+02</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4253880D-01</td>
<td>0.1181791D+01</td>
<td>0.2778149D+02</td>
<td>171</td>
<td>0.1009571D+02</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.5139043D-01</td>
<td>0.1041540D+01</td>
<td>0.2026719D+02</td>
<td>126</td>
<td>0.7465111D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.5975584D-01</td>
<td>0.1000000D+01</td>
<td>0.1673477D+02</td>
<td>104</td>
<td>0.6128105D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.6293964D-01</td>
<td>0.1010099D+01</td>
<td>0.1604870D+02</td>
<td>100</td>
<td>0.5834523D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.6496292D-01</td>
<td>0.1041556D+01</td>
<td>0.1603309D+02</td>
<td>99</td>
<td>0.5778462D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.6556783D-01</td>
<td>0.1097655D+01</td>
<td>0.1674076D+02</td>
<td>102</td>
<td>0.5976287D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.6469247D-01</td>
<td>0.1183371D+01</td>
<td>0.1829225D+02</td>
<td>111</td>
<td>0.6466473D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.6248742D-01</td>
<td>0.1304949D+01</td>
<td>0.2088338D+02</td>
<td>125</td>
<td>0.7313640D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.5925523D-01</td>
<td>0.1469671D+01</td>
<td>0.2480238D+02</td>
<td>146</td>
<td>0.8614722D+01</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.5111975D-01</td>
<td>0.1962853D+01</td>
<td>0.3839717D+02</td>
<td>219</td>
<td>0.1317920D+02</td>
<td>0.4500000D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.4263131D-01</td>
<td>0.2743628D+01</td>
<td>0.6435711D+02</td>
<td>356</td>
<td>0.2194638D+02</td>
<td>0.4500000D+02</td>
</tr>
</tbody>
</table>

The Collocation Method

From (5.7.11), we obtain,

\[
\begin{align*}
    a_{j1} &= A_0^j (P_j) = 2 \\
    a_{j2} &= A_1^j (P_j) = (6x-2) \\
    a_{j3} &= A_2^j (P_j) = (12x^2-6x) \\
    b_j &= x^3 \\
\end{align*}
\]

for \( j=1,2 \) and 3 \hspace{1cm} (6.2.8)

The points \( x_j = \frac{1}{(k+2)} \), \( j=1,2 \) and 3 are chosen to be the collocation points:

\[
\begin{bmatrix}
    2 & -1/2 & -3/4 \\
    2 & 1 & 0 \\
    2 & 5/2 & -9/4 \\
\end{bmatrix}
\begin{bmatrix}
    a_0 \\
    a_1 \\
    a_2 \\
\end{bmatrix}
= 
\begin{bmatrix}
    1/64 \\
    1/8 \\
    27/64 \\
\end{bmatrix}
\]  
(6.2.9)
Again, when we solve for \( a_0, a_1 \) and \( a_2 \) we obtain,

\[
a_0 = 0.057292, \quad a_1 = 0.010417 \quad \text{and} \quad a_2 = 0.125000.
\]

Then,

\[
u_3(x) = 0.057292x(1-x) + 0.010417 x^2(1-x) + 0.125x^3(1-x) .
\]

The comparison between \( u_n, n=2,3, \) and 4 with the exact solution is given in Table 6.2.3. Since not all the diagonal elements of the coefficient matrix \( A \) in (6.2.9) are positive, we evaluate the condition numbers of \( A^TA \) and the results obtained are shown in Table 6.2.4.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( n=2 )</th>
<th>( n=3 )</th>
<th>( n=4 )</th>
<th>Exact solution ( (\times10) ) ( u_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.071111</td>
<td>0.103000</td>
<td>0.0998400</td>
<td>0.0998400</td>
</tr>
<tr>
<td>0.4</td>
<td>0.168889</td>
<td>0.195500</td>
<td>0.194880</td>
<td>0.1948800</td>
</tr>
<tr>
<td>0.6</td>
<td>0.231111</td>
<td>0.260500</td>
<td>0.261120</td>
<td>0.261120</td>
</tr>
<tr>
<td>0.8</td>
<td>0.195556</td>
<td>0.233000</td>
<td>0.236160</td>
<td>0.236160</td>
</tr>
</tbody>
</table>

**TABLE 6.2.3:** Comparison of the solution by approximate collocation method with the exact solution.
<table>
<thead>
<tr>
<th>ω</th>
<th>Smallest E.V. eigenvalues</th>
<th>Largest eigenvalues</th>
<th>K-Condition Number</th>
<th>Iteration</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2604369d-01</td>
<td>0.2308698d+01</td>
<td>0.8861911d+02</td>
<td>601</td>
<td>0.3077965d+02</td>
<td>0.65000000d+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.3174035d-01</td>
<td>0.1791364d+01</td>
<td>0.5643806d+02</td>
<td>387</td>
<td>0.2023789d+02</td>
<td>0.47336000d+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.3877075d-01</td>
<td>0.1423031d+01</td>
<td>0.3670373d+02</td>
<td>254</td>
<td>0.1375969d+02</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4701917d-01</td>
<td>0.1177563d+01</td>
<td>0.2504432d+02</td>
<td>177</td>
<td>0.9828152d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.5569464d-01</td>
<td>0.1041458d+01</td>
<td>0.1869944d+02</td>
<td>134</td>
<td>0.7518213d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.6291794d-01</td>
<td>0.1000000d+01</td>
<td>0.1589372d+02</td>
<td>113</td>
<td>0.6329837d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.6516456d-01</td>
<td>0.1010098d+01</td>
<td>0.1550073d+02</td>
<td>109</td>
<td>0.6086220d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.6612218d-01</td>
<td>0.1041506d+01</td>
<td>0.1575123d+02</td>
<td>109</td>
<td>0.6071861d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.6567836d-01</td>
<td>0.1097179d+01</td>
<td>0.1570534d+02</td>
<td>114</td>
<td>0.6304126d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.6390777d-01</td>
<td>0.1181190d+01</td>
<td>0.1848273d+02</td>
<td>125</td>
<td>0.6819481d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.6104304d-01</td>
<td>0.1298200d+01</td>
<td>0.2126696d+02</td>
<td>141</td>
<td>0.7677100d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.5740196d-01</td>
<td>0.1453314d+01</td>
<td>0.2531819d+02</td>
<td>165</td>
<td>0.8963485d+01</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.4904808d-01</td>
<td>0.1901132d+01</td>
<td>0.3876058d+02</td>
<td>245</td>
<td>0.1333991d+02</td>
<td>0.45000000d+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.4079025d-01</td>
<td>0.2579870d+01</td>
<td>0.6322516d+02</td>
<td>392</td>
<td>0.2143039d+02</td>
<td>0.45000000d+02</td>
</tr>
</tbody>
</table>

**TABLE 6.2.4**: Tabulation of eigenvalues, condition numbers for the coefficient matrix \( B_0 \) and the number of iterations required to achieve an accuracy of \( 5 \times 10^{-6} \), for the collocation method

**Over-Determined Collocation Method**

From (5.7.14) we take \( m = 6 \), thus giving the collocation points as

\[
x_j = \frac{j}{7}, \quad j=1,2,3,4,5,6
\]

This choice of collocation points will lead to the over-determined system,

\[
\begin{bmatrix}
2.000 & -1.1429 & -0.6122 \\
2.000 & -0.2857 & -0.7347 \\
2.000 & 0.5714  & -0.3673 \\
2.000 & 1.4286  & 0.4098  \\
2.000 & 2.2857  & 1.8367  \\
2.000 & 3.1429  & 3.673 \\
\end{bmatrix}\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix} = \begin{bmatrix}
0.0029 \\
0.0233 \\
0.0787 \\
0.1866 \\
0.3644 \\
0.6297 \\
\end{bmatrix}
\]
The usual method of least squares yields the following set of equations,

\[
\begin{bmatrix}
24.2000 & 12.0000 & 8.5714 \\
12.0000 & 18.8571 & 17.1429 \\
8.5714 & 17.1429 & 18.1574 \\
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix}
= 
\begin{bmatrix}
2.5714 \\
3.1137 \\
3.0262 \\
\end{bmatrix}
\]
giving \( a_0 = 0.053912 \), \( a_1 = 0.017177 \), and \( a_2 = 0.12500 \).

Thus, the approximate solution is

\[
u_3(x) = 0.053912x(1-x) + 0.017177x^2(1-x) + 0.125x^3(1-x).
\]

The results obtained for this system are given in Tables 6.2.5 and 6.2.6.

<table>
<thead>
<tr>
<th>x</th>
<th>Approximate Solution, ( u_n \times 10^2 )</th>
<th>Exact Solution ( u_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n=2 )</td>
<td>( n=3 )</td>
</tr>
<tr>
<td>0.2</td>
<td>0.093440</td>
<td>0.099755</td>
</tr>
<tr>
<td>0.4</td>
<td>0.206720</td>
<td>0.193878</td>
</tr>
<tr>
<td>0.6</td>
<td>0.273280</td>
<td>0.261212</td>
</tr>
<tr>
<td>0.8</td>
<td>0.226560</td>
<td>0.236245</td>
</tr>
</tbody>
</table>

TABLE 6.2.5: Comparison of the solutions by approximate overdetermined collocation method with the exact solution
### TABLE 6.2.6: Tabulation of eigenvalues, condition numbers for the coefficient matrix $B_w$ and the number of iterations required to achieve an accuracy of $5 \times 10^{-6}$ for the overdetermined collocation method

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>K-condition number</th>
<th>Iteration number</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.5647790D-01</td>
<td>0.2294800D+01</td>
<td>0.4063190D+02</td>
<td>287</td>
<td>0.1413620D+02</td>
<td>0.2966662D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6829403D-01</td>
<td>0.1784588D+01</td>
<td>0.2613095D+02</td>
<td>187</td>
<td>0.9396900D+01</td>
<td>0.2632812D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.8238856D-01</td>
<td>0.1420000D+01</td>
<td>0.1723540D+02</td>
<td>124</td>
<td>0.6500634D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.9799697D-01</td>
<td>0.1176766D+01</td>
<td>0.1200819D+02</td>
<td>88</td>
<td>0.4779148D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.1126911D+00</td>
<td>0.1041427D+01</td>
<td>0.9241430D+01</td>
<td>69</td>
<td>0.3821607D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.1220383D+00</td>
<td>0.1000000D+01</td>
<td>0.8194151D+01</td>
<td>60</td>
<td>0.3409816D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.1232761D+00</td>
<td>0.1010098D+01</td>
<td>0.8193782D+01</td>
<td>60</td>
<td>0.3380845D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.1219097D+00</td>
<td>0.1041459D+01</td>
<td>0.8542874D+01</td>
<td>61</td>
<td>0.3470717D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.1181610D+00</td>
<td>0.1096710D+01</td>
<td>0.9281485D+01</td>
<td>65</td>
<td>0.3690042D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.1125380D+00</td>
<td>0.1178962D+01</td>
<td>0.1047612D+02</td>
<td>72</td>
<td>0.4058567D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.1056660D+00</td>
<td>0.1291264D+01</td>
<td>0.1220246D+02</td>
<td>82</td>
<td>0.4660734D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.9813668D-01</td>
<td>0.1436705D+01</td>
<td>0.1463984D+02</td>
<td>97</td>
<td>0.5377494D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.8286968D-01</td>
<td>0.1841506D+01</td>
<td>0.2222171D+02</td>
<td>143</td>
<td>0.7833628D+01</td>
<td>0.2432812D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.6899370D-01</td>
<td>0.2429386D+01</td>
<td>0.3521170D+02</td>
<td>224</td>
<td>0.1210713D+02</td>
<td>0.2432812D+02</td>
</tr>
</tbody>
</table>

The Least Squares Method

For this method of approximation, we seek the solution of the system (5.7.21). From (6.2.8) we have,

$$ A Q_1 = 2, \quad A Q_2 = 6x - 2, \quad \text{and} \quad A Q_3 = (12x^2 - 6x). $$

Therefore,

$$ a_{11} = (AQ_1, AQ_1) = 4, \quad a_{22} = (AQ_2, AQ_2) = 4, \quad a_{33} = (AQ_3, AQ_3) = 4.8, $$

$$ a_{12} = a_{21} = (AQ_1, AQ_2) = 2, \quad a_{13} = a_{31} = (AQ_1, AQ_3) = 2, $$

$$ a_{23} = a_{32} = (AQ_2, AQ_3) = 4, \quad b_1 = (f, AQ_1) = 5, $$

$$ b_2 = (f, AQ_2) = 7 \quad \text{and} \quad b_3 = (f, AQ_3) = 8. $$
Thus, (5.7.22) becomes,

\[
\begin{pmatrix}
4 & 2 & 2 \\
2 & 4 & 4 \\
2 & 4 & 4.8
\end{pmatrix}
\begin{pmatrix}
a_0 \\
a_1 \\
a_2
\end{pmatrix}
= 
\begin{pmatrix}
5 \\
7 \\
8
\end{pmatrix}
\] (6.2.11)

Again, by Crout's factorization method, we obtain:

\[a_0 = 0.050000, \quad a_1 = 0.025000 \text{ and } a_3 = 0.125000.\]

The approximate solution is then,

\[u_3(x) = 0.05x(1-x) + 0.025x^2(1-x) + 0.125x^3(1-x).\]

Similar to the previous methods, we give the results for this method in Tables 6.2.7 and 6.2.8.

<table>
<thead>
<tr>
<th>x</th>
<th>Approximate Solution, $u_n (\times 10)$</th>
<th>Exact Solution ($\times 10$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=2</td>
<td>n=3</td>
</tr>
<tr>
<td>0.2</td>
<td>0.128000</td>
<td>0.096000</td>
</tr>
<tr>
<td>0.4</td>
<td>0.264000</td>
<td>0.192000</td>
</tr>
<tr>
<td>0.6</td>
<td>0.336000</td>
<td>0.264000</td>
</tr>
<tr>
<td>0.8</td>
<td>0.272000</td>
<td>0.240000</td>
</tr>
</tbody>
</table>

TABLE 6.2.7: Comparison of the solutions by the least squares method with the exact solution
<table>
<thead>
<tr>
<th>( \omega )</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>Iteration</th>
<th>N-Condition number</th>
<th>M-Condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.8569999D-01</td>
<td>0.2272446D+01</td>
<td>178</td>
<td>0.9278665D+01</td>
<td>0.1900000D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.1028450D+00</td>
<td>0.1774040D+01</td>
<td>117</td>
<td>0.6252180D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.1225668D+00</td>
<td>0.1415726D+01</td>
<td>79</td>
<td>0.4411083D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.1430483D+00</td>
<td>0.1175724D+01</td>
<td>58</td>
<td>0.3341684D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.1598883D+00</td>
<td>0.1041385D+01</td>
<td>46</td>
<td>0.2788481D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.1666667D+00</td>
<td>0.1000000D+01</td>
<td>42</td>
<td>0.2617621D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.1649048D+00</td>
<td>0.1010096D+01</td>
<td>42</td>
<td>0.2659728D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.1598884D+00</td>
<td>0.1041385D+01</td>
<td>44</td>
<td>0.2788481D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.1523123D+00</td>
<td>0.1095982D+01</td>
<td>47</td>
<td>0.3011295D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.1430483D+00</td>
<td>0.1175724D+01</td>
<td>52</td>
<td>0.3341684D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.1329170D+00</td>
<td>0.1281885D+01</td>
<td>60</td>
<td>0.3799131D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.1225669D+00</td>
<td>0.1415726D+01</td>
<td>71</td>
<td>0.4411082D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.1028450D+00</td>
<td>0.1774040D+01</td>
<td>102</td>
<td>0.6252179D+01</td>
<td>0.1800000D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.8569999D-01</td>
<td>0.2272446D+01</td>
<td>153</td>
<td>0.9278663D+01</td>
<td>0.1900000D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.2.8:** Tabulation of eigenvalues, condition numbers for the coefficient matrix \( B_{ij} \) and the number of iterations required to achieve an accuracy of \( 5 \times 10^{-6} \) for the least squares method.
<table>
<thead>
<tr>
<th>n</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Over-Determined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P-condition Number</td>
<td>No. of iterations</td>
<td>K-condition Number</td>
<td>No. of iterations</td>
</tr>
<tr>
<td>2</td>
<td>( \omega )</td>
<td>8.550</td>
<td>52</td>
<td>5.828</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>110.428</td>
<td>639</td>
<td>88.647</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1923.347</td>
<td>&gt;</td>
<td>2360.557</td>
</tr>
<tr>
<td>2</td>
<td>( \omega )</td>
<td>2.666</td>
<td>17</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>16.034</td>
<td>99</td>
<td>1.1</td>
</tr>
<tr>
<td>4</td>
<td>1.3</td>
<td>157.170</td>
<td>952</td>
<td>1.3</td>
</tr>
</tbody>
</table>

TABLE 6.2.9: Comparisons of the number of iterations and condition numbers of the four methods for \( \omega = 0 \) and \( \omega = \omega_{opt} \) - Problem 1

> more than 10,000 iterations.
FIG. 6.2.1: P-CON Vs W - PROBLEM 1
FIG 6.2.2  ERROR DISTRIBUTION (POLYNOMIAL APP.)

PROBLEM ONE
From the comparisons made so far for this particular problem it is difficult to decide which is the 'best' method. It would appear in Figure 6.2.1 that the Rayleigh-Ritz and Over Determined Collocation methods give the best distribution of errors for the three parameter approximations, (i.e. \( n=3 \)). However, from the relative Euclidean error norm \( \| E \|_2 \) as defined in (4.2.22), where \( i=n=2,\ldots,7 \) is the number of parameters approximation, we see that an accurate answer up to six decimal places can be obtained when a four parameter approximation is used (see Table 6.2.11). This is perhaps due to the simple nature of the problem and the fact that the exact answer is a polynomial of order five, thus an approximation of order five would be expected to produce a very close approximation to the exact solution.

<table>
<thead>
<tr>
<th>Methods</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rayleigh Ritz Galerkin</td>
<td>0.99483</td>
<td>1.93714</td>
<td>2.62286</td>
<td>2.36571</td>
</tr>
<tr>
<td>Collocation</td>
<td>1.03000</td>
<td>1.95500</td>
<td>2.60500</td>
<td>2.33000</td>
</tr>
<tr>
<td>Over-determined Collocation</td>
<td>0.99755</td>
<td>1.93878</td>
<td>2.61212</td>
<td>2.36245</td>
</tr>
<tr>
<td>Least Squares</td>
<td>0.96000</td>
<td>1.92000</td>
<td>2.64000</td>
<td>2.40000</td>
</tr>
<tr>
<td>Exact Method</td>
<td>0.99840</td>
<td>1.94880</td>
<td>2.61120</td>
<td>2.36160</td>
</tr>
</tbody>
</table>

\[ \text{TABLE 6.2.10: Approximate solution} \times 10^2 \text{ to the boundary value problem 1 for } n=3 \]

We thus require another criteria of comparison where we can decide which is the 'best' method to choose from in solving this type of problem.

\(^{\dagger}\text{errors} = u_0(x) - u_3(x), \ 0<x<1.\)
<table>
<thead>
<tr>
<th>n</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Over-Determined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.79517</td>
<td>16.03964</td>
<td>2.55067</td>
<td>7.89130</td>
</tr>
<tr>
<td>3</td>
<td>0.24364</td>
<td>5.55451</td>
<td>0.27375</td>
<td>0.66308</td>
</tr>
<tr>
<td>4</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>5</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

**TABLE 6.2.11:** Error norm for different methods - Problem 1

<table>
<thead>
<tr>
<th>n</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Over-Determined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>110.4</td>
<td>88.65</td>
<td>40.63</td>
</tr>
<tr>
<td>6</td>
<td>.866x10^6</td>
<td>.287x10^7</td>
<td>.357x10^6</td>
<td>.166x10^6</td>
</tr>
<tr>
<td>10</td>
<td>.346x10^{12}</td>
<td>.112x10^{14}</td>
<td>.201x10^{12}</td>
<td>.620x10^{11}</td>
</tr>
</tbody>
</table>

**TABLE 6.2.12:** Comparisons of condition numbers of the coefficient matrices for the four different methods at \( \omega = 0 \) and \( \omega = \omega_{opt} \)
Table 6.2.9 indicates that, for this problem the preconditioning reduces the condition number of the coefficient matrix and the iteration number required at the optimum \( \omega \) is roughly of the same order of the square root of that at the \( \omega=0 \) (without preconditioning), which agrees with the theory in Chapter 5. The more important deduction from these tables is that of all the methods used to find the approximated solutions for this problem, the method of Least Squares leads to the algebraic solution (6.1.1) with the least condition numbers. The comparison of the \( P \)-condition numbers for the 3, 6 and 10 terms of approximations given in Table 6.2.12 further justifies this claim.

It is to be noted that although the Collocation method is simple to use, and in this particular problem the method yields quite remarkable accuracy in the results in view of the primitive nature of the method and the slight amount of computation involved, its condition number is a factor of two higher than that of the Least Squares method.

### A Comparison of Different Basis Functions in the Least Squares Method

So far we have only used the sequence of polynomials as the coordinate/basis functions \( \{ Q_n \} \). In order to seek for the 'best' sequence for the basis functions, two more sequences that are in accordance with the principle described in 5.8 were used and the condition numbers and the number of iterations required to achieve the same accuracy (i.e. \( 0.5 \times 10^{-6} \)) for the preconditioned Simultaneous Displacement methods were analysed and the results compared. The sequences were:

1. Sequence of transcendental function (sequence b),

\[
Q_k(x) = x(e^k - e^{-k}), \quad k=1,2,3,...
\]  \hspace{1cm} (6.2.12)

which satisfies the boundary condition (6.2.1a), i.e.,

\[
Q_k(0) = Q_k(1) = 0.
\]  \hspace{1cm} (6.2.12a)
2. Sequence of trigonometric functions (sequence c),

\[ Q_k = \sin k\pi x, \quad k=1, 2, 3, \ldots \]  \hspace{1cm} (6.2.13)

which also satisfies the boundary condition (6.2.1a).

**Sequence b Exponential Basis Functions**

In this case, again for \( k=3 \),

\[ Q_1 = x(e^x - e), \quad Q_2 = x(e^{2x} - e^2) \quad \text{and} \quad Q_3 = x(e^{3x} - e^3), \]  \hspace{1cm} (6.2.14)

Therefore,

\[ A_0 Q_1 = e^x (2x), \quad A_0 Q_2 = 2e^{2x} (2+2x) \]

and

\[ A_0 Q_3 = 3e^{3x} (2+3x). \]  \hspace{1cm} (6.2.15)

Thus, (5.7.22) becomes,

\[
\begin{bmatrix}
22.7644 & 120.371 & 484.484 \\
120.371 & 679.977 & 2860.80 \\
484.484 & 2860.80 & 12401.7 \\
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\end{bmatrix}
= 
\begin{bmatrix}
-1.59141 \\
-9.58358 \\
-41.9584 \\
\end{bmatrix}
\hspace{1cm} (6.2.16)
\]

giving,

\[ a_1 = 0.488566, \quad a_2 = -0.016229, \quad \text{and} \quad a_3 = -0.0015481. \]

The approximate solution is,

\[ u_3(x) = 0.048857x(e^x - e) - 0.016229x(e^{2x} - e^2) - 0.001548x(e^{3x} - e^3). \]

The relevant results are given in Table 6.2.13, 6.2.14 and 6.2.15.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( \text{Approximate Solution} \ u_n \times 10^2 )</th>
<th>( \text{Exact Solution} \ u_0 \times 10^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=2</td>
<td>n=3</td>
<td>n=4</td>
</tr>
<tr>
<td>0.2</td>
<td>1.00837</td>
<td>1.01701</td>
</tr>
<tr>
<td>0.4</td>
<td>2.00891</td>
<td>1.99342</td>
</tr>
<tr>
<td>0.6</td>
<td>2.67911</td>
<td>2.63395</td>
</tr>
<tr>
<td>0.8</td>
<td>2.39144</td>
<td>2.35931</td>
</tr>
</tbody>
</table>

**TABLE 6.2.13:** Comparison of the solutions to problem 1 by approximate Least Squares method with exponential basis functions for different values of \( n \).
<table>
<thead>
<tr>
<th>ω</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>P-condition number</th>
<th>Iteration</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.6932841D-03</td>
<td>0.2909978D+01</td>
<td>0.4197382D+04</td>
<td>&gt;</td>
<td>0.1399829D+04</td>
<td>0.2806308D+04</td>
</tr>
<tr>
<td>0.2</td>
<td>0.8517929D-03</td>
<td>0.2020848D+01</td>
<td>0.2372464D+04</td>
<td>&gt;</td>
<td>0.7919082D+04</td>
<td>0.2172214D+04</td>
</tr>
<tr>
<td>0.4</td>
<td>0.1057850D-02</td>
<td>0.1483765D+01</td>
<td>0.1402623D+04</td>
<td>9658</td>
<td>0.4689587D+03</td>
<td>0.1620782D+04</td>
</tr>
<tr>
<td>0.6</td>
<td>0.1322069D-02</td>
<td>0.1185073D+01</td>
<td>0.8963639D+03</td>
<td>6268</td>
<td>0.3002123D+03</td>
<td>0.1152014D+04</td>
</tr>
<tr>
<td>0.8</td>
<td>0.1646844D-02</td>
<td>0.1041592D+01</td>
<td>0.6324779D+03</td>
<td>4457</td>
<td>0.2119223D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.0</td>
<td>0.2008422D-02</td>
<td>0.1000000D+01</td>
<td>0.4979034D+03</td>
<td>3512</td>
<td>0.1666770D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.1</td>
<td>0.218245O4D-02</td>
<td>0.1010100D+01</td>
<td>0.4628388D+03</td>
<td>3259</td>
<td>0.1548399D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.2</td>
<td>0.2333660D-02</td>
<td>0.1041595D+01</td>
<td>0.4463352D+03</td>
<td>3134</td>
<td>0.1492294D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.3</td>
<td>0.2446114D-02</td>
<td>0.1098046D+01</td>
<td>0.4488940D+03</td>
<td>3137</td>
<td>0.1500018D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2505568D-02</td>
<td>0.1185327D+01</td>
<td>0.4730772D+03</td>
<td>3285</td>
<td>0.1580342D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.5</td>
<td>0.2503760D-02</td>
<td>0.1311565D+01</td>
<td>0.5238381D+03</td>
<td>3609</td>
<td>0.1749479D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.6</td>
<td>0.2441060D-02</td>
<td>0.1487090D+01</td>
<td>0.6091984D+03</td>
<td>4156</td>
<td>0.2034253D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>1.8</td>
<td>0.2173917D-02</td>
<td>0.2038200D+01</td>
<td>0.9375706D+03</td>
<td>6244</td>
<td>0.3103232D+03</td>
<td>0.1064679D+04</td>
</tr>
<tr>
<td>2.0</td>
<td>0.1818200D-02</td>
<td>0.2964749D+01</td>
<td>0.1630595D+04</td>
<td>&gt;</td>
<td>0.5443698D+03</td>
<td>0.1064679D+04</td>
</tr>
</tbody>
</table>

**TABLE 6.2.14**: Tabulation of eigenvalues condition numbers of \( B_w \) and the number of iterations for the Least Squares method with exponential basis functions - Problem 1

<table>
<thead>
<tr>
<th>n</th>
<th>ω</th>
<th>P-condition Number</th>
<th>Number of iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>0.6052534x10^2</td>
<td>410</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0.419738x10^4</td>
<td>&gt;</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
<td>0.1563546x10^2</td>
<td>111</td>
</tr>
<tr>
<td>3</td>
<td>1.2</td>
<td>0.446330x10^3</td>
<td>3134</td>
</tr>
</tbody>
</table>

**TABLE 6.2.15**: Least Squares method with exponential basis function - Problem 1
Sequence c Trigonometric Basis Functions

The functions in (6.2.13) are the eigenfunctions to the operator

$$AU = -\frac{\partial^2 U}{\partial x^2}, \quad U(0) = U(1) = 0.$$  \hspace{1cm} (6.2.17)

By limiting ourselves to \(k \leq 3\), we have,

$$Q_1 = \sin \pi x, \quad Q_2 = \sin 2\pi x \quad \text{and} \quad Q_3 = \sin 3\pi x.$$  \hspace{1cm} (6.2.18)

Therefore,

$$AQ_k = -(k\pi)^2 \sin k\pi x, \quad k = 1, 2, 3, \ldots.$$  \hspace{1cm} (6.2.19)

Remembering (5.7.22), i.e.,

$$a_{ij} = \int_0^1 (AQ_i, AQ_j)dx,$$

and that the \(AQ_k\) are orthogonal (see Appendix A), therefore the only contribution from the summation in (5.7.21) occurs when \(i=j\). Thus, the equation (5.7.21) can be written as,

$$\int (AQ_i, AQ_j)dx = \int f_1 AQ_j, \quad i = 1, 2, 3.$$  \hspace{1cm} (6.2.20)

which leads to the equation,

$$\begin{bmatrix} 48.7045 & 779.273 & 0 \\ 779.273 & 3945.07 & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1.23173 \\ -5.32826 \\ 8.78816 \end{bmatrix}.$$  \hspace{1cm} (6.2.21)

Clearly, the coefficient matrix of (6.2.21) is trivial to solve and would not require any iterative method. When we solve this system, we obtain,

$$a_1 = 0.025289, \quad a_2 = -0.0068375 \quad \text{and} \quad a_3 = 0.0022276.$$  

The approximate solution is then,

$$u_3(x) = 0.025289 \sin \pi x - 0.0068375 \sin 2\pi x + 0.0022276 \sin 3\pi x.$$  

A comparison of these solutions for various points \(x\) and for various terms of approximations is given in Table 6.2.16.

We give in Table 6.2.17 the approximate solutions for the Least Squares method using these different basis functions for \(n=3\) while the relative error distribution is given in Figure 6.2.3. The Euclidean error norm given in Table 6.2.18 indicates that a polynomial approximation gives the most accurate answer for this problem followed by an exponential approximation.
<table>
<thead>
<tr>
<th>x</th>
<th>Approximate Solution $u_n$ ($\times 10^2$)</th>
<th>Exact Solution $u_0$ ($\times 10^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n=2$</td>
<td>$n=3$</td>
</tr>
<tr>
<td>0.2</td>
<td>0.83622</td>
<td>1.04808</td>
</tr>
<tr>
<td>0.4</td>
<td>2.00332</td>
<td>1.87238</td>
</tr>
<tr>
<td>0.6</td>
<td>2.80711</td>
<td>2.67617</td>
</tr>
<tr>
<td>0.8</td>
<td>2.13679</td>
<td>2.34865</td>
</tr>
</tbody>
</table>

TABLE 6.2.16: Comparison of the solution to the problem by approximate Least Squares method with trigonometric basis functions for different values of $n$.

<table>
<thead>
<tr>
<th>Basis functions</th>
<th>x location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.2</td>
</tr>
<tr>
<td>Exact Solution</td>
<td>0.99840</td>
</tr>
<tr>
<td>Polynomial</td>
<td>0.96000</td>
</tr>
<tr>
<td>Exponential</td>
<td>1.01701</td>
</tr>
<tr>
<td>Trigonometric</td>
<td>1.04808</td>
</tr>
</tbody>
</table>

TABLE 6.2.17: Approximate solutions ($\times 10^2$) for the Least Squares method using different basis functions.

<table>
<thead>
<tr>
<th>n</th>
<th>Basis Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Polynomial</td>
</tr>
<tr>
<td>2</td>
<td>7.89130</td>
</tr>
<tr>
<td>3</td>
<td>0.66308</td>
</tr>
<tr>
<td>4</td>
<td>0.00000</td>
</tr>
<tr>
<td>5</td>
<td>0.00000</td>
</tr>
<tr>
<td>6</td>
<td>0.00000</td>
</tr>
<tr>
<td>7</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

TABLE 6.2.18: Euclidean norm for Least Squares method - Problem 1
FIG 6.2.3  ERROR DISTRIBUTION (LEAST SQUARES)
PROBLEM ONE
It is apparent from Table 6.2.19 that using transcendental functions as basis functions for the Least Squares approximation gives large P-condition numbers and can increase without bound as \( n \) increases, thus poor convergence rates will result in the iterative process used. Ideally, one would choose the orthogonal functions for the basis functions in the approximation of the Least Squares method to obtain the diagonal coefficient matrix in (5.7.22). For the method of Rayleigh-Ritz, Mikhlin (1967) proves that if any orthonormal system that is complete in \( H_A \) is taken for the basis functions, the eigenvalues of the Ritz method will be bounded from above and below by the constant \( c_2 \) and \( c_1 \); the P-condition number of the Ritz matrix is then bounded by the quantity \( c_2/c_1 \). The chief difficulty with the Least Squares method, or in fact in all the variational methods is in choosing orthogonal functions appropriate to the problems, especially when the non-linear equation is considered. The treatment of non-linear equations is more cumbersome with the variational methods than it is with the finite element/difference method. The reason is that the unknowns \( a_i \) in (6.1.2) have a global character whereas the unknowns in the finite element/difference formulation have a purely local character. We will describe the method of finite elements in Section 6.4.
6.3 PROBLEM 2

We now consider the two point boundary-value problem,

\[-(1+x) \frac{\partial^2 u}{\partial x^2} - \frac{\partial u}{\partial x} = x, \quad 0 < x < 1,\]  

(6.3.1)

with the boundary conditions,

\[u(0) = u(1) = 0.\]  

(6.3.1a)

This differential equation may be written in self-adjoint form as,

\[-\frac{3}{\partial x}(1+x) \frac{\partial u}{\partial x} = x.\]  

(6.3.2)

We begin with the sequence of polynomials as the coordinate function (6.2.2) and again we restrict ourselves to terms for \(k \leq 2\).

Rayleigh-Ritz

We have,

\[[Q_i, Q_j] = \int_0^1 P \frac{\partial Q_i}{\partial x} \frac{\partial Q_j}{\partial x} \, dx, \quad i, j = 0, 1, 2.\]  

(6.3.3)

where \(P = (1+x)\), and

\[(f, Q_i) = \int_0^1 x \frac{\partial Q_i}{\partial x} \, dx, \quad i = 0, 1, 2, \ldots\]  

(6.3.4)

Thus, in a similar manner to (6.2.5a), we obtain the Rayleigh-Ritz system,

\[\begin{bmatrix}
0.5000 & 0.2833 & 0.1833 \\
0.2833 & 0.2333 & 0.1810 \\
0.1833 & 0.1810 & 0.1571
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2
\end{bmatrix} =
\begin{bmatrix}
0.0833 \\
0.0500 \\
0.0333
\end{bmatrix} \]  

(6.3.5)

Solving (6.3.5) for \(a_0\), \(a_1\) and \(a_2\), the approximation is then given by,

\[u_3(x) = 0.140449(x(1-x)) + 0.058968x^2(1-x) - 0.01963x^3(1-x).\]  

(6.3.6)

A similar comparison between \(u_n(x)\), \(n = 3, 4\) and 5 and \(u_0(x)\) is given in Table 6.3.1. Table 6.3.2 gives the largest and smallest eigenvalues, P-condition number of the iteration required, N and M-condition numbers of the preconditioned Rayleigh-Ritz matrix for \(n = 3\). Fig. 6.3.1 shows the P-condition numbers plotted for various values of the preconditioning parameter \(\omega\) for different methods of approximation. In Fig. 6.3.2 we give a comparison of the relative errors in the four approximate solutions of this problem. Finally Table 6.3.9 compares the P-condition numbers and the number of iterations for values of the preconditioning parameter \(\omega = 0\) and \(\omega = \omega_{opt}\).
### TABLE 6.3.1: Rayleigh-Ritz Method - Problem 2

<table>
<thead>
<tr>
<th>$x$</th>
<th>Approximate Solution, $u_n \times 10^2$</th>
<th>Exact Solution $u_0 \times 10^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n=2$</td>
<td>$n=4$</td>
</tr>
<tr>
<td>0.2</td>
<td>2.42337</td>
<td>2.42368</td>
</tr>
<tr>
<td>0.4</td>
<td>3.86157</td>
<td>3.86452</td>
</tr>
<tr>
<td>0.6</td>
<td>4.05034</td>
<td>4.04841</td>
</tr>
<tr>
<td>0.8</td>
<td>2.80090</td>
<td>2.79976</td>
</tr>
</tbody>
</table>

### TABLE 6.3.2: Rayleigh-Ritz Method - Problem 2

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>P-condition number</th>
<th>Iterations</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.1759127D-01</td>
<td>0.2634800D+01</td>
<td>0.1492104D+03</td>
<td>897</td>
<td>0.5025921D+02</td>
<td>0.1039644D+03</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2150985D-01</td>
<td>0.1920573D+01</td>
<td>0.8928807D+02</td>
<td>546</td>
<td>0.3044812D+02</td>
<td>0.7834370D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2648095D-01</td>
<td>0.1460461D+01</td>
<td>0.5515140D+02</td>
<td>347</td>
<td>0.1919611D+02</td>
<td>0.5678343D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.3260065D-01</td>
<td>0.1182646D+01</td>
<td>0.3627675D+02</td>
<td>234</td>
<td>0.1290712D+02</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.3960539D-01</td>
<td>0.1041554D+01</td>
<td>0.2629829D+02</td>
<td>172</td>
<td>0.9449727D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.4643428D-01</td>
<td>0.1000000D+02</td>
<td>0.2153582D+02</td>
<td>142</td>
<td>0.7699152D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.4916105D-01</td>
<td>0.1010099D+01</td>
<td>0.2054674D+02</td>
<td>135</td>
<td>0.7305088D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.5102695D-01</td>
<td>0.1041567D+01</td>
<td>0.2041209D+02</td>
<td>134</td>
<td>0.7211733D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.5179612D-01</td>
<td>0.1097758D+01</td>
<td>0.2119383D+02</td>
<td>138</td>
<td>0.7438684D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.5137617D-01</td>
<td>0.1183875D+01</td>
<td>0.2304326D+02</td>
<td>149</td>
<td>0.8034794D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.4984733D-01</td>
<td>0.1306609D+01</td>
<td>0.2621223D+02</td>
<td>168</td>
<td>0.9083487D+01</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.4742832D-01</td>
<td>0.1473928D+01</td>
<td>0.3107697D+02</td>
<td>196</td>
<td>1.071129D+02</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.4105087D-01</td>
<td>0.1980371D+01</td>
<td>0.4824187D+02</td>
<td>296</td>
<td>1.649875D+02</td>
<td>0.5666854D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.3422137D-01</td>
<td>0.2793024D+01</td>
<td>0.8161637D+02</td>
<td>489</td>
<td>0.2779118D+02</td>
<td>0.5666854D+02</td>
</tr>
</tbody>
</table>
The Collocation Method

From (5.7.11), we have,

$$A_{Q_k}(P_j) = -(1+x) [k(k+1)x^{(k-1)}-(k+1)(k+2)x^k]-(k+1)x^k-(k+2)x^{(k+1)},$$

for \( k=0,1,2 \), \( (6.3.7) \)

The collocation points are chosen to be \( x_j = \frac{j}{(k+2)}, j=1,2 \) and 3, giving

\[
\begin{bmatrix}
2.0000 & -0.9375 & -1.0625 \\
3.0000 & 1.250 & 2.500 \\
4.0000 & 4.563 & 3.9375 \\
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\end{bmatrix} = 
\begin{bmatrix}
0.2500 \\
0.5000 \\
0.7500 \\
\end{bmatrix}
\quad (6.3.8)
\]

from which we obtain,

\( a_0 = 0.141509, \quad a_1 = 0.056604 \quad \text{and} \quad a_2 = -0.018868 \),

thus giving the final approximate solution as,

$$u_3(x) = 0.141509x(1-x) + 0.056604x^2(1-x) - 0.018868x^3(1-x), \quad (6.3.9)$$

The results for this method are given in Tables 6.3.3 and 6.3.4 and Figure 6.3.1.

<table>
<thead>
<tr>
<th>x</th>
<th>Approximate solution, ( u_n \times 10^2 )</th>
<th>Exact solution ( u_0 \times 10^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n=3 )</td>
<td>( n=4 )</td>
</tr>
<tr>
<td>0.2</td>
<td>2.43321</td>
<td>2.42666</td>
</tr>
<tr>
<td>0.4</td>
<td>3.86717</td>
<td>3.86641</td>
</tr>
<tr>
<td>0.6</td>
<td>4.04841</td>
<td>4.05004</td>
</tr>
<tr>
<td>0.8</td>
<td>2.79547</td>
<td>2.80178</td>
</tr>
</tbody>
</table>

**TABLE 6.3.3:** Collocation Method
The Over-Determined Collocation Method

The collocation points (of 6.3.7) are chosen to be,

\[ x = \frac{j}{(1+m)} \quad , \quad m=1,2,\ldots,6, \]

which in turn leads to the over-determined system,

\[
\begin{bmatrix}
1.5714 & -1.5306 & -0.7493 \\
1.2143 & -0.6939 & -1.0962 \\
2.7143 & 0.5102 & -0.7609 \\
3.2857 & 2.0816 & 0.5364 \\
3.8571 & 4.0204 & 3.0758 \\
4.4286 & 6.3265 & 7.1371
\end{bmatrix}
\begin{bmatrix}
a_0 \\ a_1 \\ a_2
\end{bmatrix}
= \begin{bmatrix}
0.1429 \\
0.2857 \\
0.4286 \\
0.5714 \\
0.7143 \\
0.8571
\end{bmatrix}
\]

(6.3.10)
The method of least squares when used to solve this set yield the following set of equations,

\[
\begin{bmatrix}
59.7143 & 47.8571 & 39.6414 \\
47.8571 & 63.6064 & 60.1545 \\
39.6414 & 60.1545 & 63.0276
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2
\end{bmatrix}
= \begin{bmatrix}
10.4286 \\
9.2857 \\
7.8746
\end{bmatrix}
\]

(6.3.11)

giving,

\[a_0 = 0.140686, \quad a_1 = 0.058116 \quad \text{and} \quad a_3 = -0.019013.\]

Thus, the over-determined collocation approximation is,

\[u_3(x) = 0.140686x(1-x) + 0.058116x^2(1-x) - 0.019013x^3(1-x).\]

(6.3.12)

The relevant results are given in Tables 6.3.5 and 6.3.6 and Figure 6.3.1.

<table>
<thead>
<tr>
<th>(x)</th>
<th>(\text{Approximate Solution (x10^2)})</th>
<th>(\text{Exact Solution u_o (x10^2)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n=2)</td>
<td>(n=4)</td>
<td>(n=5)</td>
</tr>
<tr>
<td>0.2</td>
<td>2.42478</td>
<td>2.42425</td>
</tr>
<tr>
<td>0.4</td>
<td>3.86138</td>
<td>3.86492</td>
</tr>
<tr>
<td>0.6</td>
<td>4.04908</td>
<td>4.04890</td>
</tr>
<tr>
<td>0.8</td>
<td>2.80018</td>
<td>2.80023</td>
</tr>
</tbody>
</table>

TABLE 6.3.5: Over-determined collocation - Problem 2
<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>P-condition number</th>
<th>Iteration</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.3427552D-01</td>
<td>0.2508656D+01</td>
<td>0.7319090D+02</td>
<td>532</td>
<td>0.5074647D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4168372D-01</td>
<td>0.1875109D+01</td>
<td>0.4498222D+02</td>
<td>331</td>
<td>0.3679678D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.5073996D-01</td>
<td>0.1447928D+01</td>
<td>0.2853624D+02</td>
<td>214</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6118173D-01</td>
<td>0.1181012D+01</td>
<td>0.1930334D+02</td>
<td>148</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.7175930D-01</td>
<td>0.1041522D+01</td>
<td>0.1451411D+02</td>
<td>113</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.7973609D-01</td>
<td>0.1000000D+01</td>
<td>0.1254137D+02</td>
<td>98</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.8166725D-01</td>
<td>0.1010000D+01</td>
<td>0.1236847D+02</td>
<td>96</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.8183356D-01</td>
<td>0.1041537D+01</td>
<td>0.1272750D+02</td>
<td>98</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.8023464D-01</td>
<td>0.1097446D+01</td>
<td>0.1367995D+02</td>
<td>105</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.7711801D-01</td>
<td>0.1182302D+01</td>
<td>0.1533109D+02</td>
<td>116</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.7288496D-01</td>
<td>0.1301359D+01</td>
<td>0.1785497D+02</td>
<td>134</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.6797103D-01</td>
<td>0.1460418D+01</td>
<td>0.2148589D+02</td>
<td>159</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.7</td>
<td>0.6275845D-01</td>
<td>0.1665950D+01</td>
<td>0.2654543D+02</td>
<td>195</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.5753757D-01</td>
<td>0.1925324D+01</td>
<td>0.3346203D+02</td>
<td>242</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.4777819D-01</td>
<td>0.2640725D+01</td>
<td>0.5527051D+02</td>
<td>393</td>
<td>0.3654185D+02</td>
<td>0.3654185D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.3.6:** Over-Determined Collocation - Problem 2

The Least Squares Method

From (6.3.1), the least squares system (5.7.21) becomes,

\[
\begin{bmatrix}
10.3333 & 9.6667 & 9.8000 \\
9.6667 & 14.5333 & 15.8000 \\
9.8000 & 15.8000 & 18.9714
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2
\end{bmatrix}
= 
\begin{bmatrix}
1.8333 \\
1.9167 \\
1.9500
\end{bmatrix}
\]

(6.3.13)

giving as a solution,

\[a_0 = 0.139549, \quad a_1 = 0.060117 \quad \text{and} \quad a_2 = -0.019367\]

Thus, the least squares approximate solution is given by,

\[u_3(x) = 0.139549x(1-x) + 0.060117x^2(1-x) - 0.019367x^3(1-x).\]  

(6.3.14)

The results for this method are given in Tables 6.3.7 and 6.3.8 and Figure 6.3.1.
<table>
<thead>
<tr>
<th>$x$</th>
<th>Approximate solution, $u_n \times 10^2$</th>
<th>Exact solution $u_0 \times 10^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n=3$</td>
<td>$n=4$</td>
</tr>
<tr>
<td>0.2</td>
<td>2.41276</td>
<td>2.42246</td>
</tr>
<tr>
<td>0.4</td>
<td>3.85192</td>
<td>3.86456</td>
</tr>
<tr>
<td>0.6</td>
<td>4.04752</td>
<td>4.04903</td>
</tr>
<tr>
<td>0.8</td>
<td>2.80395</td>
<td>2.79961</td>
</tr>
</tbody>
</table>

**TABLE 6.3.7:** Least Squares - Problem 2

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>P-condition number</th>
<th>Iteration</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.3814416D-01</td>
<td>0.2631438D+01</td>
<td>0.6899665D+02</td>
<td>470</td>
<td>0.2333488D+02</td>
<td>0.4614106D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4624318D-01</td>
<td>0.1922661D+01</td>
<td>0.4148747D+02</td>
<td>288</td>
<td>0.1420272D+02</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.5632190D-01</td>
<td>0.1460575D+01</td>
<td>0.2593262D+02</td>
<td>184</td>
<td>0.9067421D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6771110D-01</td>
<td>0.1182547D+01</td>
<td>0.174646D+02</td>
<td>127</td>
<td>0.6269284D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.7891525D-01</td>
<td>0.1041549D+01</td>
<td>0.1319832D+02</td>
<td>97</td>
<td>0.4832718D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.8653594D-01</td>
<td>0.1000000D+01</td>
<td>0.1155589D+02</td>
<td>85</td>
<td>0.4257893D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.8776062D-01</td>
<td>0.1010099D+01</td>
<td>0.1150971D+02</td>
<td>85</td>
<td>0.4231314D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.8693429D-01</td>
<td>0.1041554D+01</td>
<td>0.1198094D+02</td>
<td>88</td>
<td>0.4378835D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>0.8421818D-01</td>
<td>0.1097613D+01</td>
<td>0.1303297D+02</td>
<td>95</td>
<td>0.4720335D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>0.8003738D-01</td>
<td>0.1183059D+01</td>
<td>0.1478146D+02</td>
<td>107</td>
<td>0.5293556D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.7491698D-01</td>
<td>0.1303717D+01</td>
<td>0.1740216D+02</td>
<td>124</td>
<td>0.6157558D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.6922978D-01</td>
<td>0.1466083D+01</td>
<td>0.2114346D+02</td>
<td>149</td>
<td>0.7397254D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.7</td>
<td>0.6367671D-01</td>
<td>0.1677499D+01</td>
<td>0.2634400D+02</td>
<td>184</td>
<td>0.9129012D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.5817777D-01</td>
<td>0.1946357D+01</td>
<td>0.334553D+02</td>
<td>230</td>
<td>0.1150762D+01</td>
<td>0.3417188D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.4819755D-01</td>
<td>0.2696626D+01</td>
<td>0.5594944D+02</td>
<td>377</td>
<td>0.1907032D+02</td>
<td>0.3417188D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.3.8:** Least Squares Method - Problem 2
TABLE 6.3.9: Comparisons of the number of iterations and condition numbers of the four methods for \( w=0 \) and \( w=w_{\text{opt}} \) - Problem 2

> more than 10,000 iterations.
### TABLE 6.3.10: Comparison of methods for the approximate solution \( (x_{lo}) \) to the boundary value Problem 2 for \( n=3 \)

<table>
<thead>
<tr>
<th>Method</th>
<th>( 0.2 )</th>
<th>( 0.4 )</th>
<th>( 0.6 )</th>
<th>( 0.8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rayleigh-Ritz Galerkin</td>
<td>2.42337</td>
<td>3.86157</td>
<td>4.05034</td>
<td>2.80090</td>
</tr>
<tr>
<td>Collocation</td>
<td>2.43321</td>
<td>3.86717</td>
<td>4.04841</td>
<td>2.79547</td>
</tr>
<tr>
<td>Over-Determined Collocation</td>
<td>2.42478</td>
<td>3.86138</td>
<td>4.04908</td>
<td>2.80018</td>
</tr>
<tr>
<td>Least Squares</td>
<td>2.41276</td>
<td>3.85192</td>
<td>4.04752</td>
<td>2.80395</td>
</tr>
<tr>
<td>Exact Methods</td>
<td>2.42414</td>
<td>3.86433</td>
<td>4.04820</td>
<td>2.80007</td>
</tr>
</tbody>
</table>

### TABLE 6.3.11: Euclidean norm for different methods \( (\times 10^6) \)

<table>
<thead>
<tr>
<th>( n )</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Over-Determined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.25586</td>
<td>0.62478</td>
<td>0.31392</td>
<td>0.48207</td>
</tr>
<tr>
<td>3</td>
<td>0.02908</td>
<td>0.09755</td>
<td>0.03704</td>
<td>0.08271</td>
</tr>
<tr>
<td>4</td>
<td>0.00312</td>
<td>0.02863</td>
<td>0.00668</td>
<td>0.01173</td>
</tr>
<tr>
<td>5</td>
<td>0.00043</td>
<td>0.00467</td>
<td>0.00087</td>
<td>0.00156</td>
</tr>
<tr>
<td>6</td>
<td>0.00008</td>
<td>0.00141</td>
<td>0.00024</td>
<td>0.00020</td>
</tr>
<tr>
<td>7</td>
<td>0.00004</td>
<td>0.00023</td>
<td>0.00005</td>
<td>0.00004</td>
</tr>
</tbody>
</table>

### TABLE 6.3.12: Comparison of condition numbers for Problem 2

<table>
<thead>
<tr>
<th>( n )</th>
<th>( w )</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Over-Determined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.50</td>
<td>1.4921×10^5</td>
<td>1.3596×10^2</td>
<td>n 7.3191×10^5</td>
<td>6.8987×10^5</td>
</tr>
<tr>
<td>6</td>
<td>6.00</td>
<td>1.2323×10^6</td>
<td>2.8662×10^6</td>
<td>12 3.5721×10^5</td>
<td>3.9073×10^5</td>
</tr>
<tr>
<td>10</td>
<td>10.00</td>
<td>5.0057×10^11</td>
<td>-</td>
<td>20 3.9481×10^11</td>
<td>1.4125×10^11</td>
</tr>
<tr>
<td>3</td>
<td>3.50</td>
<td>1.2041×10^4</td>
<td>1.1020963×10</td>
<td>1.1 1.2368×10^5</td>
<td>1.1 1.1510×10^5</td>
</tr>
<tr>
<td>6</td>
<td>6.00</td>
<td>4.8487×10^4</td>
<td>1.411947×10^5</td>
<td>1.4 1.5588×10^4</td>
<td>1.4 1.6138×10^4</td>
</tr>
<tr>
<td>10</td>
<td>10.00</td>
<td>8.8870×10^9</td>
<td>-</td>
<td>1.5 6.8780×10^9</td>
<td>1.5 2.4657×10^9</td>
</tr>
</tbody>
</table>

*Note: The tables above compare the performance of different approximation methods and their associated error norms for a specific boundary value problem.*
FIG. 6.3.1: P-CON VS \( W \) _ PROBLEM 2
FIG 6.3.2  ERROR DISTRIBUTION (POLYNOMIAL APP)
PROBLEM TWO
It is apparent from the results that for this problem, an accurate answer was not achieved even when a five parameter approximation is used. Thus, if it is desired to construct a more exact approximation, it is necessary to use higher-order systems (6.1.1), and hence the condition numbers become larger. (Note that for the same number of parameter approximations, i.e. n=3, the conditioned numbers of all the coefficient matrices for this problem are larger than that of Problem 1). Consequently, the errors accumulated during the computation of the matrices of the system (6.1.1) as well as in solving this system can reach appreciable proportions, hence the rate of convergence will be reduced. In order to accelerate the rate of convergence, we used the preconditioned technique described earlier. Again, we wish to determine the method that gives the lowest P- or K-condition numbers.

Although for n=2, the over-determined method leads to the algebraic solution (6.1.1) with the least P-condition numbers, but as n increases, the P-condition numbers become larger than that of the Least Squares method (see Table 6.3.12). We therefore conclude that for this problem, the least squares method has a faster rate of convergence. Analogous to the previous section, we now attempt to determine the 'best' sequence of basis functions in the sense that the coefficient matrix A of the system (6.1.1) has the least condition number.

A Comparison of Different Basis Functions in the Least Squares Method

Sequence b Exponential Basis Functions

From (6.2.14), we have

$$A_{0}k = -(1+x)ke^{kx}(2+kx) - e^{kx}(1+kx) - e^{k}, \quad k=1,2,3,$$

(6.3.15)

Therefore, (5.7.22) becomes,
\[
\begin{bmatrix}
79.424 & 451.093 & 1899.37 \\
451.093 & 2653.09 & 11448.8 \\
1899.37 & 11448.8 & 5029.66
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3
\end{bmatrix}
= 
\begin{bmatrix}
-5.0774 \\
-27.9590 \\
-115.045
\end{bmatrix}
\]

(6.3.16)

giving,
\[a_1 = -0.118643, \quad a_2 = 0.096208 \quad \text{and} \quad a_3 = 0.30846 \times 10^{-5}.
\]

Thus, the final approximate solution is given by,
\[u_3(x) = -0.11864x(e^{-x} - e) + 0.096208x(e^{2x} - e^2) + 0.000003x(e^{3x} - e^3).
\]

Further results are given in Table 6.3.13 and 6.3.14 and Figure 6.3.3.

<table>
<thead>
<tr>
<th>x</th>
<th>Approximate Solution (u_n(x) \times 10^2)</th>
<th>Exact Solution (u_0(x) \times 10^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=3</td>
<td>n=4</td>
</tr>
<tr>
<td>0.2</td>
<td>2.41604</td>
<td>2.40915</td>
</tr>
<tr>
<td>0.4</td>
<td>3.83128</td>
<td>3.84771</td>
</tr>
<tr>
<td>0.6</td>
<td>4.02803</td>
<td>4.04726</td>
</tr>
<tr>
<td>0.8</td>
<td>2.79967</td>
<td>2.79923</td>
</tr>
</tbody>
</table>

TABLE 6.3.13: Exponential Basis Function - Problem 2
<table>
<thead>
<tr>
<th>( \omega )</th>
<th>Smallest eigenvalue</th>
<th>Largest eigenvalue</th>
<th>P-condition number</th>
<th>Iteration</th>
<th>N-condition number</th>
<th>M-condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.3428006D-03</td>
<td>0.2949491D+01</td>
<td>0.8604101D+04</td>
<td>&gt;</td>
<td>0.2868515D+04</td>
<td>0.5732583D+04</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4212839D-03</td>
<td>0.2033629D+01</td>
<td>0.4827218D+04</td>
<td>&gt;</td>
<td>0.1609795D+04</td>
<td>0.4470109D+04</td>
</tr>
<tr>
<td>0.4</td>
<td>0.5236139D-03</td>
<td>0.1486402D+01</td>
<td>0.2837836D+04</td>
<td>&gt;</td>
<td>0.9471850D+03</td>
<td>0.3367210D+04</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6555291D-03</td>
<td>0.1185314D+01</td>
<td>0.1808186D+04</td>
<td>&gt;</td>
<td>0.6036705D+03</td>
<td>0.2423886D+04</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8191502D-03</td>
<td>0.1041596D+01</td>
<td>0.1271557D+04</td>
<td>9277</td>
<td>0.4245826D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.0</td>
<td>0.1004263D-02</td>
<td>0.1000000D+01</td>
<td>0.9575511D+03</td>
<td>7290</td>
<td>0.3324075D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.1</td>
<td>0.1095098D-02</td>
<td>0.1010100D+01</td>
<td>0.9223834D+03</td>
<td>6749</td>
<td>0.3078659D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.2</td>
<td>0.1175738D-02</td>
<td>0.1041597D+01</td>
<td>0.8859095D+03</td>
<td>6469</td>
<td>0.2956560D+02</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.3</td>
<td>0.1237819D-02</td>
<td>0.1098071D+01</td>
<td>0.8871015D+03</td>
<td>6455</td>
<td>0.2960343D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.4</td>
<td>0.1273578D-02</td>
<td>0.1185453D+01</td>
<td>0.9308049D+03</td>
<td>6738</td>
<td>0.3106162D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.5</td>
<td>0.1278008D-02</td>
<td>0.1311996D+01</td>
<td>0.1026594D+04</td>
<td>7379</td>
<td>0.3425945D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.6</td>
<td>0.1250494D-02</td>
<td>0.1488246D+01</td>
<td>0.1190126D+04</td>
<td>8479</td>
<td>0.3971930D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>1.8</td>
<td>0.118797D-02</td>
<td>0.2043441D+01</td>
<td>0.1826462D+04</td>
<td>&gt;</td>
<td>0.6096305D+03</td>
<td>0.2030710D+04</td>
</tr>
<tr>
<td>2.0</td>
<td>0.936534D-03</td>
<td>0.2980900D+01</td>
<td>0.3181821D+04</td>
<td>&gt;</td>
<td>0.1062037D+04</td>
<td>0.2030710D+04</td>
</tr>
</tbody>
</table>

**TABLE 6.3.14:** Exponential Basis Function - Problem 2

**Sequence c: Trigonometric Basis Function**

From (6.2.18), we have,

\[
A_{Q_k} = (1+x)(k\pi)^2 \sin k\pi x - \sin k\pi x, \; k=1,2,3.
\]  

(6.3.17)

Therefore, the Least Squares system is given by,

\[
10^3 \begin{bmatrix} 0.1211 & -0.1645 & 0.0463 \\ -0.1645 & 1.8479 & -1.2009 \\ 0.0463 & -1.2009 & 9.2718 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 5.647 \\ -12.566 \\ 18.637 \end{bmatrix}
\]  

(6.3.17a)

\[a_3(x) = 0.04343 \sin \pi x - 0.001931 \sin 2\pi x + 0.001543 \sin 3\pi x.
\]  

(6.3.18)

The relevant results for this approximation are given in Tables 6.3.15, 6.3.16, 6.3.17 and Fig. 6.3.3.
TABLE 6.3.15: Comparisons of approximate solutions with the exact solutions for the Least Squares method using trigonometric basis functions

<table>
<thead>
<tr>
<th>x location</th>
<th>Approximate solutions, $u_n^a$ ($\times 10^2$)</th>
<th>Exact solution, $u_n^e$ ($\times 10^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>2.51611, 2.41847, 2.45907</td>
<td>2.42414</td>
</tr>
<tr>
<td>0.4</td>
<td>3.92661, 3.90899, 3.91840</td>
<td>3.86433</td>
</tr>
<tr>
<td>0.6</td>
<td>4.15362, 4.07449, 4.09530</td>
<td>4.04820</td>
</tr>
<tr>
<td>0.8</td>
<td>2.88342, 2.86345, 2.85908</td>
<td>2.80007</td>
</tr>
</tbody>
</table>

TABLE 6.3.16: Comparison of condition numbers of Least Squares method with trigonometric basis functions for various values of $\omega$.

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>Smallest eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No. of Iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>5.679692D+00</td>
<td>1.1474980D+01</td>
<td>0.2596936D+01</td>
<td>18</td>
<td>0.1328330D+01</td>
<td>0.37346190D+01</td>
</tr>
<tr>
<td>0.2</td>
<td>6.197884D+00</td>
<td>1.1342785D+01</td>
<td>0.2166521D+01</td>
<td>15</td>
<td>0.1210799D+01</td>
<td>0.3626301D+01</td>
</tr>
<tr>
<td>0.4</td>
<td>6.774090D+00</td>
<td>1.225521D+01</td>
<td>0.1809131D+01</td>
<td>13</td>
<td>0.1122216D+01</td>
<td>0.3540133D+01</td>
</tr>
<tr>
<td>0.6</td>
<td>7.410073D+00</td>
<td>1.122653D+01</td>
<td>0.1515036D+01</td>
<td>10</td>
<td>0.1059751D+01</td>
<td>0.3476115D+01</td>
</tr>
<tr>
<td>0.8</td>
<td>8.092048D+00</td>
<td>1.038099D+01</td>
<td>0.1282863D+01</td>
<td>8</td>
<td>0.1021527D+01</td>
<td>0.3434245D+01</td>
</tr>
<tr>
<td>1.0</td>
<td>8.69769D+00</td>
<td>1.000000D+01</td>
<td>0.1149731D+01</td>
<td>7</td>
<td>0.1065560D+01</td>
<td>0.3414526D+01</td>
</tr>
<tr>
<td>1.1</td>
<td>8.705101D+00</td>
<td>1.010026D+01</td>
<td>0.1160269D+01</td>
<td>7</td>
<td>0.1007726D+01</td>
<td>0.3416956D+01</td>
</tr>
<tr>
<td>1.2</td>
<td>8.436775D+00</td>
<td>1.038785D+01</td>
<td>0.1231404D+01</td>
<td>8</td>
<td>0.1014729D+01</td>
<td>0.3416956D+01</td>
</tr>
<tr>
<td>1.3</td>
<td>8.120819D+00</td>
<td>1.080702D+01</td>
<td>0.1330780D+01</td>
<td>9</td>
<td>0.1027691D+01</td>
<td>0.3426477D+01</td>
</tr>
<tr>
<td>1.4</td>
<td>8.806657D+00</td>
<td>1.130716D+01</td>
<td>0.1484000D+01</td>
<td>10</td>
<td>0.1046792D+01</td>
<td>0.3441535D+01</td>
</tr>
<tr>
<td>1.5</td>
<td>7.802200D+00</td>
<td>1.186519D+01</td>
<td>0.1581561D+01</td>
<td>11</td>
<td>0.1072271D+01</td>
<td>0.3462131D+01</td>
</tr>
<tr>
<td>1.6</td>
<td>7.209951D+00</td>
<td>1.247267D+01</td>
<td>0.1729924D+01</td>
<td>12</td>
<td>0.1104432D+01</td>
<td>0.3488264D+01</td>
</tr>
<tr>
<td>1.8</td>
<td>6.663944D+00</td>
<td>1.382879D+01</td>
<td>0.2075166D+01</td>
<td>15</td>
<td>0.1190330D+01</td>
<td>0.3557143D+01</td>
</tr>
<tr>
<td>2.0</td>
<td>6.167614D+00</td>
<td>1.538030D+01</td>
<td>0.2493719D+01</td>
<td>17</td>
<td>0.1308232D+01</td>
<td>0.3648171D+01</td>
</tr>
<tr>
<td>Polynomial approximation</td>
<td>Exponential approximation</td>
<td>Trigonometric approximation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------</td>
<td>---------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-condition number</td>
<td>No.of iters.</td>
<td>K-condition number</td>
<td>No.of iters.</td>
<td>P-condition number</td>
<td>No.of iters</td>
<td></td>
</tr>
<tr>
<td>$\omega$</td>
<td>$\omega$</td>
<td>$\omega$</td>
<td>$\omega$</td>
<td>$\omega$</td>
<td>$\omega$</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>8.470</td>
<td>60</td>
<td>114.494</td>
<td>800</td>
<td>2.0676</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>68.987</td>
<td>470</td>
<td>8604.101</td>
<td>$&gt;$</td>
<td>2.597</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>994.376</td>
<td>6595</td>
<td>$0.6767 \times 10^6$</td>
<td>$&gt;$</td>
<td>3.005</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>2.700</td>
<td>20</td>
<td>29.711</td>
<td>218</td>
<td>1.0</td>
<td>1.138</td>
</tr>
<tr>
<td>3</td>
<td>11.510</td>
<td>85</td>
<td>887.102</td>
<td>6455</td>
<td>1.0</td>
<td>1.150</td>
</tr>
<tr>
<td>4</td>
<td>91.012</td>
<td>663</td>
<td>$0.4066 \times 10^5$</td>
<td>$&gt;$</td>
<td>1.1</td>
<td>1.118</td>
</tr>
</tbody>
</table>

**TABLE 6.3.17**: Comparisons of P-condition numbers and the number of iterations for the Least Squares method using three different basis functions

> more than 10,000 iterations
FIG 6.3.3  ERROR DISTRIBUTION, LEAST SQUARES  Problem Two
<table>
<thead>
<tr>
<th>Basis Functions</th>
<th>Polynomial</th>
<th>Trigonometric</th>
<th>Exponential</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.48207</td>
<td>1.87680</td>
<td>0.16080</td>
</tr>
<tr>
<td>3</td>
<td>0.08271</td>
<td>0.82204</td>
<td>0.16096</td>
</tr>
<tr>
<td>4</td>
<td>0.01173</td>
<td>0.49089</td>
<td>0.10399</td>
</tr>
<tr>
<td>5</td>
<td>0.00156</td>
<td>0.43006</td>
<td>0.04439</td>
</tr>
<tr>
<td>6</td>
<td>0.00020</td>
<td>0.25880</td>
<td>0.01490</td>
</tr>
<tr>
<td>7</td>
<td>0.00004</td>
<td>0.30315</td>
<td>0.00440</td>
</tr>
</tbody>
</table>

**Table 6.3.18:** Euclidean norms for the Least Squares method – Problem 2

Although, the trigonometric basis function in this case is not the orthogonal function for this problem (hence, the coefficient matrix A is not a diagonal matrix), it is however the eigenfunction of the operator A. Consequently the P-condition numbers of A using this sequence as a basis function, are much smaller (by a factor 10 for n=3) than using the sequence of polynomials or even more so when the sequence of transcendental functions is used (a factor of $10^3$ for n=3). However, Table 6.3.18 indicates that the Euclidean Norm for this approximation for n=7 is of a factor $10^2$ and $10^3$ larger than the exponential approximation and the polynomial approximation respectively. Therefore if it is desired to find only a crude approximation, then the Least Squares method with the trigonometric approximation function is the best method to use since it possesses the fastest rate of convergence. However, if accuracy is of a paramount importance, the polynomial basis function should be used (see also Table 6.3.17).
6.4 PROBLEM 3

We again consider the problem of asymmetric viscous flow in a channel, i.e. Example 5.2.1.

Comparisons in terms of accuracy and error distribution have already been given in Chapter 5. We now proceed with the comparison of the condition numbers of the coefficient matrix $A$ of the system (6.1.1) with and without preconditioning using the four variational methods as in the previous section.

The performance of the preconditioned methods with this problem are shown in the accompanying tables and diagrams in Tables 6.4.1 to 6.4.4, where the maximum and minimum eigenvalues and condition numbers for the coefficient matrix $B_\omega$ versus the preconditioning parameter $\omega$ for the four different variational methods were tabulated. Clearly, a minimum value of the condition number is achieved and for this value of $\omega$ the number of iterations required to achieve an accuracy of $0.5 \times 10^{-6}$ for the Simultaneous Displacement method is minimum. It is also apparent from these tables that the method of Least Squares gives the least condition numbers.

Since the approximate solutions for this problem is symmetric about the $y$ axis, we give only the results for $y$ negative. These solutions are given in Table 6.4.5a. The tabulated results together with Table 6.4.5b indicate that the solutions for $n=3$ agree more closely with the exact solution than that for $n=6$ for the collocation and over-determined collocation methods, whereas for the methods of Rayleigh-Ritz and Least Squares the accuracy of the solution increases with $n$. Figure 6.4.2 compares the errors in the four approximate solutions of problem 3.

All the above results were obtained when the approximation $u_n(x,y)$, given by,

$$u_n(x,y) = (1-e^{-\pi/2}) \cos \frac{\pi}{2} y(1-x) + x(1-x)(1-y^2) a_j x^j y^j$$

(6.4.1)
<table>
<thead>
<tr>
<th>ω</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No.of iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.1788535D+00</td>
<td>0.2151698D+01</td>
<td>0.1203050D+02</td>
<td>82</td>
<td>0.4374787D+01</td>
<td>0.9824525D+01</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2103374D+00</td>
<td>0.1717900D+01</td>
<td>0.8167350D+01</td>
<td>56</td>
<td>0.3114046D+01</td>
<td>0.8923420D+01</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2448915D+00</td>
<td>0.1395452D+01</td>
<td>0.5698245D+01</td>
<td>40</td>
<td>0.2328598D+01</td>
<td>0.8997051D+01</td>
</tr>
<tr>
<td>0.6</td>
<td>0.2782897D+00</td>
<td>0.1171549D+01</td>
<td>0.4209820D+01</td>
<td>30</td>
<td>0.1874353D+01</td>
<td>0.9119770D+01</td>
</tr>
<tr>
<td>0.8</td>
<td>0.3013493D+00</td>
<td>0.1041224D+01</td>
<td>0.3455205D+01</td>
<td>25</td>
<td>0.1660342D+01</td>
<td>0.9291576D+01</td>
</tr>
<tr>
<td>0.9</td>
<td>0.3051839D+00</td>
<td>0.1010093D+01</td>
<td>0.3309784D+01</td>
<td>24</td>
<td>0.1626993D+01</td>
<td>0.9395887D+01</td>
</tr>
<tr>
<td>1.0</td>
<td>0.3025821D+00</td>
<td>0.1000000D+01</td>
<td>0.3304888D+01</td>
<td>24</td>
<td>0.1638514D+01</td>
<td>0.9512469D+01</td>
</tr>
<tr>
<td>1.1</td>
<td>0.2938675D+00</td>
<td>0.1010093D+01</td>
<td>0.3304888D+01</td>
<td>25</td>
<td>0.1694093D+01</td>
<td>0.9641324D+01</td>
</tr>
<tr>
<td>1.2</td>
<td>0.2803548D+00</td>
<td>0.1041068D+01</td>
<td>0.3713398D+01</td>
<td>27</td>
<td>0.1794608D+01</td>
<td>0.9782450D+01</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2456831D+00</td>
<td>0.1165306D+01</td>
<td>0.4743127D+01</td>
<td>34</td>
<td>0.2142302D+01</td>
<td>0.1010152D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.2093845D+00</td>
<td>0.1363176D+01</td>
<td>0.6510394D+01</td>
<td>46</td>
<td>0.2723656D+01</td>
<td>0.1046967D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.1766067D+00</td>
<td>0.1630809D+01</td>
<td>0.9234126D+01</td>
<td>64</td>
<td>0.3614762D+01</td>
<td>0.1088692D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.1487827D+00</td>
<td>0.1974039D+01</td>
<td>0.1326794D+02</td>
<td>90</td>
<td>0.4933939D+01</td>
<td>0.1135325D+02</td>
</tr>
</tbody>
</table>

TABLE 6.4.1: Comparison of condition numbers and number of iterations with various values of ω for Rayleigh-Ritz method

<table>
<thead>
<tr>
<th>ω</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No.of iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2026709D+00</td>
<td>0.2325926D+01</td>
<td>0.1147637D+02</td>
<td>86</td>
<td>0.4277791D+01</td>
<td>0.9728057D+01</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2360475D+00</td>
<td>0.1797005D+01</td>
<td>0.7612895D+01</td>
<td>58</td>
<td>0.2943039D+01</td>
<td>0.7305957D+01</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2686239D+00</td>
<td>0.1422261D+01</td>
<td>0.5294619D+01</td>
<td>42</td>
<td>0.2163249D+01</td>
<td>0.6638062D+01</td>
</tr>
<tr>
<td>0.6</td>
<td>0.2928175D+00</td>
<td>0.1176360D+01</td>
<td>0.4017381D+01</td>
<td>32</td>
<td>0.1760272D+01</td>
<td>0.6994318D+01</td>
</tr>
<tr>
<td>0.8</td>
<td>0.2980123D+00</td>
<td>0.1041368D+01</td>
<td>0.3494380D+01</td>
<td>28</td>
<td>0.1627091D+01</td>
<td>0.7493076D+01</td>
</tr>
<tr>
<td>0.9</td>
<td>0.2914303D+00</td>
<td>0.1010095D+01</td>
<td>0.3465993D+01</td>
<td>28</td>
<td>0.1643301D+01</td>
<td>0.7795893D+01</td>
</tr>
<tr>
<td>1.0</td>
<td>0.2793971D+00</td>
<td>0.1000000D+01</td>
<td>0.3579136D+01</td>
<td>29</td>
<td>0.1710379D+01</td>
<td>0.8134336D+01</td>
</tr>
<tr>
<td>1.1</td>
<td>0.2634242D+00</td>
<td>0.1010093D+01</td>
<td>0.3834474D+01</td>
<td>31</td>
<td>0.1827817D+01</td>
<td>0.8508405D+01</td>
</tr>
<tr>
<td>1.2</td>
<td>0.2452164D+00</td>
<td>0.1041136D+01</td>
<td>0.4245785D+01</td>
<td>34</td>
<td>0.1996956D+01</td>
<td>0.8918099D+01</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2075059D+00</td>
<td>0.1164521D+01</td>
<td>0.5611990D+01</td>
<td>44</td>
<td>0.2504315D+01</td>
<td>0.9844364D+01</td>
</tr>
<tr>
<td>1.6</td>
<td>0.1731075D+00</td>
<td>0.1354018D+01</td>
<td>0.7821833D+01</td>
<td>61</td>
<td>0.3278814D+01</td>
<td>0.1051313D+01</td>
</tr>
<tr>
<td>1.8</td>
<td>0.1441278D+00</td>
<td>0.1602932D+01</td>
<td>0.1112160D+02</td>
<td>85</td>
<td>0.4401977D+01</td>
<td>0.1212440D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.1204962D+01</td>
<td>0.1917900D+01</td>
<td>0.1591727D+02</td>
<td>122</td>
<td>0.6002650D+01</td>
<td>0.1347817D+02</td>
</tr>
</tbody>
</table>

TABLE 6.4.2: Comparison of condition numbers and number of iterations with various values of ω for the Collocation method
<table>
<thead>
<tr>
<th>( w )</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No. of iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2316800D+00</td>
<td>0.2239306D+01</td>
<td>0.9665514D+01</td>
<td>75</td>
<td>0.3648629D+01</td>
<td>0.8479763D+01</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2687453D+00</td>
<td>0.1758508D+01</td>
<td>0.6543400D+01</td>
<td>52</td>
<td>0.2582017D+01</td>
<td>0.6507048D+01</td>
</tr>
<tr>
<td>0.4</td>
<td>0.3061642D+00</td>
<td>0.1409607D+01</td>
<td>0.4604088D+01</td>
<td>36</td>
<td>0.1941012D+01</td>
<td>0.6323937D+01</td>
</tr>
<tr>
<td>0.6</td>
<td>0.3365678D+00</td>
<td>0.1174159D+01</td>
<td>0.3488626D+01</td>
<td>28</td>
<td>0.1596998D+01</td>
<td>0.6563762D+01</td>
</tr>
<tr>
<td>0.8</td>
<td>0.3477264D+00</td>
<td>0.1041299D+01</td>
<td>0.2994593D+01</td>
<td>24</td>
<td>0.1470384D+01</td>
<td>0.6899517D+01</td>
</tr>
<tr>
<td>0.9</td>
<td>0.3428366D+00</td>
<td>0.1010094D+01</td>
<td>0.2946284D+01</td>
<td>24</td>
<td>0.1474483D+01</td>
<td>0.7103369D+01</td>
</tr>
<tr>
<td>1.0</td>
<td>0.3311614D+00</td>
<td>0.1000000D+01</td>
<td>0.3019676D+01</td>
<td>25</td>
<td>0.1519997D+01</td>
<td>0.7331203D+01</td>
</tr>
<tr>
<td>1.1</td>
<td>0.3142880D+00</td>
<td>0.1010091D+01</td>
<td>0.3213902D+01</td>
<td>26</td>
<td>0.1606323D+01</td>
<td>0.7583019D+01</td>
</tr>
<tr>
<td>1.2</td>
<td>0.2942425D+00</td>
<td>0.1041002D+01</td>
<td>0.3537906D+01</td>
<td>29</td>
<td>0.1734231D+01</td>
<td>0.7858818D+01</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2514235D+00</td>
<td>0.1161076D+01</td>
<td>0.4618011D+01</td>
<td>37</td>
<td>0.2124148D+01</td>
<td>0.8482364D+01</td>
</tr>
<tr>
<td>1.6</td>
<td>0.2114908D+00</td>
<td>0.1341015D+01</td>
<td>0.6340775D+01</td>
<td>50</td>
<td>0.2721431D+01</td>
<td>0.9201839D+01</td>
</tr>
<tr>
<td>1.8</td>
<td>0.1773603D+00</td>
<td>0.1573703D+01</td>
<td>0.8872918D+01</td>
<td>70</td>
<td>0.3581791D+01</td>
<td>1.001724D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.1492049D+00</td>
<td>0.1863936D+01</td>
<td>0.1249246D+02</td>
<td>97</td>
<td>0.4792183D+01</td>
<td>1.092858D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.4.3:** Comparison of condition numbers and number of iterations with various values of \( w \) for the Overdetermined Collocation method

<table>
<thead>
<tr>
<th>( w )</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No. of iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.3568047D+00</td>
<td>0.1947718D+01</td>
<td>0.5458779D+01</td>
<td>41</td>
<td>0.2232709D+01</td>
<td>0.5630904D+01</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4045573D+00</td>
<td>0.1616490D+01</td>
<td>0.3995703D+01</td>
<td>30</td>
<td>0.1753035D+01</td>
<td>0.4683779D+01</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4540039D+00</td>
<td>0.1355964D+01</td>
<td>0.2986679D+01</td>
<td>23</td>
<td>0.1440500D+01</td>
<td>0.4425043D+01</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4987428D+00</td>
<td>0.1162479D+01</td>
<td>0.2330819D+01</td>
<td>18</td>
<td>0.1257306D+01</td>
<td>0.4504292D+01</td>
</tr>
<tr>
<td>0.8</td>
<td>0.5246752D+00</td>
<td>0.1040773D+01</td>
<td>0.1983652D+01</td>
<td>15</td>
<td>0.1178140D+01</td>
<td>0.4615241D+01</td>
</tr>
<tr>
<td>0.9</td>
<td>0.5245020D+00</td>
<td>0.1010079D+01</td>
<td>0.1925787D+01</td>
<td>15</td>
<td>0.1172643D+01</td>
<td>0.4682602D+01</td>
</tr>
<tr>
<td>1.0</td>
<td>0.5139657D+00</td>
<td>0.1000000D+01</td>
<td>0.1945655D+01</td>
<td>15</td>
<td>0.1188428D+01</td>
<td>0.4757889D+01</td>
</tr>
<tr>
<td>1.1</td>
<td>0.4945704D+00</td>
<td>0.1010054D+01</td>
<td>0.2042286D+01</td>
<td>16</td>
<td>0.1225042D+01</td>
<td>0.4841101D+01</td>
</tr>
<tr>
<td>1.2</td>
<td>0.4692622D+00</td>
<td>0.1039165D+01</td>
<td>0.2214466D+01</td>
<td>17</td>
<td>0.1282446D+01</td>
<td>0.4932237D+01</td>
</tr>
<tr>
<td>1.4</td>
<td>0.4118436D+00</td>
<td>0.1133142D+01</td>
<td>0.2751390D+01</td>
<td>21</td>
<td>0.1461227D+01</td>
<td>0.5138285D+01</td>
</tr>
<tr>
<td>1.6</td>
<td>0.3559032D+00</td>
<td>0.1256021D+01</td>
<td>0.3529110D+01</td>
<td>27</td>
<td>0.1730858D+01</td>
<td>0.5376032D+01</td>
</tr>
<tr>
<td>1.8</td>
<td>0.3063456D+00</td>
<td>0.1401926D+01</td>
<td>0.4576288D+01</td>
<td>35</td>
<td>0.2101740D+01</td>
<td>0.5645478D+01</td>
</tr>
<tr>
<td>2.0</td>
<td>0.2640067D+00</td>
<td>0.1569697D+01</td>
<td>0.5945672D+01</td>
<td>45</td>
<td>0.2588647D+01</td>
<td>0.5946625D+01</td>
</tr>
</tbody>
</table>

**TABLE 6.4.4:** Comparison of condition numbers and number of iterations with various values of \( w \) for the Least Squares method
<table>
<thead>
<tr>
<th>y location (x=0.5)</th>
<th>-0.6</th>
<th>-0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=3</td>
<td>n=6</td>
</tr>
<tr>
<td>Rayleigh-Ritz/</td>
<td>0.2670674</td>
<td>0.2680357</td>
</tr>
<tr>
<td>Galerkin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collocation</td>
<td>0.2635437</td>
<td>0.2630235</td>
</tr>
<tr>
<td>Overdetermined</td>
<td>0.2631022</td>
<td>0.2630276</td>
</tr>
<tr>
<td>Collocation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Least Squares</td>
<td>0.2606128</td>
<td>0.2635684</td>
</tr>
<tr>
<td>Exact Solution</td>
<td>0.2679937</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 6.4.5a**: Comparisons of approximate solutions to Problem 3

<table>
<thead>
<tr>
<th>y location (x=0.5)</th>
<th>-0.6</th>
<th>-0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=3</td>
<td>n=6</td>
</tr>
<tr>
<td>Rayleigh-Ritz/</td>
<td>9.263×10^{-3}</td>
<td>-4.199×10^{-4}</td>
</tr>
<tr>
<td>Galerkin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collocation</td>
<td>4.450×10^{-2}</td>
<td>4.970×10^{-2}</td>
</tr>
<tr>
<td>Overdetermined</td>
<td>4.892×10^{-2}</td>
<td>4.966×10^{-2}</td>
</tr>
<tr>
<td>Collocation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Least Squares</td>
<td>7.381×10^{-2}</td>
<td>4.425×10^{-2}</td>
</tr>
</tbody>
</table>

**TABLE 6.4.5b**: Errors for Problem 3
is used where $k$ and $\ell$ go from 0 to 2 (for $n=6$) according to the following table (see also (5.2.17)).

<table>
<thead>
<tr>
<th></th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$Q_3$</th>
<th>$Q_4$</th>
<th>$Q_5$</th>
<th>$Q_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$\ell$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**TABLE 6.4.6a**

or in the FORTRAN statement,

```
MF=2
DO 1 KA=O,MF
DO 1 K=KA,O,-1
  L=KA-K+1
  ................
1 CONTINUE
```

In Table 6.4.7, the values of $k$ and $\ell$ in (6.4.1) go from 0 to 2 (for $n=6$) by the following table,

<table>
<thead>
<tr>
<th></th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$Q_3$</th>
<th>$Q_4$</th>
<th>$Q_5$</th>
<th>$Q_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$\ell$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**TABLE 6.4.6b**

or in the FORTRAN statement,

```
MF=2
DO 1 K=O,MF
DO 1 L=O,K
  ................
1 CONTINUE
```

It is apparent that from Table 6.4.7, the Least Squares method gives a smaller condition number. As in the previous two examples, we now wish to determine the best basis functions for the Least Squares method.
<table>
<thead>
<tr>
<th>n</th>
<th>Rayleigh-Ritz</th>
<th>Collocation</th>
<th>Overdetermined Collocation</th>
<th>Least Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P-Condition Number</td>
<td>No. of iters</td>
<td>K-Condition</td>
<td>No. of iters</td>
</tr>
<tr>
<td>3</td>
<td>( \omega )</td>
<td>12.201</td>
<td>82</td>
<td>0.2257\times10^2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.02388\times10^3</td>
<td>1459</td>
<td>0.3807\times10^4</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0.7276\times10^4</td>
<td>&gt;</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>( \omega )</td>
<td>2.9671</td>
<td>21</td>
<td>0.3898\times10^1</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>0.2247\times10^2</td>
<td>161</td>
<td>0.4375\times10^3</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.3529\times10^3</td>
<td>2515</td>
<td>*</td>
</tr>
</tbody>
</table>

* Note: The minimum eigenvalue of the coefficient matrix was too small for the available routine to accurately evaluate the P- and K-Condition numbers.

**TABLE 6.4.7:** Comparisons of condition numbers and the number of iterations using polynomial approximation functions.
FIG. 6.1.1: P-CON Vs W - PROBLEM 3
FIG 6.4.2 - ERROR DISTRIBUTION, POLY APP
Selection of the 'Best' Basis Functions

From equation (5.2.17) we have the general approximation $u_n(x,y)$ for $u_d$ by,

$$u_n(x,y) = \psi(x,y) + \sum_{k=1}^{n} a_k Q_k(x,y), \quad (6.4.2)$$

where $Q_k(x,y)$ is the basis function to be chosen from the sequence of transcendental functions and the trigonometric functions.

Sequence b - Transcendental Basis Functions

In this case, taking advantage of the symmetry about $y$, we have,

$$Q_k = x(e^{kx} - e^k)(1+y)(e^{2\xi y} - e^{2\xi}), \quad k, \xi = 1, 2, 3, \quad (6.4.3)$$

where $k$ and $\xi$ are chosen as in Table 6.4.6a.

By this choice, the homogeneous boundary condition in (5.2.15) is satisfied. From (6.4.3) we have,

$$AQ_k = (1+y)(e^{2\xi y} - e^{2\xi})ke^{kx} (2+kx) + x(e^{kx} - e^k)2\xi e^{2\xi y}(2+2k(1+y)), \quad k, \xi = 1, 2, 3. \quad (6.4.4)$$

The Least Squares systems are given by

$$\begin{bmatrix}
0.1993D+04 & 0.9929D+04 & 0.2132D+05 & 0.3795D+05 & 0.1048D+06 & 0.1915D+06 \\
0.9929D+04 & 0.5229D+05 & 0.1048D+06 & 0.2085D+06 & 0.5429D+06 & 0.9321D+06 \\
0.2132D+05 & 0.1048D+06 & 0.2559D+06 & 0.3959D+05 & 0.1235D+07 & 0.2513D+07 \\
0.3795D+05 & 0.2085D+06 & 0.3959D+05 & 0.8581D+06 & 0.2135D+07 & 0.3486D+07 \\
0.1048D+06 & 0.5429D+06 & 0.1235D+07 & 0.2135D+07 & 0.6234D+07 & 0.1196D+08 \\
0.1915D+06 & 0.9321D+06 & 0.2513D+07 & 0.3486D+07 & 0.1196D+08 & 0.2641D+08 \\
\end{bmatrix}$$

$$\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6 \\
\end{bmatrix} = \begin{bmatrix}
-0.4805D+02 \\
-0.2008D+03 \\
-0.4304D+03 \\
-0.6712D+03 \\
-0.1798D+04 \\
-0.3360D+04 \\
\end{bmatrix} \quad (6.4.5)$$

The approximate solutions and the errors are given in Table 6.4.10a and 6.4.10b respectively. Other results are given in Tables 6.4.8 and 6.4.11 and Figure 6.4.1.
<table>
<thead>
<tr>
<th>ω</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition No. of</th>
<th>N-Condition No.</th>
<th>M-Condition No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2038892D-01</td>
<td>0.2882340D+01</td>
<td>0.1413680D+03</td>
<td>1002</td>
<td>0.9372371D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2455662D-01</td>
<td>0.2011161D+01</td>
<td>0.8189894D+02</td>
<td>590</td>
<td>0.6106318D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2856533D-01</td>
<td>0.1481214D+01</td>
<td>0.5185357D+02</td>
<td>382</td>
<td>0.6525332D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>0.3121443D-01</td>
<td>0.1184722D+01</td>
<td>0.3795429D+02</td>
<td>286</td>
<td>0.7219419D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>0.3124879D-01</td>
<td>0.1041584D+01</td>
<td>0.333198D+02</td>
<td>254</td>
<td>0.8099731D+02</td>
</tr>
<tr>
<td>0.9</td>
<td>0.3019033D-01</td>
<td>0.1010100D+01</td>
<td>0.3345772D+02</td>
<td>256</td>
<td>0.8609722D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>0.2856907D-01</td>
<td>0.1000000D+01</td>
<td>0.3500289D+02</td>
<td>268</td>
<td>0.9166269D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>0.2657647D-01</td>
<td>0.1010100D+01</td>
<td>0.3800729D+02</td>
<td>291</td>
<td>0.9769373D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>0.2440229D-01</td>
<td>0.1041580D+01</td>
<td>0.4268368D+02</td>
<td>327</td>
<td>1.041903D+03</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2007711D-01</td>
<td>0.1184261D+01</td>
<td>0.5898564D+02</td>
<td>447</td>
<td>1.185802D+03</td>
</tr>
<tr>
<td>1.6</td>
<td>0.1628792D-01</td>
<td>0.1475605D+01</td>
<td>0.9059504D+02</td>
<td>675</td>
<td>1.348324D+03</td>
</tr>
<tr>
<td>1.8</td>
<td>0.1321103D-01</td>
<td>0.1983772D+01</td>
<td>0.1501603D+03</td>
<td>1095</td>
<td>0.5044957D+03</td>
</tr>
<tr>
<td>2.0</td>
<td>0.1078900D-01</td>
<td>0.2799982D+01</td>
<td>0.2595220D+03</td>
<td>1860</td>
<td>0.8684676D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.4.8:** Condition numbers and the number of iterations for the Least Squares method using exponential basis functions

**Sequence c - Trigonometric Basis Functions**

In this case, \( Q_k = \sin k\pi x \sin 2\pi y, \ k, \ell = 0,1,2 \)

satisfies the homogeneous boundary condition in (5.2.15). We have,

\[
AQ_k = -(k\pi)^2 \sin(2\pi y) \sin k\pi x - (2\pi)^2 \sin(k\pi x)\sin(2\pi y)
\]

\( k, \ell = 1,2,3. \) \ (6.4.6)

The Right hand side of the Least Squares system (5.7.22) will involve the integration

\[
\int_R \int fAQ_k dR,
\]

where \( f = -\frac{\pi^2}{4} \cos \frac{\pi}{2} y[1-(1-e^{-\pi})x] \).

Therefore (6.4.7) becomes,

\[
\left[ \left[ \frac{1}{1} \right] - \frac{\pi^2}{4} \cos \frac{\pi}{2} y[1-(1-e^{-\pi})x] AQ_k \right] dx dy
\]

\ (6.4.8)
Consider this integration with respect to the variable $y$, then,

$$
\int_{-1}^{1} \cos \frac{\pi}{2} y \sin 2\pi \alpha y \, dy, \quad \alpha = 1, 2, 3. \quad (6.4.9)
$$

It can be shown that for any $\alpha$ (6.4.9) will be zero. Therefore, this choice of basis function could not give a correct approximation to the problem 3.

**Sequence d**

Combination of trigonometric and exponential basis functions.

In view of sequence c, we will now choose the basis functions as

$$
Q_k = \sin k\pi x(1+y)(e^{2\pi y} - e^{2\pi \alpha}), \quad k, \alpha = 1, 2, 3, \quad (6.4.10)
$$

where $k$ and $\alpha$ are chosen similar to the Table 6.4.6a. The homogeneous boundary condition (5.2.15) is satisfied. From (6.4.10) we have,

$$
AQ_k = \sin(k\pi x)2\pi e^{2\pi y}(1+y)(1+y)(e - e^{2\pi \alpha} - (1+y)(e^{2\pi y} - e^{2\pi \alpha})(k\pi)^2 \sin(k\pi x)
$$

$$
k, \alpha = 1, 2, 3. \quad (6.4.11)
$$

The Least Squares system for this basis function is given by,

$$
\begin{bmatrix}
0.5062D+04 & 0 & 0.5595D+05 & 0 & 0 & 0.5150D+06 \\
0 & 0 & 0.4829D+05 & 0 & 0 & 0.4650D+06 \\
0.5595D+05 & 0 & 0.7020D+06 & 0 & 0 & 0.7100D+07 \\
0 & 0 & 0 & 0.2221D+06 & 0 & 0 \\
0 & 0.4650D+06 & 0 & 0 & 0.4691D+07 & 0 \\
0.5150D+06 & 0 & 0.7100D+07 & 0 & 0 & 0.7691D+08
\end{bmatrix}
$$

$$
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6
\end{bmatrix} =
\begin{bmatrix}
0.8133D+02 \\
0.7067D+02 \\
0.7284D+03 \\
0.2006D+03 \\
0.8121D+03 \\
0.5686D+04
\end{bmatrix} \quad (6.4.12)
$$
The relevant results are given in Tables 6.4.9, 6.4.10 and 6.4.11 and Figure 6.4.3.

It is apparent that the polynomial basis functions possess the least P-condition numbers. Consequently, the rate of convergence is fastest. We conclude therefore that the Least Squares method with polynomial basis functions is the 'best' choice for this problem.

<table>
<thead>
<tr>
<th>(\omega)</th>
<th>Smallest Eigenvalue</th>
<th>Largest Eigenvalue</th>
<th>P-Condition Number</th>
<th>No. of iters</th>
<th>N-Condition Number</th>
<th>M-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>6.136855D-01</td>
<td>1.938631D+01</td>
<td>0.3158998D+02</td>
<td>233</td>
<td>0.1188128D+02</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>0.2</td>
<td>7.394632D-01</td>
<td>1.608883D+01</td>
<td>0.2175745D+02</td>
<td>162</td>
<td>0.8578068D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>0.4</td>
<td>8.826554D-01</td>
<td>1.347876D+01</td>
<td>0.152706D+02</td>
<td>114</td>
<td>0.6385137D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>0.6</td>
<td>1.028166D+00</td>
<td>1.157119D+01</td>
<td>0.1125421D+02</td>
<td>86</td>
<td>0.5015064D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>0.8</td>
<td>1.144178D+00</td>
<td>1.039794D+01</td>
<td>0.9087697D+01</td>
<td>70</td>
<td>0.4268793D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.0</td>
<td>1.189710D+00</td>
<td>1.000000D+01</td>
<td>0.840541D+01</td>
<td>65</td>
<td>0.4032233D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.1</td>
<td>1.177946D+00</td>
<td>1.009987D+01</td>
<td>0.8574132D+01</td>
<td>67</td>
<td>0.4090812D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.2</td>
<td>1.144178D+00</td>
<td>1.039794D+01</td>
<td>0.9887596D+01</td>
<td>71</td>
<td>0.4268793D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.3</td>
<td>1.092462D+00</td>
<td>1.089017D+01</td>
<td>0.996847D+01</td>
<td>77</td>
<td>0.4572996D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.4</td>
<td>1.028166D+00</td>
<td>1.157119D+01</td>
<td>0.1125421D+02</td>
<td>87</td>
<td>0.5015063D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.5</td>
<td>0.9566981D-01</td>
<td>1.243558D+01</td>
<td>0.1299844D+02</td>
<td>99</td>
<td>0.5611742D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.6</td>
<td>0.8826555D-01</td>
<td>1.1347876D+01</td>
<td>0.1527069D+02</td>
<td>116</td>
<td>0.6385136D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.7</td>
<td>0.8094766D-01</td>
<td>1.1469727D+01</td>
<td>0.1815652D+02</td>
<td>137</td>
<td>0.73628354D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>1.8</td>
<td>0.7394633D-01</td>
<td>1.160883D+01</td>
<td>0.2175744D+02</td>
<td>163</td>
<td>0.8578066D+01</td>
<td>0.2521623D+02</td>
</tr>
<tr>
<td>2.0</td>
<td>0.6136855D-01</td>
<td>1.1938631D+01</td>
<td>0.3158998D+02</td>
<td>234</td>
<td>0.1188128D+02</td>
<td>0.2521623D+02</td>
</tr>
</tbody>
</table>

**TABLE 6.4.9:** Comparison of condition numbers and number of iterations with various values of \(\omega\) for the Least Squares method using exp-sine basis functions.
### TABLE 6.4.10a: Comparison of approximate solutions to the boundary value problem of Problem 3

<table>
<thead>
<tr>
<th>y location (x=0.5)</th>
<th>-0.6</th>
<th>-0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basis Functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polynomials</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>0.2606128</td>
<td>0.4233278</td>
</tr>
<tr>
<td>n=6</td>
<td>0.2635684</td>
<td>0.4276188</td>
</tr>
<tr>
<td>Exponential</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>0.2866713</td>
<td>0.4488873</td>
</tr>
<tr>
<td>n=6</td>
<td>0.2774671</td>
<td>0.4376549</td>
</tr>
<tr>
<td>Trigonometric and Exponential</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>0.2899569</td>
<td>0.4551459</td>
</tr>
<tr>
<td>n=6</td>
<td>0.2802959</td>
<td>0.4429611</td>
</tr>
<tr>
<td>Exact</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>0.2679937</td>
<td>0.4336229</td>
</tr>
<tr>
<td>n=6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TABLE 6.4.10b: Errors to the boundary value problem of Problem 3

<table>
<thead>
<tr>
<th>y location (x=0.5)</th>
<th>-0.6</th>
<th>-0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basis Functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polynomial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>0.7381×10^{-2}</td>
<td>0.1030×10^{-1}</td>
</tr>
<tr>
<td>n=6</td>
<td>0.4425×10^{-2}</td>
<td>0.6004×10^{-2}</td>
</tr>
<tr>
<td>Exponential</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>-0.1868×10^{-1}</td>
<td>-0.1526×10^{-1}</td>
</tr>
<tr>
<td>n=6</td>
<td>-0.9473×10^{-2}</td>
<td>-0.4032×10^{-2}</td>
</tr>
<tr>
<td>Trigonometric and Exponential</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n=3</td>
<td>-0.2196×10^{-1}</td>
<td>-0.2152×10^{-1}</td>
</tr>
<tr>
<td>n=6</td>
<td>0.1230×10^{-1}</td>
<td>-0.9338×10^{-2}</td>
</tr>
<tr>
<td>n</td>
<td>Polynomial</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>------------</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>P-Condition Number</td>
<td>No.of iters.</td>
</tr>
<tr>
<td>3</td>
<td>0.3159×10²</td>
<td>233</td>
</tr>
<tr>
<td>6</td>
<td>0.1906×10⁵</td>
<td>8123</td>
</tr>
<tr>
<td>10</td>
<td>&gt; 0.3910×10⁵</td>
<td>&gt;</td>
</tr>
</tbody>
</table>

**TABLE 6.4.11:** Comparisons of condition numbers and the number of iterations for the least Squares method using three different basis functions.
FIG 6.4.3  ERROR DISTRIBUTION (LEAST SQUARES)
PROBLEM THREE
6.5 THE FINITE ELEMENT METHOD

Several considerations emerge from the variational methods described in the earlier sections. First, the choice of coordinate functions determines how good the approximate solution is. Second, it should be emphasized that the completeness property of sequence of functions is important. Third, for more parameters or more complex problems the integration connected with the method can be difficult. Furthermore, with complicated boundaries and boundary conditions in conjunction with two- or three-dimensional problems, it becomes difficult to choose suitable coordinate functions.

Some of the difficulties mentioned above can be overcome by the method of Finite Elements. The assumed trial functions in the finite element method are not defined over the whole solution domain, and they have to satisfy no boundary conditions, but only certain continuity conditions. Thus, the method can be applied to problems of more complex shape.

The basic idea of the finite element method is to divide the solution domain into a finite number of subdomains (elements). These elements are connected only at node points in the domain and on the element boundaries. In this way the solution domain is discretized and represented as a patchwork of elements. Therefore the coefficient matrix $A$ in (6.1.1) will be sparse. That is it has few non-zero elements and they are clustered around the main diagonal. Thus, the solution of (6.1.1) can be obtained with accuracy and economy by the use of sparse matrix techniques so that advantage can be taken of the sparseness in order to reduce both the computation time and the storage requirement as discussed in Chapter 5.

Frequently the boundaries of the finite elements are straight lines or planes. In the one-dimensional case a curve (the solution) is approximated by a series of straight line segments. In two (or more) dimensions, the
solution is approximated by portions of planes (or hyper-planes). The domain of applicability of each segment of the piecewise linear function is termed a 'finite element'.

A convenient way to obtain an appreciation for the finite element method is to proceed to a simple example. Let us apply the finite element method to the approximate minimization of the function

$$J[y] = \int_0^1 \left[ \frac{1}{2} (y')^2 + \alpha y \right] dx, \quad (6.5.1a)$$

with the boundary,

$$y(0) = y(1) = 0. \quad (6.5.1b)$$

Let us use as an approximation a curve consisting of five straight line segments having their end points equally spaced in the x direction as given in Figure 6.5.1

![Figure 6.5.1: Straight-line segments used to obtain an approximate minimization of (6.5.1)](image)

Each interval \((x_i, x_{i+1})\) is an 'element' in this case, and an approximate function is,

$$y = y_i + \frac{1}{(x_{i+1}-x_i)} \left( y_{i+1} - y_i \right) \left( x - x_i \right), \quad (6.5.2)$$

or, because \(x_1 = 1/5, \ x_2 = 2/5, \ etc.,\)
Also, in any interval we have,

\[ y' = 5(y_{i+1} - y_i) \]  \hspace{1cm} (6.5.4)

The functional (6.5.1) can now be written as,

\[ J[y] = \frac{4}{5} \sum_{i=0}^{X_{i+1}} \left[ 25(y_{i+1} - y_i)^2 + 5y_i + 5s(x_i)(y_{i+1} - y_i) \right] dx \]  \hspace{1cm} (6.5.5)

or,

\[ J[y] = \sum_{i=0}^{X_{i+1}} \frac{5}{2}(y_{i+1} - y_i)^2 + \frac{\alpha}{5} y_i + \frac{5\alpha}{2}(x_i^2 - x_i^2)(y_{i+1} - y_i) - \alpha x_i(y_{i+1} - y_i) \]  \hspace{1cm} (6.5.6)

The functional has again been reduced to an ordinary function. The variables or parameters in this case are \( y_1, y_2, y_3, y_4 \) using the previously stated boundary conditions \( y(0) = y(1) = 0 \). Approximate minimization requires that,

\[ \frac{\partial J}{\partial y_i} = 0, \hspace{0.5cm} i=1,2,3 \text{ and } 4. \]

Thus, the equations obtained are,

\[
\begin{align*}
\frac{\partial J}{\partial y_1} &= 10y_1 - 5y_2 + \alpha/5 = 0, \\
\frac{\partial J}{\partial y_2} &= -5y_1 + 10y_2 - 5y_3 + \alpha/5 = 0, \\
\frac{\partial J}{\partial y_3} &= -5y_2 + 10y_3 - 5y_4 + \alpha/5 = 0, \\
\frac{\partial J}{\partial y_4} &= -5y_2 + 10y_4 + \alpha/5 = 0.
\end{align*}
\]  \hspace{1cm} (6.5.7)

In obtaining each equation of the set (6.5.7) two contributions came from (6.5.6), for example, \( y_3 \) is a parameter in two line segments, one from \( (x_2, y_2) \) to \( (x_3, y_3) \), the other from \( (x_3, y_3) \) to \( (x_4, y_4) \).

Equation (6.5.7) can be rewritten as

\[
\begin{bmatrix}
2 & -1 & 0 \\
-1 & 2 & -1 \\
-1 & 2 & -1 \\
0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4
\end{bmatrix}
= \frac{\alpha}{5}
\begin{bmatrix}
-1 \\
-1 \\
-1 \\
-1
\end{bmatrix}
\]  \hspace{1cm} (6.5.8)
Several advantages of the finite element method may be appreciated from the above example:

1. The use of variable size elements (unequal $\Delta x$ in the one-dimensional case) presents no problem.
2. The boundary conditions are matched easily by the coordinate functions.
3. Integrations are carried out easily because only linear functions are used.

The application to problems of higher dimensions follows essentially the same procedure. We will now discuss the application of the finite element method to certain two-dimensional steady-flow problems in porous media.

Consider the problem of finding a solution to,

$$\frac{\partial}{\partial x}(k \frac{\partial u}{\partial x}) + \frac{\partial}{\partial y}(k \frac{\partial u}{\partial y}) = f(x,y),$$

(6.5.9)

in a region $R$ with values of $u$ specified on the boundary curve, $C$. $K$ may vary spatially but not as a function of $u$. The variational formulation of this problem requires the minimization of the functional

$$J[u] = \int_R \left\{ \frac{1}{2} [K(\frac{\partial u}{\partial x})^2 + K(\frac{\partial u}{\partial y})^2] + fU \right\} \, dx \, dy,$$

(6.5.10)

with the same boundary conditions to be satisfied.

![Figure 6.5.2: A typical triangular element with nodes i, j and k](image)

The region $R$ is divided into triangular elements as in Figure 6.5.2. Some other shape may also be used. The linear approximating function in each element has the form,

$$u = \beta_1 x + \beta_2 y + \beta_3.$$

(6.5.11)
The parameters that we want to use in the minimization (i.e. what we want to solve for) are the nodal values of U. Therefore, we solve for the coefficients $\beta_1, \beta_2$ and $\beta_3$ of (6.5.11) by using the fact that the plane used in approximation on the element $i,j,k$ must pass through the nodal values. Hence,

\[
\begin{align*}
  u_i &= \beta_1 x_i + \beta_2 y_i + \beta_3 \\
  u_j &= \beta_1 x_j + \beta_2 y_j + \beta_3 \\
  u_k &= \beta_1 x_k + \beta_2 y_k + \beta_3
\end{align*}
\] (6.5.12)

By using Cramer's Rule to solve (6.5.12) for $\beta_1, \beta_2$ and $\beta_3$ and substituting the result into (6.5.11) we obtain,

\[
u = (N_i u_i + N_j u_j + N_k u_k),
\] (6.5.13)

where,

\[
\begin{align*}
  N_i &= [(x_j y_k - x_k y_j) + (y_i - y_k)x + (x_k - x_j)y]/2\Delta, \\
  N_j &= [(x_k y_i - x_i y_k) + (y_j - y_i)x + (x_i - x_k)y]/2\Delta, \\
  N_k &= [(x_i y_j - x_j y_i) + (y_i - y_j)x + (x_j - x_i)y]/2\Delta,
\end{align*}
\]

and

\[
2\Delta = \det \begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{vmatrix} = \text{twice the area of triangle } i,j,k.
\]

Each element contributes to a portion of the total value of the integral (6.5.10) and the set of difference equations is obtained by the conditions for minimization.

\[
\frac{\partial J}{\partial u_m} = 0, \quad m=1,2,...,n,
\] (6.5.14)

where $n$ is the number of nodes. Many elements may contribute to $\partial J/\partial u_m$ for each $m$ (i.e., a node generally will be common to several elements), so that (6.5.14) may be written as,

\[
\sum_e \frac{\partial J}{\partial u_m} = 0, \quad m=1,...,n.
\] (6.5.15)

where $J_e$ denotes the contribution of the $e$th element to (6.5.10) and the summation is over all contributing elements.
By substituting (6.5.13) into (6.5.10) we obtain,

\[ J_e = \int \int \{ \sum_i \left( \frac{\partial N_i}{\partial x} u_i + \frac{\partial N_i}{\partial y} u_j + \frac{\partial N_k}{\partial x} u_k \right)^2 
+ k \left( \frac{\partial N_i}{\partial y} u_i + \frac{\partial N_j}{\partial y} u_j + \frac{\partial N_k}{\partial y} u_k \right)^2 
+ f \sum_{i<j} \left( N_i u_i + N_j u_j + N_k u_k \right) \} \, dx \, dy. \]  

(6.5.16)

We are interested in computing \( \partial J/\partial u_m \) for a typical \( m \)th node, since the computations for all nodes are similar. As stated previously, node \( i \) is common to several elements so that we must ultimately compute,

\[ \partial J/\partial u_i = \sum_e \frac{\partial J_e}{\partial u_i}. \]

Fortunately, one term of the summation gives the form for the other terms. Therefore, it is necessary to explicitly consider the contribution of only a typical finite element. Differentiating (6.5.16) by \( u_i \) gives,

\[ \frac{\partial J_e}{\partial u_i} = \int \int \left\{ \frac{\partial^2 N_i}{\partial x^2} u_i + \frac{\partial^2 N_i}{\partial x \partial y} u_j + \frac{\partial^2 N_k}{\partial x^2} u_k \right\} \left( \frac{\partial N_i}{\partial x} + \frac{\partial N_j}{\partial y} + \frac{\partial N_k}{\partial y} \right) + k \left( \frac{\partial^2 N_i}{\partial y^2} u_i + \frac{\partial^2 N_j}{\partial y^2} u_j + \frac{\partial^2 N_k}{\partial y^2} u_k \right) \right\} \, dx \, dy. \]  

(6.5.17)

By using the definitions of \( N_i, N_j, \) and \( N_k, \) we may write (6.5.17)

\[ \frac{\partial J_e}{\partial u_i} = \left\{ \left[ (y_j-y_k) u_i + (y_k-y_i) u_j + (y_i-y_j) u_k \right] \left[ x_i x_j k^2 + (x_k-x_j) u_i + (x_j-x_k) u_j \right] \right\} \int \int_{e} K \, dx \, dy + \int \int_{e} f \left[ \left( y_j y_k - x_j x_k \right) + (y_j-y_k) x + (x_k-x_j) y \right] \, dx \, dy. \]  

(6.5.18)

To simplify the integration assume that \( k \) and \( f \) are constant in each element. Although this assumption is not necessary, the results obtained will generally be sufficiently accurate to warrant it. Next, note that,

\[ \int \int_{e} f \, dx \, dy = \Delta, \]  

(6.5.19)

and

\[ \int \int_{e} \left[ \frac{(x_j y_k - x_k y_j) + (y_j y_k - x_j x_k)}{2 \Delta} \right] \, dx \, dy = \]  

(6.5.20)
where \( \bar{x} = (x_i + x_j + x_k)/3 \) and \( \bar{y} = (y_i + y_j + y_k)/3 \) are the coordinates of the centroid of the triangular element. The right side of (6.5.20) is found to be equal to \( \Delta/3 \) on expansion. Equation (6.5.18) may thus be written,

\[
\frac{\partial J}{\partial u_i} = M_{11} u_i + M_{12} u_j + M_{13} u_k + f \Delta/3 ,
\]

Equation (6.5.21) can be used in (6.5.15) to yield the set of difference equations used in obtaining an approximation to the solution of (6.5.9).

Consider the equation for node 0 in the regular mesh network shown in Figure 6.5.3. The elements containing node 0 are numbered from 1 to 6. Using (6.5.21) with \( i = 0 \), is the central node,

\[
\frac{\partial J}{\partial u_0} = M_{00} u_0 + M_{01} u_1 + M_{02} u_2 + M_{03} u_3 + M_{04} u_4 + M_{05} u_5 + M_{06} u_6 + f \Delta/3 ,
\]

where,
\[
\frac{f_1 A}{3} = \frac{f_1 (kh^2)}{3} = \frac{f_1 h^2}{6}
\]

and

\[
M_o = \frac{k_1}{2h} \left( (y_{SE} - y_S)^2 + (x_{SE} - x_S)^2 \right)
\]
\[
= \frac{k_1}{2} h^2 .
\]

Similarly,

\[
M_s = \frac{k_1}{2h} \left( (y_{SE} - y_O) (y_{SE} - y_S) + (x_{SE} - x_O)(x_{SE} - x_S) \right)
\]
\[
= \frac{k_1}{2} \left\{ (-h*0) + (-h) * (h) \right\} = -\frac{1}{2} h^2
\]

\[
M_{ss} = \frac{k_1}{2h} \left( (y_{SE} - y_S)(y_{SE} - y_S) + (x_{SE} - x_S)(x_{SE} - x_S) \right)
\]
\[
= 0.
\]

Thus, we have for element 1,

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=1} = \frac{k_1}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_S}{2} \right) + \frac{f_1 h^2}{6}
\]

In a similar manner, we find for the other contributing finite elements,

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=2} = \frac{k_2}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_E}{2} \right) + \frac{f_2 h^2}{6}
\]

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=3} = \frac{k_3}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_E}{2} - \frac{h^2 U_N}{2} \right) + \frac{f_3 h^2}{6}
\]

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=4} = \frac{k_4}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_E}{2} - \frac{h^2 U_N}{2} \right) + \frac{f_4 h^2}{6}
\]

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=5} = \frac{k_5}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_N}{2} \right) + \frac{f_5 h^2}{6}
\]

\[
\frac{\partial J}{\partial U_0} \bigg|_{i=6} = \frac{k_6}{2} \left( \frac{h^2 U_0}{2} - \frac{h^2 U_N}{2} - \frac{h^2 U_S}{2} \right) + \frac{f_6 h^2}{6}
\]

and

The difference equation is found by summing the above (i.e. \( \sum \frac{\partial J}{\partial U_0} \)) to obtain,

\[
-u_0 \left( \frac{k_1}{2} + \frac{k_2}{2} + \frac{k_3}{2} + \frac{k_4}{2} + \frac{k_5}{2} + \frac{k_6}{2} \right) + u_0 \left( \frac{k_1}{2} + \frac{k_2}{2} + \frac{k_3}{2} + \frac{k_4}{2} + \frac{k_5}{2} + \frac{k_6}{2} \right) + u_E \left( \frac{k_2}{2} + \frac{k_3}{2} + \frac{k_4}{2} + \frac{k_5}{2} + \frac{k_6}{2} \right) + u_N \left( \frac{k_3}{2} + \frac{k_4}{2} \right)
\]

\[
+ \frac{h^2}{6} \left( f_{1,2} + f_{3,4} + f_{5,6} \right)
\]

\[
= 0 \quad (6.5.22)
\]

For the special case of const \( k=1 \) and \( f(x,y) = 0 \) (6.5.9) is the Laplace
equation and (6.5.22) becomes,

\[-4u_O + u_W + u_N + u_E + u_S = 0\]

which is exactly the same as the five point finite difference approximation derived in Chapter 1.

If the only purpose of the finite element method were to re-derive the finite-difference approximation, it would have lost much of its value. It is important then to emphasise some of the advantages commonly attributed to the finite element method.

1) The material properties in adjacent elements do not have to be the same. This allows the method to be applied to the non-homogeneous and anisotropic regions with relative ease.

2) Irregularly shaped boundaries can be approximated using elements with straight sides or matched exactly using elements with curved boundaries. Thus, the method is not limited to 'nice' shapes with easily defined boundaries.

3) The size of the elements can be varied. This property allows the element grid to be expanded or refined as the need for accuracy is required.

4) The preceding properties can be incorporated into one general purpose computer code, whereas with the finite difference method, except for relatively simple problems and simple geometries, the formulation of the discrete equations must be done by hand, with pencil and paper.

Fletcher (1978) gives a brief description of the Galerkin finite element method and concentrated on aspects that are connected with the traditional Galerkin method. He has shown that the former method also leads to the system of system (6.1.1) but has two advantages over the later formulation:

(i) it leads to a sparse solution matrix A that can be accurately and economically factorised

(ii) by reducing the integration over the whole domain to the numerical
integration over discrete, distortable elements arbitrary boundaries are easily handled.

Traditionally the finite element method has been treated as a variational method. However, although many inviscid flows and a few viscous flows possess a variational formulation the general fluid flow problem governed by the Navier-Stokes equations does not possess a useful variational formulation (Fletcher, 1978). In this section, therefore we will include this problem in our work.

The general applicability of the finite element method makes it a powerful and versatile tool for a wide range of problems. Hence a number of computer program packages have been developed for the solution of a variety of problems. Some of the program can be used for the solution of problems belonging to the different branches of engineering with little or no modification. A summary of these packages can be found in (Rao, 1982) page 33. We will be using the TWODEP package from IMSL that has recently become available at Loughborough University of Technology.
6.6 **TWODEPEP PACKAGE** - A finite element program with automatic user controlled mesh grading (Granville Sewell)

**Introduction**

TWODEPEP is a small, easy to use finite element program which solves a large class of elliptic, parabolic and eigenvalue problems in general two-dimensional regions. It has a preprocessor program which allows the user to supply the problem in a greatly simplified form. The user inputs an initial triangulation with only enough triangles to define the region and with the automatic mesh refinement and grading capabilities in the software package will further minimize the user effort in solving application oriented problems in partial differential equations.

**Problem Definition**

The most general form of the equations solved by the TWODEPEP is:

\[
\begin{align*}
C_1(x,y,u,v,t) \frac{\partial u}{\partial t} &= \frac{2}{\partial x}(OXX) + \frac{2}{\partial y}(OXY) + F_1(x,y,u,v,u,v,t) \\
C_2(x,y,u,v,t) \frac{\partial v}{\partial t} &= \frac{2}{\partial x}(OXY) + \frac{2}{\partial y}(OYY) + F_2(x,y,u,v,u,v,t)
\end{align*}
\]

for \((x,y)\) in the region \(R\), with

\[
\begin{align*}
u &= \text{FB1}(s,t) \\
v &= \text{FB2}(s,t) 
\end{align*}
\]

for \(s\) on part of the boundary \(\partial R_1\)

and,

\[
\begin{align*}
OXX \cdot n + OXY \cdot n &= \text{GB1}(s,u,v,t) \\
OYX \cdot n + OYY \cdot n &= \text{GB2}(s,u,v,t) (\partial R_2), \text{ where } (n_x, n_y) \text{ is the unit outward normal to the boundary.}
\end{align*}
\]

and \(u = U_0(x,y)\)

\(v = V_0(x,y),\) for \(t = T_0\)

where \(OXX, OXY, OYX, OYY\) are functions of \(u, y, u_x, v_y, u_x, v_y, u, v\) and \(t\).

**Method of Solution**

The finite element used is the standard six-node triangle with quadratic basis functions, with one edge curved when adjacent to a curved
boundary, according to the isoparametric method. For the parabolic problem, either the implicit or Crank-Nicolson scheme may be used to discretize the time variable. Optionally, a Richardson extrapolation may be done to double the order of convergence of the time discretization.

In all cases, the algebraic equations are solved by Newton's method. If round-off error appears to be present in the solution of a linear elliptic problem, it can often be diminished by doing an extra iteration. Parabolic and non-linear elliptic problems are less likely to suffer from round-off error, since repeated iteration (one iteration per time step) is done on them. The linear system which must be solved to complete a Newton iteration is solved directly by Gaussian elimination without row interchanges. The Cuthill-McKee algorithm and a special bandwidth reduction algorithm are used to number the nodes and give this linear system a minimum banded structure. Symmetry is also taken advantage of in the elimination process. If the matrix is too large to keep in core, the frontal method is used to organise its storage out of core in which case core storage is only proportional to the number of triangles, and the execution time increases only moderately.

The user inputs an initial triangulation with only enough triangles to define the region, and supplies a function $D\text{3EST}(x,y)$ which guides the refinement of this triangulation. The optimal choice of $D\text{3EST}(x,y)$ is given in Appendix B of the TWODEPEP manual.

Each time a triangle is partitioned, it is divided by a line from the midpoint of its longest side to the opposite vertex. If this side is not on the boundary, the triangle which shares that side must also be divided to avoid non-conforming elements with discontinuous basis functions.

The Preprocessor

TWODEPEP has a preprocessor program which reads the user input describing
the problem in a format designed to minimise the user effort. Little knowledge of Fortran is required to define a problem in this format. Printer plots of the initial triangulation (for error checking) and of the centres of the triangles in the final triangulation are also made.

The detailed description of TWODEPPEP mathematical background is discussed in Appendix B2 of TWODEPPEP manual.

Problem Solved by TWODEPPEP Package

Problem 4

We consider the Laplace equation,

\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \quad 0 \leq x \leq a, \quad 0 \leq y \leq b
\]  

(6.6.1)

with the boundary conditions,

\[
\begin{align*}
  u(x,y) &= 0 \text{ on } x=0 \text{ and } x=a \\
  u(x,y) &= 0 \text{ on } y=0 \\
  u(x,y) &= f(x,y) \text{ on } y=b
\end{align*}
\]

(6.6.1a)

where \( f(x) = \frac{\partial u_0(x)}{\partial x} \), \( u_0 = \text{constant} \).

For our work we take \( u_0, a \) and \( b \) to be 5, 1 and 1 respectively. The initial triangulation is eight and is given by Figure 6.6.1,

\[
\begin{array}{c}
\text{(0,0)} \\
\text{(0,1)} \\
\text{(1,0)} \\
\text{(1,1)} \\
\end{array}
\]

![Figure 6.6.1](image)

and 200 triangles are requested for the final triangulation. The input
NEQ= 1 NTF= 8 NTF= 9 NTF= 200 NDIM= 2

****
**** PROBLEM 4 : DELTA(U)=0
**** 0<X<a 0<Y<b
****
OXX = UX
OXX/UX = 1.0
OXY = UY
OXY/UY = 1.0

****
RHS OF EQUATION 1
****
G1 = 0.0
****
SYMMETRY = 2
ARC=-5
F1 = 4*5*X*(1.0-X)
ARC=-6
F1 = 4*5*X*(1.0-X)

****
PLT1 = 1
****
XA = 0.0
IX = 0.1
IX = 10
YA = 0.0
HY = 0.1
HY = 10

****
VXY = 0.,0. 0.5,0. 1.0,0. 1.,0.5 1.,1. 0.5,1. 0.,1.
VXY = 0.,0.5 0.5,0.5
INDC = 1,2,9 2,3,9 3,4,9 4,5,9 5,6,9 6,7,9 7,8,9 8,1,9
I = -1 -2 -3 -4 -5 -6 -7 -8
END.

FIGURE 6.6.2
The result obtained is compared with the exact number given by,

\[ u(x,y) = \frac{16U}{\pi^3} \sum_{n=1}^{\infty} \frac{(1-(-1)^n)}{n^3 \sinh(\frac{n\pi x}{a}) \sin(\frac{n\pi x}{a})} \]  

and is given in Table 6.6.1 where the summation is over 30 terms (n=30).

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>finite element</th>
<th>exact answer</th>
<th>errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>0.04409</td>
<td>0.04412</td>
<td>-0.00003</td>
</tr>
<tr>
<td>0.1</td>
<td>0.4</td>
<td>0.22438</td>
<td>0.22347</td>
<td>+0.00091</td>
</tr>
<tr>
<td>0.1</td>
<td>0.7</td>
<td>0.62370</td>
<td>0.62440</td>
<td>0.00070</td>
</tr>
<tr>
<td>0.1</td>
<td>1.0</td>
<td>1.80000</td>
<td>1.80028</td>
<td>-0.00028</td>
</tr>
<tr>
<td>0.4</td>
<td>0.1</td>
<td>0.13565</td>
<td>0.13569</td>
<td>-0.00004</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>0.68570</td>
<td>0.68569</td>
<td>+0.00001</td>
</tr>
<tr>
<td>0.4</td>
<td>0.7</td>
<td>1.88810</td>
<td>1.88571</td>
<td>0.00239</td>
</tr>
<tr>
<td>0.4</td>
<td>1.0</td>
<td>4.80000</td>
<td>4.79990</td>
<td>0.00001</td>
</tr>
<tr>
<td>0.7</td>
<td>0.1</td>
<td>0.11567</td>
<td>0.11545</td>
<td>0.00022</td>
</tr>
<tr>
<td>0.7</td>
<td>0.4</td>
<td>0.58347</td>
<td>0.58382</td>
<td>-0.00035</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>1.61200</td>
<td>1.61286</td>
<td>-0.00086</td>
</tr>
<tr>
<td>0.7</td>
<td>1.0</td>
<td>4.20000</td>
<td>4.20012</td>
<td>-0.00012</td>
</tr>
</tbody>
</table>

TABLE 6.6.1

It can be seen that the maximum error in the solution \( u \) was about 0.24%.

Problem 5

We now solve problem 3 using the TWODEPEP finite element package.

The initial triangulation used is shown in Figure 6.6.3 and the 200 triangles are requested for the final triangulation. The input format for this problem is given in Figure 6.6.4. Finally, the result obtained is compared with the exact answer and with the solution from the Rayleigh-Ritz method and is
tabulated in Table 6.6.2. A more accurate solution could be achieved if the final triangulation requested is increased.

\[ u = \cos(\pi y/2) \]

\[ u = e^{-\pi/2 \cos(\pi y/2)} \]

**FIGURE 6.6.3**

<table>
<thead>
<tr>
<th>y location (x=0.5)</th>
<th>Rayleigh-Ritz (n=10) Solution</th>
<th>Finite Element Solution</th>
<th>Exact Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.45594</td>
<td>0.45594</td>
<td>0.45594</td>
</tr>
<tr>
<td>0.2</td>
<td>0.43363</td>
<td>0.43369</td>
<td>0.43362</td>
</tr>
<tr>
<td>0.4</td>
<td>0.36887</td>
<td>0.36876</td>
<td>0.36886</td>
</tr>
<tr>
<td>0.6</td>
<td>0.26800</td>
<td>0.26812</td>
<td>0.26799</td>
</tr>
<tr>
<td>0.8</td>
<td>0.14089</td>
<td>0.14075</td>
<td>0.14089</td>
</tr>
</tbody>
</table>

**TABLE 6.6.2:** Comparison of the Solutions to Problem 5

**Problem 6**

TWODEPEP can solve many fluid mechanics problems. As an example, we consider a basic two-dimensional, steady-state, viscous, incompressible flow problem taken from (Greenspan, 1974). This problem can be formulated as follows. Let the points (0,0), (1,0), (1,1) and (0,1) be denoted by A, B, C and D respectively (see Figure 6.6.5). Let S be the square whose vertices are A, B, C, D and denote its interior by R. On R the equations of motion to be satisfied are the two-dimensional, steady-state, Navier-Stokes equations, that is,
PROBLEM

OXX
OXX/UX
OXY
OXY/UY

RHS OF EQUATION 1

F1

ARC=-2
FB1
ARC=-4
FB1

PLOT

XH
NX
YH
NY

VXY
bc
l
END.

EXP(-3.1415927/2) * COS(3.1415927*Y/2)
COS(3.1415927*Y/2)

FIGURE 6.6.4
\[ \nabla^2 U = -V, \]  
(6.6.3)

\[ \nabla^2 V + R \left( \frac{\partial U}{\partial x} \frac{\partial V}{\partial y} - \frac{\partial U}{\partial y} \frac{\partial V}{\partial x} \right) = 0, \quad R \geq 0, \]  
(6.6.4)

where \( U \) is the stream function, \( V \) is the vorticity and \( R \) is a non-negative constant called the Reynolds number. On \( S \) the boundary conditions to be satisfied are:

- \( U = 0, \frac{\partial U}{\partial x} = 0 \), on \( AD \)  
(6.6.5)

- \( U = 0, \frac{\partial U}{\partial y} = 0 \), on \( AB \)  
(6.6.6)

- \( U = 0, \frac{\partial U}{\partial x} = 0 \), on \( BC \)  
(6.6.7)

- \( U = 0, \frac{\partial U}{\partial y} = -1 \), on \( CD \)  
(6.6.8)

The analytical problem is defined on \( R+S \) by (6.6.3)-(6.6.8) and is shown diagrammatically in Figure 6.6.5. Physically, one can think of a fluid contained between walls \( DA, AB \) and \( BC \), while a force is applied on the surface of the fluid in the direction from \( C \) to \( D \).

The initial triangulation used for this problem is given by Figure 6.6.6. We solve for \( R=50 \) and we give the input format in Figure 6.6.7. Along the line \( y=1 \), for example, we have \( U=0 \) and \( U_y=-1 \), hence the boundary conditions may be formulated there as \( GB1=-1 \) and \( GB2=\beta U \), where \( \beta \) is a large
number so that the second condition is almost equivalent to $U=0$. The
results are tabulated in Table 6.6.3 and the contour is given in Figure 6.6.8.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure666}
\caption{Figure 6.6.6}
\end{figure}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|}
\hline
$x$ & $y$ & $U$ & $V$ \\
\hline
0.0 & 0.0 & -0.62988E-20 & -0.95495E-02 \\
0.3 & 0.3 & 0.16235E-01 & 0.10807E+00 \\
0.6 & 0.3 & 0.21255E-01 & 0.11792E-01 \\
0.9 & 0.3 & 0.26059E-02 & 0.45884E+00 \\
0.3 & 0.6 & 0.58628E-01 & 0.86098E+00 \\
0.6 & 0.6 & 0.73360E-01 & -0.13487E+01 \\
0.9 & 0.6 & 0.86343E-02 & 0.15475E+01 \\
0.3 & 0.9 & 0.65287E-01 & -0.58560E+01 \\
0.6 & 0.9 & 0.70753E-01 & -0.51721E+01 \\
0.9 & 0.9 & 0.23844E-01 & -0.16845E+01 \\
1.0 & 1.0 & 0.26603E-07 & -0.40332E+02 \\
\hline
\end{tabular}
\caption{Table 6.6.3}
\end{table}
NBQ= 2 NTF= 100 NDIM= 35000

****

FLUID PROBLEM

****

NAVIER STOKES EQUATION

****

OXX = UX
OXY = UY
OYX = VX
OYY = VY

****

RHS OF EQUATION 1 AND 2

****

F1 = -V
F2 = -50*(UX*VY-UY*VX)

****

GB1 IS THE BOUNDARY CONDITION OF UX

****

GB2 IS THE BOUNDARY CONDITION OF U

****

GB2=BETA*U, WHERE BETA IS LARGE TO FORCE U TO BE 0

****

ARC=1001
GB1 = 0.
GB2 = -1.0D20*U

ARC=1002
GB1 = 0.
GB2 = -1.0D20*U

ARC=1003
GB1 = -1.
GB2 = -1.0D20*U

ARC=1004
GB1 = 0.
GB2 = -1.0D20*U

****

PLOT = 1

****

XA = 0.0
XB = 0.1
NX = 10
YA = 0.0
HY = 0.1
NY = 10

****

VXX = 0.0,0.0 1.0,0.0 1.0,1.0 0.0,1.0 0.5,0.5
IABC = 1,2,5 2,3,5 3,4,5 4,1,5
I = 1001 1002 1003 1004

END.

FIG. 6.6.7
FIG. 6.6.8: Values of Stream Function \( U \)
Problem 6
CHAPTER 7

APPROXIMATE PRECONDITIONING OF THE MODIFIED

CONJUGATE GRADIENT METHOD
In Chapter 3, the method of conjugate gradient (C.G.) for solving the system of linear algebraic equations,
\[ Au = b, \]  
(7.1.1)
was introduced, where A is nxn symmetric and positive definite. One of the important properties of this method is that in the absence of round-off errors the solution is obtained in at most n iteration steps. Apart from the primary advantages listed in Chapter 2, the method has a number of attractive properties when regarded not as a direct method for the solution of dense systems of linear equations but as an iterative method for the solution of large sparse systems. These are:

(i) It does not require an estimation of acceleration parameters
(ii) It takes advantage of the distribution of the eigenvalues of the iteration operator [Concus, et al, 1976]
(iii) It is easy to program and
(iv) It is relatively efficient [Lipitakis, 1978]

The C.G. method is especially useful for large sparse systems (7.1.1) arising from the discretisation of a boundary value problem for elliptic partial differential equations. Although it is known to converge in most cases within much less than n iterations, the convergence is not rapid enough to compare with other methods, such as SSOR method. [Nodera et al, 1981]. To improve the convergence rates, Evans (1968), (1973) has successfully applied the preconditioning technique with the C.G. algorithm. Since then, a number of authors have described the preconditioning C.G. for solving (7.1.1) (c.f. Axelsson, 1974; Concus et al, 1976; Meijerink and Van der Vorst, 1976; Wong, 1979).

We will now follow the notation of Hestenes and Stiefel (1980) to rewrite the C.G. algorithm, previously given in Section 3.4.
Algorithm 7.A

Initial step Select a point $u_1$ and compute

$$p_1 = r_1 = b - Au_1$$  \hfill (7.A.1)

Iterative steps Having obtained $u_k$, $r_k$, and $p_k$ compute $u_{k+1}$, $r_{k+1}$ and $p_{k+1}$ by the formulae

$$a_k = \frac{c_k}{d_k}, \quad d_k = (p_k, Ap_k), \quad c_k = (p_k, r_k) \quad \text{or} \quad c_k = (r_k, r_k)$$  \hfill (7.A.2)

$$u_{k+1} = u_k + a_k p_k, \quad r_{k+1} = r_k - a_k Ar_k$$  \hfill (7.A.3)

$$b_k = -(p_k, Ar_{k+1})/d_k \quad \text{or} \quad b_k = \frac{(r_{k+1}, r_{k+1})}{c_k}$$  \hfill (7.A.4)

$$p_{k+1} = r_{k+1} + b_k p_k.$$  \hfill (7.A.5)

Termination Terminate the $k$th step if $|u^{(k+1)} - u^{(k)}| < \varepsilon$, where $\varepsilon$, small, is a predetermined tolerance.

It can be seen that the matrix $A$ need not be stored as an array in memory; at each stage of the iteration procedure it is necessary to compute only the product of $Ar$ for a given vector $r$. The sparsity of the matrix can then be effectively used and only the non-zero coefficients are used in the algorithms above and thus the computer storage usage is minimal, requiring only 4 vectors $u_k, r_k, p_k$ and working space for temporarily storing $Ap_k$ and $Ar_{k+1}$. It may seem that here we require an extra working space for $Ar_{k+1}$, but it is not so as we can re-use the same location for storing $Ap_k$ since the vector $Ar_{k+1}$ has to be evaluated after the whole vector $Ap_k$ has been evaluated and used, and vice versa.

As has been pointed out in Chapter 4 a faster rate of convergence of the C.G. can be achieved if the matrix $A$ has clustered eigenvalues which will be the case if $A$ does not differ very much from the identity matrix. Unfortunately, for matrices arising from the finite difference/element equation of elliptic/parabolic boundary value problems the eigenvalues are
usually uniformly spread between the lower and upper extremes. Thus, for these problems, the C.G. in its simplest form (eqs. 7.A.1-7.A.5) performs very poorly [Kershaw, 1978].

To treat such matrices by the C.G., it is necessary to transform first the original set of equations into a new system

\[
P \gamma = c ,
\]

(7.1.2)

where \( P = x^{-1} A x^{-1} \), \( \gamma = X u \) and \( c = x^{-1} b \), \( x^{-1} \) being an arbitrary symmetric matrix which aims to produce a matrix \( P \) with most of its eigenvalues clustered around 1. Note that \( P \) is symmetric and positive definite.

Apply (7.1.2) to solve (7.1.1), we arrive at the following modified C.G. (MCG) defined by the following relations involving a prescribed positive definite symmetric matrix \( H \), where \( H = x^{-1} \).

Algorithm 7.B - MCG Algorithm

**Initial step** Select a positive definite symmetric matrix \( H \) and an initial point \( \bar{u}_{1} \). Compute

\[
\bar{f}_1 = b - A \bar{u}_1 , \quad \bar{E}_1 = H \bar{r}_1
\]

(7.B.1)

**Iterative steps** Having obtained \( u_k, r_k \) and \( P_k \), compute

\[
s_k = R_{k-1} d_k , \quad d_k = (s_k, P_k), \quad c_k = (P_k, r_k), \quad a_k = c_k/d_k
\]

(7.B.2)

\[
u_{k+1} = u_k + a_k P_k , \quad r_{k+1} = r_k - a_k s_k
\]

(7.B.3)

\[
P_{k+1} = H_{k-1} \frac{b_k}{c_k} P_k , \quad b_k = - (s_k, H_{k-1} r_k)/d_k
\]

(7.B.4)

**Termination** Again, we terminate at the \( k^\text{th} \) step if \( |u^{(k+1)} - u^{(k)}| < \epsilon \).

It can be seen that if \( H^{-1} = I \), (7.B) turns into (7.A). It may be shown that the direction \( \tilde{e}_k = X e_k \) are conjugate with respect to \( P \), i.e.,

\[
\tilde{e}_i P \tilde{e}_k = 0 , \text{ for any } i \neq k
\]

(7.1.3)

and the residuals

\[
r_{k} = c - P \bar{e}_k = X^{-1} r_k
\]

(7.1.4)

are mutually orthogonal.
We have earlier deduced that if \( AH = I \), the eigenvalues of \( AH \), which are the same as those of \( P \) will be clustered in the vicinity of 1, and the gradient method will converge faster. Thus, the success of the method will depend on two requirements:

1) \( AH = I \), or \( H \) must approximate \( A^{-1} \) in some way or other

2) The matrix vector multiplication \( Hr_{x+1} \) must be easily evaluated.

Such a matrix \( H \) is not easy to find in the case of sparse matrices obtained as finite difference approximations of partial differential equations. This is because in such cases the matrix \( H \) is required to have a few very large eigenvalues [Nodera, T. and Takahasi, 1981]. Computational considerations concerning the choice of \( H \) will restrict us to the form of inverse triangular components of a sparse matrix. This form will permit simple forward and back substitution processes which are easily facilitated on a computer. Furthermore, the sparsity of \( A \) will be preserved. [Evans, 1974].

Gambolati (1980) has shown that if \( A \) is diagonally dominant a good approximation \( H \) for \( A^{-1} \) is provided by the incomplete Cholesky decomposition of \( A \) (Meijerink and Van der Vorst, 1974) which gives \( H \) as

\[
H = (PP^T)^{-1}
\]

where \( P \) is lower triangular with the same sparsity as \( A \). To satisfy both requirements above, we will choose \( P \) to be \((I-\alpha L)\), where \( \alpha \) is an adjustable parameter \((0<\alpha<2)\) which is equivalent to the acceleration parameter in the preconditioning theory, previously described in the earlier chapters and \( L \) is to be defined later.

To further accelerate the convergence rates of MCG, we use the preconditioning theory given in the earlier chapters. We first write \( A \) as

\[
A = L'^{-1}D-L'^T \tag{7.1.5}
\]

where \( L' \) is strictly the lower triangular part of the matrix \( A \) and \( D \) is
diagonal part. Pre-multiplying and post-multiplying (7.1.5) by \(D^{-\frac{1}{2}}\), we have,
\[
D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = I-I-L^T.
\] (7.1.6)

Thus, using the preconditioning theory, we can rewrite the system (7.1.1) as,
\[
(I-\omega L)^{-1}A(I-\omega L^T)^{-1}((I-\omega L^T)u) = (I-\omega L)^{-1}b,
\] (7.1.7)
or
\[
Bz = b,
\] (7.1.8)
where
\[
B_{\omega} = (I-\omega L)^{-1}A(I-\omega L^T)^{-1},
\] (7.1.9)
\[
= S^{-1}A(T)^{-1}.
\]
\[
z = (I-\omega L)^T \gamma = S_T \gamma,
\] (7.1.10)
and
\[
\frac{b}{\omega} = (I-\omega L)^{-1}b = (S_T)^{-1}b.
\] (7.1.11)

The matrices \(B_{\omega}\) in (7.1.9) are usually generated from a simple computational molecule rather than stored directly in the computer memory.

For example, the approximation of the two-dimensional Laplace model problem by finite difference method (2.2.6), i.e.,
\[
4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = 0,
\] (7.1.12)
can be depicted as in Figure 7.1.1. The individual terms of equation (7.1.12) which constitute the \((I-\omega L)\) and \((I-\omega L^T)\) matrices can be seen in finite difference form to be,
\[
-\omega \frac{u_{i+1,j}}{4}, -\omega \frac{u_{i,j-1}}{4}, u_{i,j}, -\omega \frac{u_{i+1,j}}{4}, u_{i,j},
\] (7.1.13)
and
\[
\frac{u_{i,j}}{4} - \omega \frac{u_{i+1,j}}{4}, -\omega \frac{u_{i,j-1}}{4}, u_{i,j} - \omega \frac{u_{i+1,j}}{4},
\] (7.1.14)
respectively.

\[\text{FIGURE 7.1.1: Computational molecules of the Laplace operator}\]
Thus, the generation of the vector $B\gamma$ on the net can be considered as the solution of $\gamma = (S^T)u$ which is a back substitution process involving the operator as given in Figure 7.1.1(a); followed by the usual Laplace operator $\gamma = Ay$ as given in Figure 7.1.1(b) with a final solution vector of $u = S^{-1}\gamma$ to the existing vector on the net, which is a forward substitution process as given by the operator in Figure 7.1.1(c) [Evans, 1967].

In this Chapter, we applied two types of preconditioning to the MPCG. The first type uses preconditioning molecules derived from the sparse structure of the coefficient matrix $A$ as in $P, P^T, S$ and $S^T$ (Figure 7.1.1 and 7.1.2). Thus, to generate the matrix $H$ in the Algorithm 7.B, where,

$$H = (P^T)^{-1}\text{~}$$

for the two-dimensional Biharmonic model problem with a 13-point finite-difference formula (cf. equation (2.2.48)) the approximation of $P$ and $P^T$ matrices can be seen in the finite difference forms to be

$$0.1au_{i-1,j+1} + 0.1au_{i-1,j-1} - 0.05au_{i-2,j} - 0.05au_{i,j-2} - 0.4au_{i-1,j} - 0.4au_{i,j-1} + u_{i,j} = 0$$  \hspace{1cm} (7.1.16)

and

$$0.1au_{i-1,j+1} + 0.1au_{i+1,j-1} - 0.05au_{i+2,j} - 0.05au_{i,j+2} - 0.4au_{i+1,j} - 0.4au_{i,j+1} + u_{i,j} = 0$$  \hspace{1cm} (7.1.17)

respectively and they are depicted in Figure 7.1.2.
Thus, the generation of vector $H_r$ on the net can be considered as the solution of $s = P^{-1}x$ which is a forward substitution process involving the operator as given in Figure 7.1.2(b) followed by the vector $p = (P^T)^{-1}s$ which is a back substitution process as given by the operator in Figure 7.1.2(a).

It can be seen that on a computer with multiplication times large compared to the addition times the application of the preconditioning to the basic iteration methods involves (at best) a two-fold increase in extra arithmetical work (i.e. traversing the ordered points on the network 3 times instead of once) if the product terms $\omega L$, involving the coupling coefficients are stored prior to the commencement of the iteration and used in the manner indicated. Thus, for the complex molecules such as two-dimensional Biharmonic or three-dimensional Laplace problems, the extra computational time is considerable.

To reduce the amount of work involved in the application of preconditioning we will attempt to simplify the computational molecules of the matrices $P$, $P^T$, $S$ and $S^T$. With a proper choice of such simplified molecules, it is hoped that (at least) a similar range of condition numbers could be achieved. We denote this type of preconditioning as an approximate preconditioning strategy [Nodera & Takahasi, 1981]. The details will be given as we proceed to solve the following problems involving the solution of very large order sparse matrices derived from the finite difference solution of partial differential equations. The problems are:

I. Two dimensional Laplace problem with a 9 point finite difference formula

II. Two dimensional Biharmonic problem with a 13 point finite difference formula

III. Three dimensional Laplace problem with a 7 point finite difference formula
7.2 COMPUTATIONAL APPROACH

Consider the application of preconditioning to the iterative process of the MPCG algorithm (7.8.1) with regard to the solution (7.1.1). For this iterative method, we wish to determine the preconditioned matrix $B_\omega$ or the vector $B_\omega z$ in (7.8.1) in addition to the optimal choice of the matrix $H$ or the vector $Hr$ in (7.8.1).

The matrix $A$ is not stored in a compact form in the computer memory, rather it is generated. We have previously shown that the vector $B_\omega z$ can then be obtained from the application of three simple computational molecules applied on the given grid of points which was discussed earlier in Chapter 2. We give the details below, for the 13 point finite difference formula for the two-dimensional Biharmonic problem. The computational molecules are depicted in Figure 7.2.1.

![Diagram](image.png)

(a) $(I-\omega L)^{-1}$
(b) $(I-L-U)^{-1}$
(c) $(I-\omega L^T)^{-1}$

FIGURE 7.2.1
We require to generate the vector
\[ \mathbf{y} = (I - \omega L)^{-1} A (I - \omega U)^{-1} \mathbf{z} \] (7.2.1)

which we will carry out using the auxiliary vectors \( \mathbf{x} \), \( \mathbf{y} \) and \( \mathbf{v} \). The algorithm for carrying out these operations is:

1. First compute
\[ \mathbf{z} = (I - \omega L^T) \mathbf{x} , \] (7.2.2)

or
\[ x_{i,j} = z_{i,j} + \frac{2}{5} \omega (x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}) - \frac{\omega}{10} (x_{i+1,j+1} + x_{i-1,j-1}) \]

or
\[ \mathbf{w} = \mathbf{y} + \frac{2}{5} \omega (\mathbf{y} + \mathbf{z}) - \frac{\omega}{10} (\mathbf{y} + \mathbf{z}) \]

for \( j,i = n(-1)1 \).

i.e., the vector \( \mathbf{x} \) is obtained by a back substitution process using the computational molecule in Figure 7.2.1(a).

2. Next, we have,
\[ \mathbf{y} = \mathbf{A} \mathbf{x} , \] (7.2.3)

or, for \( j,i = n(-1)1 \), we compute,
\[ y_{i,j} = x_{i,j} - \frac{2}{5} (x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}) \]

\[ + \frac{1}{10} (x_{i+1,j+1} + x_{i,j+1} + x_{i,j+1} + x_{i,j-1}) \]

\[ + \frac{1}{20} (x_{i+2,j} + x_{i-1,j+2} + x_{i+1,j} + x_{i,j-1} + x_{i,j+1}) \]

i.e., the vector \( \mathbf{y} \) is obtained using the computational molecule 7.2.1(b).

3. Next, we have,
\[ \mathbf{v} = (I - \omega U) \mathbf{y} , \] (7.2.4)

or for \( j,i = n(-1)1 \),
\[ v_{i,j} = y_{i,j} + \frac{2}{5} \omega (v_{i-1,j} + v_{i+1,j}) - \frac{\omega}{10} (v_{i-1,j+1} + v_{i,j-1}) \]

\[ - \frac{\omega}{20} (v_{i-2,j} + v_{i+1,j}) \]

which is a forward substitution process using the computational molecule as depicted in Figure 7.2.1(c).
The sparseness of the original matrix has clearly been retained in the operators \((I-\omega L)^{-1}\) and \((I-\omega L^T)^{-1}\), which are the original sparse triangular components of \(A\) and which have been shown to involve forward and back substitution processes requiring only the non-zero elements of \(A\).

For the generation of the vector \(\hat{H}_x\), we give the details as follows.

iv) First, we compute,

\[ s = (I-\omega L)^{-1} r, \]

hence,

\[ r = (I-\omega L)s, \]

or, for \(j,i=n(-1)1\), we compute

\[ s_{i,j} = r_{i,j} + \frac{2}{5}(s_{i,j} + s_{i,j-1}) - \frac{a}{10}(s_{i-1,j+1} + s_{i-1,j-1}) \]

\[ - \frac{a}{20}(s_{i-2,j} + s_{i+2,j}), \]

which is a forward substitution process using the computational molecule as depicted in Figure 7.2.1(c).

v) Next, we compute

\[ p = (I-\omega L^T)^{-1} s. \]

hence

\[ s = (I-\omega L^T)p, \]

or for \(j,i=n(-1)1\), we compute

\[ p_{i,j} = s_{i,j} + \frac{2}{5}(p_{i+1,j} + p_{i,j+1}) - \frac{a}{10}(p_{i+1,j+1} + p_{i-1,j-1}) \]

\[ - \frac{a}{20}(x_{i+2,j} + x_{i,j+2}), \]

which again is a back substitution process using the computational molecule as depicted in Figure 7.2.1(a).

The initial guess for \(z_{i,j}\) is taken to be \(0.2x_{i,j}\), \(j,i=1(n)1\), and the iteration is continued for each variable \(y_k\) until an accuracy of \(0.5\times10^{-6}\) (i.e. \(e\)) is achieved before the transformation,

\[ x = (I-\omega L^T)z, \]

is carried out to give the final solution.

The error was taken to be the ratio of Euclidean Error norm \(||r_k||_2\) to \(||r_0||_2\), where \(r_k\) means the residuals at the \(k^{th}\) iteration. The maximum
and minimum eigenvalues of the coefficient matrix $B_\omega$ were determined by the judicious use of the Power method and the inverse Power method respectively. The algorithm for the Power method was given in Chapter 1. The computations were done on a Prime P400 and on an ICL 1904 machine. In addition, extreme cases of problems requiring large times were computed on an 1906A machine.
7.3 NUMERICAL RESULTS

Biharmonic Problem

We shall first consider in this section the fourth order two-dimensional Biharmonic model problem given in Chapter 2, i.e.,

\[ \frac{\partial^4 U}{\partial x^2 \partial y^2} + 2 \frac{\partial^4 U}{\partial x^2 \partial y^2} + \frac{\partial^4 U}{\partial y^4} = 0 \]  

(7.3.1)

in the bounded region \( R = (0,1) \times (0,1) \) with the boundary condition,

\[ U = s(x,y) \quad \text{for} \quad (x,y) \in C \]  

(7.3.2a)

\[ \frac{\partial U}{\partial n} = g(x,y) \quad \text{for} \quad (x,y) \in C \]  

(7.3.2b)

where \( C \) is its boundary and \( \partial / \partial n \) denotes differentiation in an outward normal direction to \( C \). We use the standard 13-point molecules (as depicted earlier in Figure 7.2.1) over a region \( R \) for mesh sizes \( h^{-1} = 19 \) and 39. Thus this problem leads to a sparse system of order 361 and 1521 equations respectively.

Method A

Table 7.3.1 indicates that when \( \omega = 0 \), i.e. when the matrix \( A \) in the algorithm 7.3 is an identity matrix that is when the iterative method reverts to the PCG method, a considerable reduction in the number of iterations is obtained at the optimum preconditioning parameter \( \omega, \omega_{\text{opt}} \). At this value of \( \omega_{\text{opt}} \), the P-conditioned number of the coefficient matrix \( B \) is minimised as indicated by Figure 7.3.1. In Figure 7.3.2 we compare the behaviour of the residual for the PCG method for different choices of the preconditioning parameter \( \omega \) (for the mesh size \( h^{-1} = 19 \)).

For the MPCG method, i.e. when \( \omega \neq 0 \), a further reduction in the number of iterations is obtained at the optimal choice of the adjustable parameter \( a \). We note from Figure 7.3.2 that when \( a \neq 0 \), the curves become flatter. This indicates that the method MPCG is less vulnerable to the wrong choice of \( \omega \).
Method B

Equation (7.3.1) can be re-written in the form of

\[ \frac{\partial^2}{\partial x^2} \left( \frac{1}{\partial x^2} + \frac{1}{\partial y^2} \right) u = 0, \tag{7.3.3} \]

or i.e. by applying the Laplace operator twice over the region \( R \).

<table>
<thead>
<tr>
<th>( \omega )</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>195</td>
<td>169</td>
<td>109</td>
<td>93</td>
<td>79</td>
<td>69</td>
<td>64</td>
<td>113</td>
</tr>
<tr>
<td>0.4</td>
<td>169</td>
<td>114</td>
<td>88</td>
<td>79</td>
<td>70</td>
<td>64</td>
<td>62</td>
<td>110</td>
</tr>
<tr>
<td>0.8</td>
<td>109</td>
<td>88</td>
<td>74</td>
<td>63</td>
<td>63</td>
<td>61</td>
<td>59</td>
<td>107</td>
</tr>
<tr>
<td>1.0</td>
<td>91</td>
<td>78</td>
<td>67</td>
<td>64</td>
<td>61</td>
<td>59</td>
<td>59</td>
<td>123</td>
</tr>
<tr>
<td>1.2</td>
<td>77</td>
<td>70</td>
<td>63</td>
<td>60</td>
<td>59</td>
<td>61</td>
<td>68</td>
<td>152</td>
</tr>
<tr>
<td>1.4</td>
<td>68</td>
<td>64</td>
<td>60</td>
<td>60</td>
<td>61</td>
<td>65</td>
<td>79</td>
<td>195</td>
</tr>
<tr>
<td>1.6</td>
<td>63</td>
<td>60</td>
<td>60</td>
<td>67</td>
<td>78</td>
<td>78</td>
<td>103</td>
<td>264</td>
</tr>
<tr>
<td>2.0</td>
<td>110</td>
<td>108</td>
<td>113</td>
<td>124</td>
<td>155</td>
<td>196</td>
<td>267</td>
<td>&gt; *</td>
</tr>
</tbody>
</table>

**TABLE 7.3.1:** The number of iterations for different values of the acceleration parameters \( \omega \) and \( \alpha \) — Method A, Problem II

If we use the standard 5-point molecule in an approximation for the Laplace operator and apply it twice over the region \( R \), we obtain a similar approximation to the 13-point finite difference approximation for the Biharmonic operator. Thus, we next choose the computational molecules for the inverse triangular sparse matrices as,

\[ \text{> more than 300 iterations.} \]
We classify this simplified molecule as the approximate preconditioning Bl.

For the purpose of comparisons, we give the behaviour of the residuals.

<table>
<thead>
<tr>
<th>ω</th>
<th>α</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>195</td>
<td>76</td>
<td>48</td>
<td>40</td>
<td>37</td>
<td>75</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td>104</td>
<td>61</td>
<td>44</td>
<td>36</td>
<td>35</td>
<td>71</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>56</td>
<td>44</td>
<td>36</td>
<td>37</td>
<td>45</td>
<td>93</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>43</td>
<td>36</td>
<td>37</td>
<td>45</td>
<td>61</td>
<td>137</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>38</td>
<td>34</td>
<td>35</td>
<td>62</td>
<td>96</td>
<td>220</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>77</td>
<td>73</td>
<td>100</td>
<td>143</td>
<td>227</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>1.6</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 7.3.2:** The number of iterations to converge for different values of acceleration parameters, α and ω - Method Bl, Problem II

The results for these approximate computational molecules are given in Table 7.3.2 and Figures 7.3.3 and 7.3.4. It is apparent from these results that method Bl gives a superior rate of convergence when compared to the method A.
FIG. 7.3.2: METHOD A - PROBLEM II

FIG. 7.3.3: METHOD B1 - PROBLEM II
FIG. 7.3.4a: ITER VS RES - METHOD A

FIG. 7.3.4b: ITER VS RES - METHOD B1
Comparing the molecules for the inverse sparse triangular matrices, we see that the number of nodes in each of the molecules in Figure 7.3.3 is one less than those of the molecules given in Figure 7.2.1. This observation leads us to the approximations of the molecules for the sparse triangular matrices which are given by their computational molecules. The approximations are:

B2. From \((I-\omega L)^T\) and \((I-\omega L)\) of the Laplace operator

\[
\begin{pmatrix}
-1/4\omega \\
1
\end{pmatrix}
\begin{pmatrix}
-1/2\omega \\
1
\end{pmatrix} =
\begin{pmatrix}
1/16\omega^2 \\
-1/8\omega^2
\end{pmatrix}
\]

**FIGURE 7.3.5**

We obtain the following approximation of \((I-\omega L)^T \) and \((I-\omega L)\) to the 13-point Biharmonic equation

\[
\begin{pmatrix}
-1/16\omega^2 \\
1/2\omega \\
-\omega^2/8
\end{pmatrix}
\begin{pmatrix}
1/2\omega \\
1/2\omega \\
-\omega^2/16
\end{pmatrix} =
\begin{pmatrix}
1 \\
1/2\omega \\
-1/16\omega^2
\end{pmatrix}
\]

**FIGURE 7.3.6**
Here we omit one node from the computational molecules in Figure 7.1.2. The modified molecule for this method is given in Figure 7.3.8.

FIGURE 7.3.7

The following strategies B4 and B5 are derived from the preceding methods and their computational molecules are depicted in Figure 7.3.9 and Figure 7.3.10.

FIGURE 7.3B
FIGURE 7.3.9

We give a similar table for the method B5 (Table 7.3.3) and summarized the results in Tables 7.3.4 and Figure 7.3.11.

<table>
<thead>
<tr>
<th>α</th>
<th>ω</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>195</td>
<td>104</td>
<td>52</td>
<td>45</td>
<td>67</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>118</td>
<td>71</td>
<td>49</td>
<td>42</td>
<td>40</td>
<td>52</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>68</td>
<td>49</td>
<td>40</td>
<td>37</td>
<td>36</td>
<td>53</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>52</td>
<td>43</td>
<td>37</td>
<td>34</td>
<td>40</td>
<td>56</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>1.2</td>
<td>1.2</td>
<td>45</td>
<td>39</td>
<td>36</td>
<td>40</td>
<td>47</td>
<td>82</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>1.4</td>
<td>1.4</td>
<td>152</td>
<td>50</td>
<td>49</td>
<td>54</td>
<td>79</td>
<td>136</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>1.6</td>
<td>1.6</td>
<td>165</td>
<td>154</td>
<td>151</td>
<td>159</td>
<td>200</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
</tbody>
</table>

**TABLE 7.3.3**: The number of iterations against the value of acceleration parameters α and ω - Method B5, Problem II
The results indicate that when the preconditioning strategy is used with the modified molecules (i.e. method B), a significant increase in the rate of convergence occurred. It is also apparent that when the approximations B4 and B5 are used the MPCG method has shown to have a rapid convergence. This is indicated by the reduction in the number of iterations when the optimum \( \alpha \) and the optimum \( \omega \) is chosen (see Table 7.3.4). For the approximations B1 and B2, we can see that it is not worth the extra work in generating the matrix \( H \) (Algorithm 7.8) when preconditioning CG (algorithm 7.1A) give nearly the same results. However it is interesting to note that to obtain a similar result, when MPCG with the matrix \( H \) chosen as suggested is applied we do not have to precondition the coefficient matrix \( A \) (i.e. when \( \omega \neq 0 \)).
<table>
<thead>
<tr>
<th>( \omega )</th>
<th>Minimum eigenvalue</th>
<th>Maximum Eigenvalue</th>
<th>P-Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.655E-03</td>
<td>3.16</td>
<td>4830</td>
</tr>
<tr>
<td>0.4</td>
<td>0.726E-03</td>
<td>1.54</td>
<td>2120</td>
</tr>
<tr>
<td>0.8</td>
<td>0.104E-02</td>
<td>1.04</td>
<td>1000</td>
</tr>
<tr>
<td>1.0</td>
<td>0.141E-02</td>
<td>1.00</td>
<td>710</td>
</tr>
<tr>
<td>1.2</td>
<td>0.210E-02</td>
<td>1.04</td>
<td>496</td>
</tr>
<tr>
<td>1.4</td>
<td>0.355E-02</td>
<td>1.19</td>
<td>355</td>
</tr>
<tr>
<td>1.6</td>
<td>0.712E-02</td>
<td>1.56</td>
<td>219</td>
</tr>
<tr>
<td>1.8</td>
<td>0.172E-01</td>
<td>2.78</td>
<td>161</td>
</tr>
<tr>
<td>1.9</td>
<td>0.249E-01</td>
<td>5.12</td>
<td>206</td>
</tr>
</tbody>
</table>

**TABLE 7.3.5**

<table>
<thead>
<tr>
<th>1/h</th>
<th>P Condition Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \omega = 0 )</td>
</tr>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>19</td>
<td>4830</td>
</tr>
<tr>
<td>39</td>
<td>7830</td>
</tr>
</tbody>
</table>

**TABLE 7.3.6**: P condition numbers of the coefficient matrix \( B \) for different methods - Problem 2

\( \omega_{\text{opt}} \) as given in Table 7.3.4.
FIG. 7.3.10: ITER VS RES METHOD B4

FIG. 7.3.11: OPTIMUM ACC. PARAMETERS
FIG. 7.3.12: P-COND VS \( w \) - PROBLEM
7.4 LAPLACE PROBLEMS

We next solve the two and three-dimensional Laplace model problems given earlier in Chapter one, i.e. Problem I and Problem III.

Problem I was solved for a mesh size \( h = 1/20 \) and \( h = 1/40 \) over the region involving the unit square with prescribed boundary values and Problem III for mesh sizes \( h = 1/10 \) and \( h = 1/20 \) within a unit cube with specified boundary values. Problem I leads to a sparse system of order 361 and 1521 equations and problem III to a similar system of 729 equations and 6859 equations.

Problem I

The computational molecules to generate the preconditioned matrix \( B \) are given in Figure 7.4.1.

\[
\begin{align*}
(I - \omega L)^{-1} & \quad A = (I-L-U) \\
(I - \omega L)^{-1} & \quad (I - \omega T)^{-1}
\end{align*}
\]

FIGURE 7.4.1
Method B

From Figure 7.4.1a and 7.4.1c we now have the modified computational molecules as depicted in Figure 7.4.2 and Figure 7.4.3 for the methods B6 and B7 respectively.

**B6**

\[
(I - \omega L)^{-1}
\]

**B7**

\[
(I - \omega U)^{-1}
\]

**FIGURE 7.4.2: Method B6**

**FIGURE 7.4.3: Method B7**
Their results are given in Tables 7.4.1, 7.4.2 and Figures 7.4.4 and 7.4.5.

<table>
<thead>
<tr>
<th>α</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>51</td>
<td>42</td>
<td>31</td>
<td>26</td>
<td>22</td>
<td>19</td>
<td>17</td>
<td>28</td>
</tr>
<tr>
<td>0.4</td>
<td>41</td>
<td>31</td>
<td>24</td>
<td>22</td>
<td>19</td>
<td>17</td>
<td>17</td>
<td>33</td>
</tr>
<tr>
<td>0.8</td>
<td>31</td>
<td>25</td>
<td>21</td>
<td>19</td>
<td>17</td>
<td>18</td>
<td>22</td>
<td>41</td>
</tr>
<tr>
<td>1.0</td>
<td>26</td>
<td>22</td>
<td>19</td>
<td>18</td>
<td>19</td>
<td>21</td>
<td>27</td>
<td>46</td>
</tr>
<tr>
<td>1.2</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>19</td>
<td>21</td>
<td>25</td>
<td>31</td>
<td>55</td>
</tr>
<tr>
<td>1.4</td>
<td>19</td>
<td>18</td>
<td>19</td>
<td>21</td>
<td>25</td>
<td>30</td>
<td>37</td>
<td>64</td>
</tr>
<tr>
<td>1.6</td>
<td>17</td>
<td>17</td>
<td>18</td>
<td>26</td>
<td>31</td>
<td>37</td>
<td>46</td>
<td>69</td>
</tr>
<tr>
<td>2.0</td>
<td>27</td>
<td>33</td>
<td>22</td>
<td>48</td>
<td>60</td>
<td>64</td>
<td>79</td>
<td>&gt;</td>
</tr>
</tbody>
</table>

**TABLE 7.4.1:** The number of iterations against the value of acceleration parameters $\omega$ and $\alpha$ - Method A, Problem I

> more than 100 iterations

<table>
<thead>
<tr>
<th>α</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>51</td>
<td>38</td>
<td>29</td>
<td>26</td>
<td>23</td>
<td>21</td>
<td>20</td>
<td>22</td>
</tr>
<tr>
<td>0.4</td>
<td>38</td>
<td>30</td>
<td>24</td>
<td>22</td>
<td>21</td>
<td>20</td>
<td>19</td>
<td>24</td>
</tr>
<tr>
<td>0.8</td>
<td>29</td>
<td>24</td>
<td>21</td>
<td>20</td>
<td>19</td>
<td>19</td>
<td>21</td>
<td>29</td>
</tr>
<tr>
<td>1.0</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>19</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>35</td>
</tr>
<tr>
<td>1.2</td>
<td>23</td>
<td>21</td>
<td>19</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>27</td>
<td>41</td>
</tr>
<tr>
<td>1.4</td>
<td>21</td>
<td>20</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>26</td>
<td>31</td>
<td>45</td>
</tr>
<tr>
<td>1.6</td>
<td>20</td>
<td>19</td>
<td>21</td>
<td>24</td>
<td>27</td>
<td>31</td>
<td>36</td>
<td>53</td>
</tr>
<tr>
<td>2.0</td>
<td>21</td>
<td>24</td>
<td>29</td>
<td>35</td>
<td>40</td>
<td>47</td>
<td>57</td>
<td>87</td>
</tr>
</tbody>
</table>

**TABLE 7.4.2:** The number of iterations against the value of acceleration parameters $\omega$ and $\alpha$ - Method B5, Problem I
FIG. 7.4.4: PROBLEM I, METHOD A AND B6

FIG. 7.4.5: OPTIMM ACC. PARAMETERS
Problem III

The computational molecules for this problem are given in Figure 7.4.7.

For the Method B, we omit one node in the ζ-plane and the computational molecules for the sparse triangular matrices \((I-\omega L)^{-1}\) and \((I-\omega L_T)^{-1}\) are then shown as in Figure 7.4.8.
The results are shown in Tables 7.4.4, 7.4.5 and Figure 7.4.6.

![Diagram](image)

### Table 7.4.4: The number of iterations against the value of acceleration parameters $\omega$ and $\alpha$ - Method A, Problem III

<table>
<thead>
<tr>
<th>$\omega$</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>42</td>
<td>27</td>
<td>19</td>
<td>18</td>
<td>16</td>
<td>15</td>
<td>15</td>
<td>28</td>
</tr>
<tr>
<td>0.4</td>
<td>30</td>
<td>22</td>
<td>16</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>18</td>
<td>32</td>
</tr>
<tr>
<td>0.8</td>
<td>19</td>
<td>17</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>19</td>
<td>25</td>
<td>42</td>
</tr>
<tr>
<td>1.0</td>
<td>18</td>
<td>15</td>
<td>14</td>
<td>16</td>
<td>19</td>
<td>23</td>
<td>28</td>
<td>46</td>
</tr>
<tr>
<td>1.2</td>
<td>15</td>
<td>15</td>
<td>16</td>
<td>19</td>
<td>23</td>
<td>27</td>
<td>35</td>
<td>56</td>
</tr>
<tr>
<td>1.4</td>
<td>14</td>
<td>14</td>
<td>20</td>
<td>23</td>
<td>28</td>
<td>34</td>
<td>41</td>
<td>71</td>
</tr>
<tr>
<td>1.6</td>
<td>15</td>
<td>18</td>
<td>24</td>
<td>30</td>
<td>36</td>
<td>41</td>
<td>53</td>
<td>91</td>
</tr>
<tr>
<td>2.0</td>
<td>26</td>
<td>33</td>
<td>45</td>
<td>53</td>
<td>66</td>
<td>77</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
</tbody>
</table>

> more than 100 iterations.
<table>
<thead>
<tr>
<th>( \omega )</th>
<th>0.0</th>
<th>0.4</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>42</td>
<td>38</td>
<td>31</td>
<td>30</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>32</td>
</tr>
<tr>
<td>0.4</td>
<td>38</td>
<td>32</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>31</td>
<td>31</td>
<td>33</td>
</tr>
<tr>
<td>0.8</td>
<td>31</td>
<td>30</td>
<td>30</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>41</td>
</tr>
<tr>
<td>1.0</td>
<td>30</td>
<td>30</td>
<td>31</td>
<td>31</td>
<td>30</td>
<td>32</td>
<td>35</td>
<td>47</td>
</tr>
<tr>
<td>1.2</td>
<td>30</td>
<td>30</td>
<td>31</td>
<td>30</td>
<td>32</td>
<td>34</td>
<td>40</td>
<td>52</td>
</tr>
<tr>
<td>1.4</td>
<td>31</td>
<td>31</td>
<td>30</td>
<td>32</td>
<td>34</td>
<td>39</td>
<td>46</td>
<td>60</td>
</tr>
<tr>
<td>1.6</td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>30</td>
<td>40</td>
<td>46</td>
<td>51</td>
<td>69</td>
</tr>
<tr>
<td>2.0</td>
<td>32</td>
<td>33</td>
<td>42</td>
<td>47</td>
<td>54</td>
<td>60</td>
<td>70</td>
<td>92</td>
</tr>
</tbody>
</table>

**TABLE 7.4.5:** The number of iterations against the value of acceleration parameters \( \alpha \) and \( \omega \) - Method B8, Problem III
7.5 CONCLUSION

We observe that for all the model problems tested the original CG and MCG algorithms i.e. without preconditioning \((w=0)\) converge very slowly as indicated by Figure 7.3.1, 7.3.4 and Tables 7.3.4. A superior rate of convergence was achieved when the preconditioning strategy was applied to the original CG algorithms. This approximate preconditioning strategy has further accelerated the already fast convergence rates and has been shown to make it practical to solve the large and sparse equations arising from the discretisation by the finite difference method of Biharmonic equations. The way we choose the coefficients of the inverse triangular sparse matrices is flexible, as long as we keep it reasonably close to the original coefficients.

The MPCG method gives better results when we omitted two of the four nodes in the original inverse triangular sparse matrices. That is, it seems to pay off to vary the adjustable parameter \(a\) and \(w\) (Methods B4 and B5). For the other choices of the approximate precondition strategies (i.e. B1, B2 and B3), it is best to use the preconditioned CG since the introduction of the matrix \(H\) seems to give little gain on the rate of convergence. However, for both algorithms, the approximate preconditioning technique does reduce the considerable amount of work.

For the Laplace problem with a 9-point finite difference formula, these approximate preconditioning techniques also seem to have the advantage in terms of the amount of work required to perform the operations since the Method B7 gives at least a similar result with the Method A.

In the three dimensional case tested the results were not so encouraging, since the approximate preconditioning result in a poorer rate of convergence, although a considerable amount of work has been saved by this technique. However, for this 3-dimensional problem the effect of
this technique was to reduce the preconditioning effect to 2-dimensions only.

The general conclusion that we can draw from these examples is that if we have more than two nodes in one space dimension, we can always reduce it to only two nodes in each dimension and thus save a significant amount of work while the advantage of the original preconditioning (i.e. method A) is at least still retained if not improved. Thus, the approximate preconditioning is very practical and competitive technique to apply when solving the very large sparse matrices that usually arise from the discretization of the elliptic partial differential equations.
REFERENCES


C SOLUTION OF PDE PROBLEMS USING VARIATIONAL METHODS
C ITERATIVE METHOD _ SIMULTANEOUS DISPLACEMENT

C IMPLICIT REAL*8(A-H,O-Z)
REAL*8 LW, LRI, LRN, LPCON, NCON, LNC, LNC
COMMOM /B10/R, CW, RDSI, BI1, BI2, BW1, RLW, RUW
DIMENSION LW(55), YY(50), RHS2(50), LRI(55), LRN(55), LPCON(55)
DIMENSION LNC(30), LNC(30)
DIMENSION RHS(20), CW(20, 20), ZZ(20), CWZ(20)
DIMENSION ZRN(20), XX(20), RHS3(20), ITERAT(30)
DIMENSION BI1(20, 20), BI2(20, 20), RN(20, 20), ZZ(20)
DIMENSION RLW(20, 20), RDL(20, 20), RUW(20, 20), BW(20, 20)
DIMENSION RR(20), DBNS(20), BW1(20, 20), R(20, 20), RDSI(20, 20)
DIMENSION BWB(20, 20), BW(20, 20), BW1(20, 20), AR(20, 20)

COMMOM /IBLK1/K, L, KK, LL
COMMOM /BB1/N
COMMOM /IBLK2/IA2, JA2, PI
COMMOM /BRLS/JJ
EXTERNAL Pl, P2, FA, FB
ABSACC=1.0D-6

C ITMAX IS THE MAXIMUM ITERATION ALLOWED

READ(1,*) MF, ITMAX
PI=3.1415927
THRES=.SE-6
LLW=21
MF1=MF+1
N=MF1*(1+MF1)/2
JJ=0
II=1

DO 121 I=1,N
BANS(I)=0.
DO 121 J=1,N
RDSI(I,J)=0.
RDL(I,J)=0.
RLW(I,J)=0.
BI1(I,J)=0.
BI2(I,J)=0.
RUW(I,J)=0.
BW(I,J)=0.
R(I,J)=0.
RN(I,J)=0.
121 CONTINUE
WRITE(1,12)
12 FORMAT(6X,'RIT MATRIX PROBLEM FLETCHER'/7X,'DIF COM OF K KK'/)
WRITE(1,199)
199 FORMAT(7X,'SIM. DISPLACEMENT METHOD')
WRITE(1,13)
13 FORMAT(7X,'MATRIX SIZE=',I5,4X,'ALPHA= 2/LAM1+LAM. N')
DO 100 K2=0, MF
DO 100 K=K2, 0, -1
L=K2-K
C
DO 200 K3=0, MF
DO 200 KK=K3, 0, -1
LL=K3-KK
C
CALL NAGF D01DAF TO PERFORM THE INTEGRATION FOR
C EVERY COMBINATION OF N AND N
C
IFAIL=1
CALL D01DAF(-1.0D 00,1.0D 00,P1,P2,FA,ABSACC,ANS,
*NPTS,IFAIL)
JJ=JJ+1
IF (JJ.GT.N) GO TO 500
GO TO 400
500 JJ=1
II=II+1
400 R(II,JJ)=ANS
200 CONTINUE
100 CONTINUE
C
TO INTEGRATE THE R.H.S. OF THE RITZ SYSTEM
C
IB2=0
DO 11 IA1=0, MF
DO 11 IA2=IA1, 0, -1
JA2=IA1-IA2
IFAIL=1
CALL D01DAF(-1.0D 00,1.0D 00,P1,P2,FB,ABSACC,ANS,NPTS,IFAIL)
C
IB2=IB2+1
BANS(IB2)=ANS
11 CONTINUE
C
TO NORMALISE THE MATRIX A
C
DO 10 I=1,N
RDSI(I,I)=SQRT(R(I,I))
RDSI(I,I)=1.0/RDSI(I,I)
10 CONTINUE
C
CALL MATNAT(R,RDSI,RDL)
CALL MATNAT(RDSI,RDL,RN)
C
DO 7007 KL=1,LLW
W=KL*0.1-0.1
C
TO OBTAIN THE INVERSE TRIANGULAR MATRICES
C
CALL RLS(RN,RLW,W,N)
CALL RUS(RN,RUW,W,N)
C
DO 70 I=1,N
RLW(I,I)=1.0D 00
RUW(I,I)=1.0D 00
70 CONTINUE
IA =20
IB =20
IFAIL=1
CALL F01AAF(RUW,IA,N,BI1,IB,Z,IFAIL)
IFAIL=1
CALL F01AAF(RLW,IA,N,BI2,IB,Z,IFAIL)

CALL MATMAT(RN,BI1,BW1)
CALL MATMAT(BI2,BW1,BW)
DO 92 I=1,N
DO 92 J=1,N
   BW(A(I,J),J)=BW(I,J)
   BW(B(I,J),I)=BW(I,J)
92 CONTINUE

CALL MAX(RMAX,BW,AR,N)
CALL SUMA(BW,SUH1,N)

CALL NAGF ROUTINE TO EVALUATE THE
EIGENVALUES OF THE MATRIX B

IFAIL=1
CALL F02AAF(BW,IA,N,RR,Z,IFAIL)
IFAIL=1
CALL F01AAF(BWA,IA,N,BWI,IB,Z,IFAIL)
CALL MAX(UHAX,BWI,AR,N)
CALL SUMA(BWI,SUH2,N)
NCON=SUH1*SUH2/N
NCON=RMAX*UHAX*N
PCON=RR(N)/RR(1)
ALPH=2./(RR(1)+RR(N))

LW(KL)=W
LR1(KL)=RR(1)
LRN(KL)=RR(N)
LPCON(KL)=PCON
LNC(KL)=NCON
LKC(KL)=NCON
DO 66 I=1,N
   RHS(I)=RDSI(I,I)*BANS(I)
66 CONTINUE

CALL MATVEC(BI2,RHS,RHS2)

RESET THE MATRIX B

DO 8008 I=1,N
DO 8008 J=1,N
   C(I,J)=BW(I,J)
8008 CONTINUE

ITERATION ON ZZ

DO 553 NH=1,N
   ZZ(NH)=0.0D 00
553 CONTINUE

ITER=0
207 ITER=ITER+1
IF(ITER.GE.ITNAX)GOTO 565
484 FORMAT(3X,'I'1 ','D','F4.1,'I7,'E20.4)
CALL MATVEC(CW,ZZ,CWZ)
DO 552 I=1,N
   ZZN(I)=ZZ(I)+ALPH*(RHS2(I)-CWZ(I))
552 CONTINUE
DO 554 I=1,N
   DIF=DABS(ZZN(I)-ZZ(I))
   IF(DIF.GT.THRES)GO TO 717
554 CONTINUE
GO TO 700
717 CONTINUE
DO 556 I=1,N
   ZZ(I)=ZZN(I)
556 CONTINUE
GO TO 207
700 CALL MATVEC(BIL,ZZ,YY)
DO 990 I=1,N
   XX(I)=RDSI(I,I)*YY(I)
990 CONTINUE
ITERAT(KL)=ITER
GO TO 7007
565 WRITE(1,484)W,ITER,DIF
7007 CONTINUE
WRITE(1,5551)
DO 504 I=1,LL1
   WRITE(1,5055)LW(I),LRI(I),LRN(I),LPCON(I),ITERAT(I)
   LNC(I),LNC(I)
504 CONTINUE
CALL MATVEC(R,XX,RHS3)
WRITE(1,1356)(RHS3(I),I=1,N)
WRITE(1,1357)(BANS(I),I=1,N)
1356 FORMAT(/15X,'R*SOLNX = RHS3 :',8F7.3/31X,8F7.3)
1357 FORMAT(/15X,'RHS OF MATRIX R:',8F7.3/31X,8F7.3)
CALL EXIT
5551 FORMAT(1H1,3X,' W',5X,'SMALLEST EV',7X,'LARGEST EV',5X,
     'PCONDITION NO',5X,'ITER',7X,'NCON',11X,'NCON')</n5055 FORMAT(3X,F4.1,3E17.7,2X,I5,2E17.7)
END
SUBROUTINE TO SEARCH FOR THE MAXIMUM ENTRY OF THE MATRIX B

```fortran
SUBROUTINE MAX(RMAX, BW, AR, N)
IMPLICIT REAL*8(A-H, O-Z)
DIMENSION BW(20, 20), AR(20, 20)
RMAX = ABS(BW(1, 1))
DO 30 I = 1, N
   DO 30 J = 1, N
      AR(I, J) = ABS(BW(I, J))
      RMAX = MAX1(RMAX, AR(I, J))
30 CONTINUE
RETURN
END
```

SUBROUTINE TO SUM ALL ENTRIES OF THE MATRIX B

```fortran
SUBROUTINE SUM(A, SUM, N)
IMPLICIT REAL*8(A-H, O-Z)
DIMENSION A(20, 20)
SUM = 0.
DO 10 I = 1, N
   DO 10 J = 1, N
      SUM = SUM + A(I, J) * A(I, J)
10 CONTINUE
SUM = SQRT(SUM)
RETURN
END
```

SUBROUTINE TO MULTIPLY THE UPPER TRIANGULAR MATRIX BY THE ACCELERATION PARAMETER W

```fortran
SUBROUTINE RUS(RN, RUW, W, N)
IMPLICIT REAL*8(A-H, O-Z)
DIMENSION RUW(20, 20), RN(20, 20)
DO 90 I = 1, N
   DO 90 J = 1, N
      RUW(I, J) = 0.0D 00
90 CONTINUE
N1 = N - 1
DO 61 I = 1, N1
   IP1 = I + 1
   DO 62 J = IP1, N
      RUW(I, J) = RN(I, J) * W
62 CONTINUE
61 CONTINUE
RETURN
END
```
SUBROUTINE TO MULTIPLY THE LOWER TRIANGULAR MATRIX BY THE ACCELERATION PARAMETER W

SUBROUTINE RLS(RN,RLW,W,N)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION RLW(20,20),RN(20,20)
DO 100 I=1,N
DO 100 J=1,N
   RLW(I,J)=0.0D 00
100 CONTINUE
DO 51 I=2,N
   KL=I-1
   DO 52 J=1,KL
      RLW(I,J)=RN(I,J)*W
   52 CONTINUE
51 CONTINUE
RETURN
END

SUBROUTINE TO PERFORM THE MATRIX VECTOR MULTIPLICATION

SUBROUTINE NATVEC(B,V,BV)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION B(20,20),V(20),BV(20)
DO 60 I=1,N
   BV(I)=0.0D 00
DO 60 J=1,N
   BV(I)=BV(I)+B(I,J)*V(J)
60 CONTINUE
RETURN
END

SUBROUTINE TO PERFORM MATRIX MATRIX MULTIPLICATION

SUBROUTINE NATHAT(D,DELA,B)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION D(20,20),DELA(20,20),B(20,20)
DO 203 K=1,N
   DO 203 I=1,N
      B(I,K)=0.0D 00
   DO 203 J=1,N
      B(I,K)=B(I,K)+D(I,J)*DELA(J,K)
203 CONTINUE
RETURN
END

SPECIFY THE LIMITS OF INTEGRATION OF VARIABLE Y

REAL*8 FUNCTION P1(Y)
REAL*8 Y
P1=0.0D 00
RETURN
END

REAL*8 FUNCTION P2(Y)
REAL*8 Y
P2=1.0D 00
RETURN
END
SUBROUTINE TO OBTAIN THE ENTRIES FOR THE RITZ SYSTEM
BY THE LEAST SQUARES METHOD
PROBLEM IN TWO DIMENSION

REAL*8 FUNCTION FA(X,Y)
IMPLICIT REAL*8(A-H,O-Z)
COMMON /IBLK1/K,L,KK,LL

X1=(X-X*X)*X**K
Y1=(1-Y*Y)*Y**2*L
IF(K.EQ.0.AND.L.EQ.0)THEN
  DXI=-2.*Y1
  DYI=-2.*X1
ELSE IF(K.EQ.0)THEN
  DXI=-2.*Y1
  DYI=X1*Y**2*L-2)*(2*L+2)+(2*L+1)*Y*Y
ELSE IF(L.EQ.0)THEN
  DXI=Y1*X**2*K-1)*(K*K+1)+(K+2)*(K+1)*X
  DYI=-2.*X1
ELSE
  DXI=Y1*X**2*K-1)*(K*K+1)+(K+2)*(K+1)*X
  DYI=X1*Y**2*L-2)*(2*L+2)+(2*L+1)*Y*Y
END IF

X2=(X-X*X)*X**KK
Y2=(1-Y*Y)*Y**2*LL
IF(KK.EQ.0.AND.LL.EQ.0)THEN
  DXJ=-2.*Y2
  DYJ=-2.*X2
ELSE IF(KK.EQ.0)THEN
  DXJ=-2.*Y2
  DYJ=X2*Y**2*LL-2)*(2*LL+2)+(2*LL+1)*Y*Y
ELSE IF(LL.EQ.0)THEN
  DXJ=Y2*X**2*K-1)*(KK*K+1)+(KK+2)*(KK+1)*X
  DYJ=-2.*X2
ELSE
  DXJ=Y2*X**2*K-1)*(KK*K+1)+(KK+2)*(KK+1)*X
  DYJ=X2*Y**2*LL-2)*(2*LL+2)+(2*LL+1)*Y*Y
END IF

ASI=DXI+DYI
ASJ=DXJ+DYJ
FA=ASI*ASJ
RETURN
END
SUBROUTINE TO INTEGRATE THE R.H.S. OF THE RITZ SYSTEM

REAL*8 FUNCTION FB(X,Y)
IMPLICIT REAL*8(A-H,O-Z)
COMMON /IBLK2/K,L,PI

BB=(PI*PI/4)*COS(PI*Y/2)*(1-(1-EXP(-PI/2))*X)
XL=(X-X*X)*X**K
YL=(1-Y*Y)*Y**(2*L)

IF(K.EQ.0.AND.L.EQ.0)THEN
 DXI=-2.*YL
 DYZ=2.*Xl
ELSE IF(K.EQ.0)THEN
 DXI=-2.*YL
 DYZ=Xl*Y**(2*L-2)*((K*(K-1))-(2*K+2)*(2*K+1)*Y**2)
ELSE IF(L.EQ.0)THEN
 DXI=Yl*X**(K-1)*((K*(K+1)-(K+2)*(K+1)*X)
 DYZ=-2.*Xl
ELSE
 DXI=Yl*X**(K-1)*((K*(K+1)-(K+2)*(K+1)*X)
 DYZ=Xl*Y**(2*L-2)*((2*L-2)*(2*L+2)*(L-1)-2*L+2)*(2*L+1)*Y**2)
END IF
ASI=DXI+DYZ
FB=BB*ASI
RETURN
END

*****************************************************************
C END OF A COMPLETE PROGRAM TO DETERMINES THE NUMBER OF ITERATIONS FOR VARIATIONAL METHODS IN CHAPTER SIX
C FOR DIFFERENT METHODS OF SOLUTION, THIS PROGRAM REQUIRES DIFFERENT SUBROUTINES FA AND FB

*****************************************************************
SUBROUTINE TO OBTAIN THE ENTRIES FOR THE RITZ SYSTEM BY THE RAYLEIGH_RITZ METHOD

REAL*8 FUNCTION FA(X, Y)
IMPLICIT REAL*8(A-H,O-Z)
COMMON /IBLK1/K, L, KK, LL
X1=(X-X*X)*X**K
Y1=(1-Y*Y)*Y**(2*L)
IF(L.EQ.0) THEN
   DXI=Y1*X**K*((K+1)-(K+2)*X)
   DYI=-2.*X1*Y
ELSE
   DXI=Y1*X**K*((K+1)-(K+2)*X)
   DYI=X1*Y**(2*L-1)*(2*L-(2*L+2)*Y*Y)
END IF
X2=(X-X*X)*X**KK
Y2=(1-Y*Y)*Y**(2*LL)
IF(LL.EQ.0) THEN
   DXJ=Y2*X**KK*((KK+1)-(KK+2)*X)
   DYJ=-2.*X2*Y
ELSE
   DXJ=Y2*X**KK*((KK+1)-(KK+2)*X)
   DYJ=X2*Y**(2*LL-1)*(2*LL-(2*LL+2)*Y*Y)
END IF
FA=DXI*DXJ+DYI*DYJ
RETURN
END

ENTRIES FOR R.H.S. OF RITZ SYSTEM
RAYLEIGH_RITZ METHOD

REAL*8 FUNCTION FB(X, Y)
IMPLICIT REAL*8(A-H,O-Z)
COMMON /IBLK2/K, L, PI

BB=-(PI*PI/4)*COS(PI*Y/2)*(1-(1-EXP(-PI/2))*X)
X1=(X-X*X)*X**K
Y1=(1-Y*Y)*Y**(2*L)
FB=BB*X1*Y1
RETURN
END
C *******************************************************************
C
C PROGRAM TO DETERMINE THE NUMBER OF ITERATIONS
C MODIFIED PRECONDITIONING CONJUGATE GRADIENT METHOD ALGORITHM 7B
C MATRICES B AND H ARE GENERATED
C
C *******************************************************************

IMPLICIT REAL*8(A-H,O-Z)
DIMENSION P(44,44),RN(44,44),Z(44,44)
DIMENSION V(44,44),VN(44,44),R(44,44)
DIMENSION HRN(44,44),S(44,44)

WRITE(1,3040)
3040 FORMAT(/5X,'ALGORITHM CGS-CG-3.5 ',3X,'BC=0'/)
WRITE(1,3031)
3031 FORMAT(/5X,'PLEASE READ N2 KNAX'/)
READ(1,*),N2,KHAX
N7=N2-1
THRES=0.5E-6

! N7 IS THE GRID SIZE
! KHAX IS THE MAXIMUM ITERATION NUMBERS ALLOWED

WRITE(1,303)N7,KHAX
303 FORMAT(/5X, 'BIHARMONIC EQ.3.5 GENERATE B & H : GRID SIZE =',I3/
* /5X, 'MAXIMUM ITERATION ALLOWED IS',I7//)
WRITE(1,3043)THRES
3043 FORMAT(/5X, 'THRESHOLD VALUE IS ',E9.2/)

! W AND ALP ARE THE ACCELERATION PARAMETERS
! FOR THE MATRICES B AND H RESPECTIVELY

WRITE(1,304)
304 FORMAT(/5X, 'PLEASE READ IW1 LLW1 T4 T5',3X,'DO LOOP FOR W'/)
READ(1,*),IW1,LLW1,T4,T5
WRITE(1,305)
305 FORMAT(/5X, 'PLEASE READ IW LLW T1 T2 DO LOOP FOR ALPHA'/)
READ(1,*),IW,LLW,T1,T2
DO 300 I=3,N2
DO 300 J=3,N2
Z(I,J)=0.
V(I,J)=0.
R(I,J)=0.
P(I,J)=0.
RN(I,J)=0.
HRN(I,J)=0.
S(I,J)=0.
RN(J,I)=0.
VN(J,I)=0.

300 CONTINUE
DO 7000 K2=IW1,LLW1
W=K2*T4+T5
DO 8000 K1=IW,LLW
ALP = T1*K1+T2

C INITIAL GUESS FOR THE VECTOR SOLUTION
C
DO 11 I=3,N2
DO 11 J=3,N2
V(J,I) = 0.2*I*J

CONTINUE
IF(K1.NE.I1) GO TO 101
IF(K2.NE.I1) GO TO 101
WRITE(1,503)

503 FORMAT(/20X,'INITIAL GUESS FOR W=',F7.2//)
WRITE(1,202)

202 FORMAT(/20X,'0.2*I*J'/)

CALL BWG(V,Z,U,U2)
DO 30 I=3,N2
DO 30 J=3,N2
R(J,I) = -Z(J,I)
30 CONTINUE

VECTOR P = H*R

CALL HALP(P,P,ALP,N2)
ITER=0

ITERATION BEGINS HERE

4000 ITER=ITER+1
K=ITER
CALL BWG(P,S,V,N2)
CALL VECVEC(P,S,D3,N2)
CALL VECVEC(P,R,C,N2)
IF(K.GE.KMAX)GO TO 8001
AK = C/D3

RESET V AND R

DO 55 I=3,N2
DO 55 J=3,N2
VN(J,I) = V(J,I) + AK*P(J,I)
RN(J,I) = R(J,I) - AK*S(J,I)
55 CONTINUE

TEST FOR CONVERGENCE

DO 9 I=3,N2
DO 9 J=3,N2
DIFF=DABS(V(J,I)-VN(J,I))
IF(DIFF.GE.THRES)GO TO 777
9 CONTINUE
GO TO 888

777 CONTINUE
CALL HALP(RN,HRN,ALP,H2)
CALL VECVEC(S,HRN,ANU,N2)
CALL VECVEC(P,S,D3,N2)
BKIJ = ANU/D3
DO 8 I=3,N2
DO 8 J=3,N2
P(J,I) = HRN(J,I) - BKIJ*P(J,I)
V(J,I) = VH(J,I)
R(J,I) = RH(J,I)
8 CONTINUE

GO TO NEXT ITERATION

GO TO 4000
8001 WRITE(1,105)KMAX
   GO TO 8000
888 WRITE(1,104)ITER
104 FORMAT(/10X,'CONVERGE WITH ITERATION EQUAL TO',15/)
   DO 1046 I=3,N2
   DO 1046 J=3,N2
      DIFF2=DBS(VN(J,I))
      IF(DIFF2.LE.0.1E-5)GO TO 7007
1046 CONTINUE
   WRITE(1,1044)
1044 FORMAT(4X,'SOLUTION VECTOR'/)
   WRITE(1,1045)((VN(I,J),I=3,N2),J=3,N2)
   GO TO 8000
7007 WRITE(1,7006)
7006 FORMAT(/11X,'ALL SOLN SATISFY THRES VALUE'//)
1045 FORMAT(5X,1E11.3//)
105 FORMAT(/10X,'ITERATION GREATER THAN ',15//)
8000 CONTINUE
7000 CONTINUE
   CALL EXIT
END
SUBROUTINE TO GENERATE THE MATRIX B

SUBROUTINE BNG(V2,Z2,W,N2)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION B(44,44),V2(44,44),E(44,44),Z2(44,44)

BIHARMONIC EQUATION - 13 POINT MOLECULE

METHOD A

DO 30 I=1,N2
DO 30 J=1,N2
    B(I,J)=0.
    E(I,J)=0.
    Z2(I,J)=0.
30 CONTINUE

DO 7 IN=3,N2
    I=N2+3-IN
    DO 7 JN=3,N2
        J=N2+3-JN
        B(I,J)=V2(I,J)+0.40*W*(B(I+1,J)+B(I,J+1))
        * -0.1*W*(B(I+1,J+1)+B(I+1,J-1))
        * -0.05*W*(B(I+2,J)+B(I,J+2))
7 CONTINUE

T=A*U

DO 2 I=3,N2
DO 2 J=3,N2
    E(I,J)=B(I,J)-0.4*(B(I+1,J)+B(I-1,J)+B(I,J+1)+B(I,J-1))
    * +0.1*(B(I+1,J+1)+B(I+1,J-1)+B(I-1,J+1)+B(I-1,J-1))
    * +0.05*(B(I+2,J)+B(I-2,J)+B(I,J+2)+B(I,J-2))
2 CONTINUE

Z2=(I-W*L)*INV*T
T=(I-W*L)*Z2

DO 3 I=3,N2
DO 3 J=3,N2
    Z2(I,J)=E(I,J)+0.4*W*(Z2(I-1,J)+Z2(I,J-1))
    * -0.1*W*(Z2(I-1,J+1)+Z2(I-1,J-1))
    * -0.05*W*(Z2(I-2,J)+Z2(I,J-2))
3 CONTINUE
RETURN
END
SUBROUTINE TO GENERATE THE MATRIX H

SUBROUTINE HALP(Q,T2,ALP,N2)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION SS(44,44),Q(44,44),T2(44,44)

DO 30 I=1,44
DO 30 J=1,44
SS(J,I)=0.
T2(J,I)=0.
30 CONTINUE

TO PERFORM THE FORWARD SUBSTITUTION PROCESS

DO 3 I=3,N2
DO 3 J=3,N2
SS(I,J)=Q(I,J)+0.4*ALP*(SS(I-1,J)+SS(I,J-1))
* -0.1*ALP*(SS(I-1,J+1)+SS(I-1,J-1))
* -0.05*ALP*(SS(I-2,J)+SS(I,J-2))
3 CONTINUE

TO PERFORM THE BACKWARD SUBSTITUTION PROCESS

DO 7 IN =3,N2
I=N2+3-IN
DO 7 JN =3,N2
J=N2+3-JN
T2(I,J)=SS(I,J)+0.4*ALP*(T2(I+1,J)+T2(I,J+1))
* -0.1*ALP*(T2(I+1,J+1)+T2(I+1,J-1))
* -0.05*ALP*(T2(I+2,J)+T2(I,J+2))
7 CONTINUE

RETURN
END

SUBROUTINE TO EVALUATE THE INNER PRODUCT

SUBROUTINE VECVEC(A,B,C,N2)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(44,44),B(44,44)
C=0.
DO 5 I=1,44
DO 5 J=1,44
C=C+A(J,I)*B(J,I)
5 CONTINUE
RETURN
END

END OF A COMPLETE PROGRAM TO DETERMINE THE NUMBER OF ITERATIONS USING MPCG METHOD
SEE SUBROUTINES EXAMPLES ON PAGE 291 AND 292
SUBROUTINE TO GENERATE THE MATRIX B

SUBROUTINE BKG(V2,Z2,W,W2)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION B(44,44),V2(44,44),E(44,44),Z2(44,44)

BIHARMONIC EQUATION — 13 POINT MOLECULE

DO 30 I=1,N2
DO 30 J=1,N2
B(I,J)=0.
E(I,J)=0.
Z2(I,J)=0.
30 CONTINUE

DO 7 IN=3,N2
I=N2+3-IN
DO 7 JN=3,N2
J=N2+3-JN
B(I,J)=V2(I,J)+0.50*W*(B(I+1,J)+B(I,J+1))
* -1.0/8.0*W*(B(I+1,J+1))
* -1.0/16.0*W*(B(I+2,J)+B(I,J+2))
7 CONTINUE

T=A*U

DO 2 I=3,N2
DO 2 J=3,N2
E(I,J)=B(I,J)-0.4*(B(I+1,J)+B(I-1,J)+B(I,J+1)+B(I,J-1))
+0.1*(B(I+1,J+1)+B(I+1,J-1)+B(I-1,J+1)+B(I-1,J-1))
+0.05*(B(I+2,J)+B(I-2,J)+B(I,J+2)+B(I,J-2))
2 CONTINUE

Z2=(I-W*L)*INV*U*Z2
T=(I-W*L)*Z2

DO 3 I=3,N2
DO 3 J=3,N2
Z2(I,J)=E(I,J)+0.5*W*(Z2(I-1,J)+Z2(I,J-1))
* -1.0/8.0*W*(Z2(I-1,J-1))
* -1.0/16.0*W*(Z2(I-2,J)+Z2(I,J-2))
3 CONTINUE
RETURN
END
SUBROUTINE TO GENERATE THE MATRIX B FOR
LAPLACE THREE DIMENSIONAL PROBLEM
9_POINT MOLECULE ; METHOD A

SUBROUTINE BWG(V2,Z2,W,N2)
IMPLICIT REAL*4(A-H,O-Z)
DIMENSION B(18,18,18),V2(18,18,18),E(18,18,18),Z2(18,18,18)

DO 30 I=1,N2
DO 30 J=1,N2
DO 30 K=1,N2
   B(I,J,K)=0.
   E(I,J,K)=0.
   Z2(I,J,K)=0.
30 CONTINUE

B=(1-*W)*INV*V2
V2=(1-W*U)*B

W1=W/6.
DO 7 IN=3,N2
   I=N2+3-IN
   DO 7 JN=3,N2
      J=N2+3-JN
      DO 7 KN=3,N2
         K=N2+3-KN
         B(I,J,K)=B(I,J,K)-B(I,J+1,K)-B(I,J,K+1)
         CONTINUE
     T=A*U

DO 2 I=3,N2
DO 2 J=3,N2
DO 2 K=3,N2
   E(I,J,K)=B(I,J,K)-(B(I+1,J,K)+B(I,J+1,K)+B(I,J,K+1))
      T=(I-W*L)*Z2
   CONTINUE

Z2=(1-W*L)*INV*T
Z2=(I-W*L)*Z2

CONTINUE

RETURN
END
*** PROGRAM TO CALCULATE PRECONDITION NUMBER ***

** GENERATE MATRICES B AND H **

** BIHARMONIC EQUATION - TWO DIMENSIONAL PROBLEM **

*** ***

IMPLICIT REAL*8(A-H,O-Z)
DIMENSION Z(44,44),V(44,44)
COMMON THRES

WRITE(1,3040)
3040 FORMAT('/5X,'CALCULATE PCON NUMBER'/)
WRITE(1,3031)
3031 FORMAT('/5X,'PLEASE READ N2 S1 S2 M '/)

M IS THE MAXIMUM ITERATION ALLOWED
S1=0.
S2 IS THE SHIFT TO CALCULATE SMALLEST EIGENVALUES

READ(1,*)N2,S1,S2,M
N7=N2-2
THRES=0.5E-6
WRITE(1,303)N7
303 FORMAT(/5X,'*BILINEAR EQUATION GENERATE B & H : GRID SIZE =',I3)
WRITE(1,3043)THRES
3043 FORMAT('/5X,'*THRESHOLD VALUE IS 'E9.2/')
WRITE(1,304)
304 FORMAT('/5X,'PLEASE READ IW1 LLW1 T4 T5',3X,'DO LOOP FOR W'/)
READ(1,*)IW1,LLW1,T4,T5
DO 300 I=1,44
DO 300 J=1,44
Z(I,J)=0.
V(I,J)=0.
300 CONTINUE

DO LOOP FOR W AT THE VICINITY OF W(OPT)

DO 7000 K2=IW1,LLW1 W=K2*T4+T5

TO EVALUATE THE LARGEST AND THE SMALLEST EIGENVALUE USING THE POWER METHOD

101 CALL POWER(V,Z,N2,ALANDL,W,H,S1)
CALL POWER(V,Z,N2,ALANDS,W,H,S2)

PRECON =ALANDL/ALANDS
WRITE(1,662)
662 FORMAT(/11X,'LANHDA(1) LANHDA(N) PRECON' /)
WRITE(1,661)ALANDL,ALANDS,PRECON
661 FORMAT(7X,3E13.3/) 
7000 CONTINUE
CALL EXIT
END
SUBROUTINE TO EVALUATE PRECONDITION NUMBER USING THE POWER METHOD

SUBROUTINE POWER(Z2, W2, N2, ALAM1, W, H, Q)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION Z2(44,44), W2(44,44)
COMMON THRES

DO 11 I=1,44
DO 11 J=1,44
  Z2(J,I)=1.0/(N2-2)
  W2(J,I)=0.
11 CONTINUE
ITER=1
CALL BWG(Z2, W2, W, N2)

DO 9 I=3,N2
DO 9 J=3,N2
  W2(J,I)=W2(J,I)-Q*(Z2(J,I))
9 CONTINUE
TEMP=0.
WSCALE=0.
DO 2 I=3,N2
DO 2 J=3,N2
  TEMP=TEMP+W2(J,I)*Z2(J,I)
  WSCALE=WSCALE+W2(J,I)*W2(J,I)
2 CONTINUE
WSCALE=DSQRT(WSCALE)
ESTOLD=TEMP

ITER=ITER+1
DO 4 I=3,N2
DO 4 J=3,N2
  Z2(J,I)=W2(J,I)/WSCALE
4 CONTINUE
CALL BWG(Z2, W2, W, N2)
DO 90 I=3,H2
DO 90 J=3,H2
  W2(J,I)=W2(J,I)-Q*Z2(J,I)
90 CONTINUE
TEMP=0.
WSCALE=0.
DO 6 I=3,N2
DO 6 J=3,N2
  TEMP=TEMP+W2(J,I)*Z2(J,I)
  WSCALE=WSCALE+W2(J,I)*W2(J,I)
6 CONTINUE
WSCALE=DSQRT(WSCALE)
ESTNEW=TEMP

IF(ABS(ESTOLD-ESTNEW) .LE. THRES) GO TO 7
IF(ITER.GE.11) GO TO 8
ESTOLD=ESTNEW
GO TO 3
7 \text{ITERN}=1 \\
\text{ALAND1}=\text{ESTNEW}+Q \\
\text{WRITE}(1,71)\text{ITERN,ALAND1,ITER} \\
\text{RETURN} \\
8 \text{ITERN}=2 \\
\text{ALAND1}=\text{ESTNEW}+Q \\
\text{WRITE}(1,81)\text{ITERN,ALAND1,M} \\
\text{RETURN} \\
71 \text{FORMAT}(5X,\text{ITERN}=',13,4X,\text{LAMDA}=',11.3,4X,\text{ITERATION}=',16) \\
81 \text{FORMAT}(5X,\text{ITERN}=',13,4X,\text{LAMDA}=',11.3,4X,\text{ITER GT}',16) \\
\text{END}