THE FAST GALERKIN METHOD FOR THE NUMERICAL SOLUTION OF INTEGRAL AND INTEGRO-DIFFERENTIAL EQUATIONS

SAAD M. HASHMI

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor of Philosophy

Department of Statistics and Computational Mathematics

October 1982

ABSTRACT

In this work we investigate the possibility of applying a Fast Galerkin scheme to linear singular integral equations of the first and second kind, and also to linear Integro-differential equations (singular or non-singular),in order to obtain stable, economic and fast methods to solve these problems numerically.

In Chapter 2, we give a generalization of the Fast Galerkin scheme of Delves (1977a). In Chapter 3 we apply the scheme on the Laplace transform inversion, while in Chapter 4 we give a reliable algorithm for the numerical solution of Cauchy singular integral equations.

We also extend, in Chapter 5, the Fast Galerkin scheme to solve general linear Integro-differential equations (singular and non-singular) of order one and two. Finally in Chapter 6 we apply the Fast Galerkin method on the eigenvalue problem.

Numerical examples for both integral and Integro-differential equations are included to illustrate the methods.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor L.M.Delves, for his invaluable suggestions and encouragement which made it possible for me to complete this work. My thanks are due to Dr. J. Schonfelder for providing us with an arbitary precision algorithm.

My thanks also to Sandra Bragh who typed the manuscript.

CONTENTS

CHAPTER 1

INTRODUCTION

This work is concerned with the numerical solution of integral equations and integro-differential equations (of order one and two). We are not principally concerned with the abstract theory of integral equations, nor with applications where integral equations arise.

We shall not give a general definition for integral equations. In fact we limit consideration to some of the most important classes of integral equations. They are summarized below. An integral equation is called linear if linear operations are performed in it upon the unknown function, that is if it has the form

$$A(s)\ \phi(s) + B(s) + \int_R k(s,t)\ \phi(t)\ dt = 0$$

where $\phi(s)$ is the unknown function, $A(s)$, $B(s)$ and $k(s,t)$ are given functions and the integration extends over some domain R of the variable t.

1.1 FREDHOLM EQUATIONS

These represent one of the most important classes of linear integral equations. Let ϕ be the unknown function in the integral equation and g, k will be the known functions where k is called the kernel.

The integral equation

$$\int_a^b k(s,t)\ \phi(t)\ dt = g(s) \qquad ; \qquad a \leq s \leq b \tag{1.1.1}$$

is termed a Fredholm integral equation of the first kind, while the integral equation

$$\phi(s) + \lambda \int_a^b k(s,t)\, \phi(t)\, dt = g(s) \qquad ; \qquad a \leq s \leq b \qquad (1.1.2)$$

is termed as a second kind Fredholm integral equation, where λ is a (possibly) complex scalar parameter.

If in (1.1.2) $g(s) = 0$ we have

$$\phi(s) + \lambda \int_a^b k(s,t)\, \phi(t)\, dt = 0 \qquad ; \qquad a \leq s \leq b \qquad (1.1.3)$$

These kind of integral equations are called homogenous equations of the second kind referred as an eigenvalue equation or a Fredholm equation of the third kind for a given value of λ . We devote chapter (6) for finding the numerical solution of a simple real eigenvalue for this kind of problem.

1.2 VOLTERRA EQUATIONS

If $k(s,t) = 0,\ t > s$ then the kernel is said to be of Volterra-type. The equations (1.1.1-3) may be written in the form

$$\int_a^s k(s,t)\, \phi(t)\, dt = g(s) \qquad ; \qquad a \leq s \qquad (1.2.1)$$

$$\phi(s) + \lambda \int_a^s k(s,t)\, \phi(t)\, dt = g(s) \qquad a \leq s \qquad (1.2.2)$$

$$\phi(s) + \lambda \int_a^s k(s,t)\, \phi(t)\, dt = 0 \qquad a \leq s \qquad (1.2.3)$$

In general a Volterra integral equation of the first kind (1.2.1) can be reduced to a Volterra integral equation of the second kind, (1.2.2) (see, [7], p.8). If $k(s,t)$ is square integrable and we need a square integrable solution $\phi(s)$, it can be shown (see [58]) that equation (1.2.3) has only the trivial eigenfunction $\phi(s) = 0$ for any finite eigenvalue λ.

1.3 SINGULAR INTEGRAL EQUATIONS

Suppose that the function $\phi(s)$ is defined in the interval $a \leq s \leq b$ and is integrable in each of the intervals $a \leq s \leq c-\varepsilon$ and $c+\varepsilon \leq s \leq b$, however small the positive number ε. The Cauchy principal value of the integral of the function $\phi(s)$ in the interval $a \leq s \leq b$ is the name given to the limit (if this exists):

$$\underset{\varepsilon \to 0}{\text{Limit}} \left[\int_a^{c-\varepsilon} \phi(s)\, ds + \int_{c+\varepsilon}^b \phi(s)\, ds \right] \qquad (1.3.1)$$

We often speak of the singular integral instead of "the Cauchy principal value of an integral".

We shall denote the principal value of an integral by the symbol

$$\int\!\!\!\!\!\!-_a^b \phi(s)\, ds$$

Integral equations containing integral in the sense of the Cauchy principal value, with integrands having a singularity in the domain of integration, will be called singular integral equations.

A function ϕ defined on $[\underline{a}, \underline{b}]$ is said to satisfy a Lipschitz condition on $[\underline{a}, \underline{b}]$ if there exists a constant $C > 0$ such that

$$|\phi(s_1) - \phi(s_2)| \leq C\,|s_1 - s_2| \qquad (1.3.2)$$

For all $s_1, s_2 \in [\underline{a}, \underline{b}]$, (this is often called a Hölder condition), one can introduce an important class of singular integrals called Cauchy-type integrals of the form

$$\frac{1}{\pi} \int_R \frac{\phi(t)}{t-s}\, dt \quad , \qquad s \in R \qquad (1.3.3)$$

it is evident that the integral exists if $\phi(t)$ satisfies a Hölder condition. (See [43]).

Lemma (1.1)

If ϕ and ψ are L_2-functions with region R then (see [64] p.163)

$$\int_R \psi(t) \int_R \frac{\phi(s)}{s-t}\, ds\, dt = \int_R ds \int_R \frac{\phi(s)\,\psi(t)}{s-t}\, dt \qquad (1.3.4)$$

Chapter (3,4) are devoted to consider the numerical solution of singular integral equations where in chapter (3) we consider the numerical solution for the inversion of Laplace Transform, while in chapter (4) we give a reliable method for the numerical solution of Cauchy-type singular integral equations.

1.4 FREDHOLM INTEGRO-DIFFERENTIAL EQUATIONS

Chapter (5) of this work is devoted for a special class of integro-differential equations (of order one and two) with linear boundary condition (s). A linear Fredholm integro-differential E.'s of order $n \geq 1$ may be defined as

$$\sum_{i=o}^{n} B_i(s) \; \phi^{(i)}(s) + \lambda \int_a^b k(s,t)\, \phi(t)\, dt = g(s) \quad ; \quad a \leq s \leq b \qquad (1.4.1)$$

Where $B_i(s)$; $0 \leq i \leq n$, are known coefficient functions; $k(s,t)$ the known kernel; λ a given parameter; and $\phi(s)$ is the unknown function (λ, a, b are finite and real).

To solve (1.4.1) we assume that the following linear boundary condition ($1 \leq n \leq 2$)

$$C\phi(\underline{b}) + D\phi'(\underline{b}) = \underline{e} \qquad (1.4.2)$$

is given where $\underline{b}$ consists of m boundary points $b_1, b_2, \ldots, b_m$ belonging to $[\underline{a}, \underline{b}]$; C, D are two given matrices each of size nxm; $\underline{e}$ a given nx1 vector; and

$$\phi(\underline{b}) = (\phi(b_1), \phi(b_2), \ldots, \phi(b_m))^t \quad ;$$

$$\phi'(\underline{b}) = (\phi'(b_1), \phi'(b_2), \ldots, \phi'(b_m))^t$$

it is also assumed that all the functions involved and the boundary condition are such that there exists a unique solution to (1.4.1) conditioned by (1.4.2).

Fredholm integro-differential equations yield in general well-conditioned problems, for they can be converted to Fredholm integral equations of the second kind (see Linz (1974)).

In chapter (5) we apply the "Fast Galerkin scheme" for solving linear Fredholm integro-differential equations (1.4.1).

In the following sections we give a brief discussion on the most widely used numerical methods to solve the integral equations. Where we restrict our discussion on equations of the second kind (1.1.2), in which we require that the parameter λ is a regular value and that $k(s,t)$, $g(s)$ at least piecewise-continuous. In this work we suppose (unless stated otherwise) that $g(s)$ is continuous for $a \leq s \leq b$, that $k(s,t)$ is continuous for $a \leq s, t \leq b$, and that we seek the solution $\phi(s)$ for $a \leq s \leq b$.

The accuracy attainable with any method for the approximate solution of (1.1.2) may be limited by the equation itself. When small perturbation in $g(s)$ cause a large change in the solution, in this case the equation is said to be ill-conditioned, equations of the first kind are often known by this, hence any numerical method must be applied with caution if accuracy is required; a desirable feature of a method is that it can be applied in a way which gives warning of ill-conditioning.

1.5 QUADRATURE METHODS

A Fredholm equation of the second kind can be approximated in a straightforward way be means of quadrature formulae; more details, with practical examples, may be found in [7, 13] .

Suppose that we have made a choice of a quadrature rule to approximate the integral

$$\int_a^b \phi(t)\, dt$$

of the form $q(\phi) = \sum_{j=o}^{N} W_j \, \phi(t_j)$ (1.5.1)

involving the N+1 points t_j and the corresponding weights W_j. Such an integration rule can be used to replace the integral equation (1.1.2) by the equation

$$\tilde{\phi}(s) = g(s) - \lambda \sum_{j=o}^{N} W_j k(s,t_j) \, \tilde{\phi}(t_j) \qquad a \leq s \leq b \qquad (1.5.2)$$

in which the solution of this functional equation may be regarded as an approximation to $\phi(s)$ which may be found by setting $s=t_i$; $i=o(1)N$ in (1.5.2) to obtain

$$\tilde{\phi}(t_i) = g(t_i) - \lambda \sum_{j=o}^{N} W_j k(t_i, t_j) \, \tilde{\phi}(t_j) \; ; \qquad i = o(1)N \qquad (1.5.3)$$

if $\tilde{\phi}(t_o), \tilde{\phi}(t_1), \ldots . \tilde{\phi}(t_N)$ satisfy these equations, can be found then we can obtain the solution of (1.5.2) on setting for all $s \in [a, b]$. However, we may represent the approximate solution by the function values of the vector solution of the linear algebraic system, written in the matrix form:

$$(I + \lambda BD)\underline{\tilde{\phi}} = \underline{g} \qquad (1.5.4)$$

where the solution $\tilde{\phi}_j$ is regarded as an approximation values of $\phi(s)$ at the points $s = t_j$,

I is the identity matrix

$\underline{g} = (g(t_o), g(t_1), \ldots . , g(t_N))^t$, $B = [k(t_i,t_j)]$

and $D = \text{diag}(W_o, W_1, \ldots . , W_N)$.

We have assumed that λ is a regular value in the equation (1.1.2) so that there is a unique solution $\phi(s)$. It may happen that the matrix $(I + \lambda BD)$ is singular for an arbitrary quadrature rule.

One can guarantee (under mild restrictions) that the matrix in (1.5.4) is non-singular, if the quadrature rule (1.5.1) is sufficiently accurate. The result of the computation is an approximation to the solution of the integral equation, and we must now estimate the error of the approximation, and study means by which the error may be reduced. This problem is naturally linked with the original choice of quadrature formulae. For the conditions to be made on the choice of the quadrature rule and the error estimate one can see ([13], p.67; [7], p.432) they depend on the parameter λ and the kernel $k(s,t)$. However the choice of the quadrature rule should also depend on the driving term $g(s)$, since the behaviour of this function has an influence on the solution $\phi(s)$, and hence on the accuracy obtainable.

1.6 EXPANSION METHODS

In the last section we obtained an approximation

$$\tilde{\phi}(s) = g(s) - \lambda \sum_{j=0}^{N} W_j \, k(s,t_j) \, \tilde{\phi}(t_j) \qquad (1.6.1)$$

to the integral equation (1.1.2) which is described by the system (1.5.4) where the approximation $\tilde{\phi}(s)$ (determined by the weights and abscissae of a quadrature rule) is a linear combination of $g(s)$ and $k(s,t_j)$. Sometimes it is convenient to choose a set of functions $\{h_i(s)\}$ (which may depend on N) but independent of $k(s,t)$ and to approximate $\phi(s)$ by

$$\phi_N(s) = \sum_{j=o}^{N} a_j \, h_j(s) \qquad (1.6.2)$$

as a linear combination of prescribed functions $h_i(s)$; i = 0(1)N for the choice of this functions (see, for example [13], p.87), then determine uniquely its expansion coefficients $\underline{a}$ by substituting (1.6.2) into the integral equation (1.1.2) will give the residual

$$\eta_N(s) = \phi_N(s) + \lambda \int_a^b k(s,t) \, \phi_N(t) \, dt - g(s) \qquad (1.6.3)$$

We cannot choose a_i ; i = 0(1)N to make $\eta_N(s)$ vanish identically unless the true solution is a linear combination of the basis $h_i(s)$; i = 0(1)N. However, suitable constraints can be imposed on the choice of a_i ; i = 0(1)N which ensure $\eta_N(s)$ is in some sense small.

There is a number of methods which are basically rather similar, employed to solve (1.6.3) for the unknowns $\underline{a}$ for example, Rayleigh-Ritz, Galerkin, Collocation and Least-squares methods, but in general it is not possible to choose $a_0, a_1, \ldots, a_N$ to ensure that $\eta(s) \equiv 0$. In the Collocation method for example the coefficients a_i ; i = 0(1)N are chosen so that

$$\phi_N(\overset{*}{s}_i) + \lambda \int_a^b k(\overset{*}{s}_i, t) \, \phi_N(t) \, dt = g(\overset{*}{s}_i)$$

where $\overset{*}{s}_i$; i = 0(1)N are selected points in [a, b]. The equations for $\underline{a}$ in the matrix form are

$$(A + \lambda B) \, \underline{a} = \underline{g} \qquad (1.6.4)$$

where

$$A_{i,j} = h_j(s_i^*) \quad \text{and} \quad B_{i,j} = U_j(s_i^*)$$

$$\text{with} \quad U_j(s) = \int_a^b k(s,t)\, h_j(t)\, dt$$

$$\underline{g} = (g(s_0^*),.....,g(s_N^*))^t \quad ; \quad \underline{a} = (a_0,.....,a_N)^t$$

The system of equations (1.6.4) for some particular choice of the points s_i^*, $i = 0(1)N$ can be singular even if λ is a regular value of the kernel $k(s,t)$. The conditioning of the system (1.6.4) depends on the choice of the basis functions $h_j(s)$ and the choice of the Collocation points s_i^* and the choice of one should be matched with the choice of the other (see Baker [7] p.396).

In the weighted Galerkin scheme we obtain a linear system of the form (1.6.4) with

$$A_{i,j} = \int_a^b W(s)\, h_j(s)\, \overline{h_i(s)}\, ds \quad ; \quad B_{i,j} = \int_a^b W(s)\, U_j(s)\, \overline{h_i(s)}\, ds$$

$$\text{with} \quad U_j(s) = \int_a^b k(s,t)\, h_j(t)\, dt \quad ; \quad g_i = \int_a^b W(s)\, g(s)\, \overline{h_i(s)}\, ds$$

where $W(s)$ is a positive function on $[\underline{a}, \underline{b}]$.

The Galerkin method reduces to Rayleigh-Ritz method if $k(s,t) = \overline{k(t,s)}$, and $W(s) = 1$.

In chapter (2) we give a comprehensive discussion on the Galerkin approach showing how this method is computationally convenient for handling such integrals in a lower cost with a satisfactory accuracy.

Finally, we attempt to give a brief mention of the previous numerical methods described in section (1.5-6). One can note that most of the methods for the numerical solution of integral equations can be regarded as expansion methods in some sense. Thus, whilst the quadrature method described in section (1.5) yields a vector $\tilde{\underline{\phi}}$ of function values, these values are used in the Nyström extension to yield the approximation:

$$\tilde{\phi}(s) = g(s) - \sum_{j=0}^{N} b_j k(s,t_j) \; ; \quad b_j = \lambda W_j \, \tilde{\phi}(t_j)$$

which shows that $\tilde{\phi}(s)$ is a linear combination of $g(s)$, $k(s,t_j)$. This suggests that with some quadrature rules, the quadrature method is particularly convenient if we like to generate an approximation of the form

$$\tilde{\phi}(s) = \sum_{r=0}^{N} \tilde{b}_r \, h_r(s)$$

from the vector $\tilde{\underline{\phi}} = (\tilde{\phi}(t_0), \tilde{\phi}(t_1), \ldots\ldots, \tilde{\phi}(t_N))^t$, where t_r and the functions $h_r(s)$ are specially matched. This idea is used by EL-Gendi [18] in an application of a quadrature method with the assumption that $\phi(s)$ is defined and well-behaved in $[-1, 1]$ then Clenshaw and Curtis [10] give the following procedure for the numerical integration of $\phi(s)$ based on the approximation

$$\phi(s) = \sum_{r=0}^{N}{}'' \; a_r \, T_r(s)$$

where

$$a_r = \frac{2}{N} \sum_{j=0}^{N}{}'' \phi(s_j)\, T_r(s_j).$$

and $s_j = \cos\left(\frac{j\pi}{N}\right)$; $j = 0(1)N$

Here $T_r(s)$ is the r-th Chebyshev polynomial. The double primes denotes a sum with first and last terms halved.

Problems which contain weak or straong singularities in the kernel can often be better or more simply treated by choosing the suitable expansion set, or of inner product, within an expansion method, than by a quadrature method, chapters (3,4) are devoted for such strong singular problems, which will suggest that the more difficult the problem the more worthwhile it is to look at expansion methods.

The stability of the methods depend on the error sources which comes from (i) setting up the equations (ii) solving the defining equations, where usually the first is the dominant one, especially if quadrature rule is used to evaluate the matrix B. For a wide discussion of the theoretical basis of stability and rate of convergence, the reader is referred to [7, 13, 14].

In chapter (2) we outline the basis of the Galerkin method followed by the Fast Galerkin technique which is the method of [14].

We apply this scheme on the Laplace transform inversion in chapter (3), while in chapter (4) we give a nice scheme using the technique of chapter (2) for computing the numerical solution of Cauchy singular integral equations [30].

In chapter (5) we adopt the scheme of [2] for solving the linear integro-differential equations of order one and two of Fredholm-type on the range [-1,1]. At the end of chapter (5) we use this technique together with that of chapter (4) for solving a singular integro-differential equation of Cauchy kernel.

Finally in chapter (6) we apply the scheme of chapter (2) to eigenvalue problem for finding the simple (real) eigenvalue.

CHAPTER 2

FAST GALERKIN METHOD

2.1 BASIS OF GALERKIN METHOD

Suppose that we are given an equation of the form:

$$K\phi - g = 0 \qquad (2.1.1)$$

Where K is an operator defined in some Hilbert space H where no further assumption is made about K.

The Galerkin method requires us to select a sequence of elements $\psi_N \in D(K)$ and to attempt to find an approximate solution in the form

$$\phi_N = \sum_{i=0}^{N} a_i \psi_i \qquad (2.1.2)$$

the coefficients a_i are determined from the condition that the inner product of the left hand side of equation (2.1.1) and the sequence ψ_i is zero, in other words after we substitute ϕ_N for ϕ in equation (2.1.1) must be orthogonal to the elements $\psi_0, \psi_1, \ldots\ldots, \psi_N$ which leads to the system of equations (linear or non-linear) depends on the operator K

$$\sum_{i=0}^{N} (K\psi_i, \psi_j) a_i = (g, \psi_j) \; ; \quad j = 0(1)N \qquad ** \qquad (2.1.3)$$

The method can also be applied to the eigenvalue problem which require to find the eigenvalues of the equation. (see [44]).

$$K\phi = \lambda\phi \qquad (2.1.4)$$

where the method approximates the eigenvalues as the roots of the equation

$$\sum_{i=0}^{N} (K\psi_i, \psi_j) a_i = \lambda \sum_{i=0}^{N} (\psi_i, \psi_j) a_i \qquad (2.1.5)$$

$$j = 0(1)N$$

** The unweighted inner product is defined as:

$$(f_1, f_2) = \int_D f_1(s) f_2(s) \, ds \quad ; \quad s \in D$$

2.2 FAST GALERKIN METHOD FOR THE SOLUTION OF I.E's

We consider here the solution of Fredholm integral equations of the second kind

$$f(s) + \lambda \int_a^b k(s, t)\, f(t)\, dt = g(s) \; ; \; a \leq s \leq b \qquad (2.2.1)$$

$$; \; \lambda \text{ is real}$$

although the method can be applied to first kind integral equation and Volterra type equations and also the equations may be linear or non-linear.

We have to select a sequence of functions $h_n(s)$ $(n = 1, 2, ...)$ in $L_2(a, b)$ such that the functions $h_n(s)$ are linearly independent and trying to find an approximate solution to $f(s)$ in $L_2(a, b)$ of the form

$$f_N(s) = \sum_{i=0}^{N} a_i h_i(s) \qquad (2.2.2)$$

where the coefficients a_i make the residual of equation (2.2.1) be zero, that is:

$$\sum_{i=0}^{N} a_i \int_a^b \left[h_i(s) + \lambda \int_a^b k(s, t)\, h_i(t)\, dt \right] h_j^*(s)\, ds$$

$$= \int_a^b g(s)\, h_j^*(s)\, ds \qquad j = 0(1)N \qquad (2.2.3)$$

We require the residual tobe orthogonal to the functions $h_j^*(s)$; if λ is not a characteristic value the system (2.2.3) has a unique solution when N is sufficiently large; as $N \longrightarrow \infty$, the approximate solution f_N in equation (2.2.2) approaches the exact solution $f(s)$ of equation (2.2.1) in the metric space of $L_2(a, b)$. (see [44]).

And we have the estimate that is

$$\| f - f_N \| \leq (1 + \varepsilon_N) \quad \| f - P_N f \| \qquad (2.2.4)$$

where P_N is the projection operator into the space spanned by the functions $h_0, h_1, \ldots, h_N$, and $\varepsilon_N \longrightarrow 0$ as $N \longrightarrow \infty$.

2.3 THE FAST GALERKIN ALGORITHM

There have been a number of methods proposed for Fredholm,Volterra-type equations based on expansions in terms of Chebyshev polynomials.

The solution of integral equations in Chebyshev series has been the subject of two papers by [19, 20] His method essentially a Collocation method and it is necessary to decide in advance how many terms in the Chebyshev series are likely to be significant; the method by Scraton [56] suggested a way to avoid this difficulty of Elliott.

He transformed the integral equation into an infinite set of algebraic equations in which the unknowns are the coefficients of the Chebyshev series and he solved the algebric system by a standard iterative procedure, in which it is not necessary to determine beforehand how many coefficients are significant as in Elliott. The method of El-Gendi [18] is also essentially a modification of the Nyström scheme.

We describe here an alternative method [14] which is a variant of the Galerkin scheme but has the advantage of being significantly faster than the standard Galerkin method.

For operations count and comparison of various methods using Chebyshev expansions the reader is referred to [14] for more details.

We consider in this chapter the solution of linear Fredholm integral equation of the second kind with smooth kernels and driving terms by assuming that the variables of equation (2.2.1) have been suitably transformed so as to reduce the range of integration to (-1, 1).

The method can be applied to first kind Fredholm equations and to Volterra type linear or non-linear equations [see chapter (3,4)]

We restrict our discussion here on the solution of Fredholm equation of the second kind.

So that the equation to be solved has the form

$$f(x) + \int_{-1}^{1} k(x, y)\, f(y)dy = g(x) \quad ; -1 \leq x \leq 1 \qquad (2.3.1)$$
$$; \ \lambda = 1$$

the method is based on expanding the defined functions numerically using the Fast Fourier Transform. $k(x, y)$ and $g(x)$ are assumed in this chapter to be smooth.

For the case when the defining equations contains a known singularity we expand the functions analytically using a set of recurrence relations which we do in chapters (3), (4).

We choose here the basis functions $h_i(x)$ of equation (2.2.2) the Chebyshev polynomials $T_i(x)$, $i = o, 1, 2, \ldots$ to retain the "natural" suffix where we count from zero and make the following expansion for $f(x)$ assuming that $f(x) \in L_2[-1, 1]$

$$f(x) \approx f_N(x) = \sum_{j=o}^{N} a_j\, T_j(x) \qquad (2.3.2)$$

The Chebyshev polynomials are orthogonal on (-1,1) with the weight function:

$$W(x) = (1 - x^2)^{-\frac{1}{2}} \qquad (2.3.3)$$

$$\int_{-1}^{1} W(x)\, T_i(x)\, T_j(x)\, dx = \pi \begin{cases} 1 \ ; i = j = o \\ \frac{1}{2} \quad i = j > o \\ o \quad i \neq j \end{cases} \qquad (2.3.4)$$

And therefore we introduce this weight function into the inner product in the equation (2.2.3) hence applying Galerkin technique on equation (2.3.1) using (2.3.2) and the property (2.3.4) we have:

$$\int_{-1}^{1} \left\{ \sum_{j=0}^{N} a_j T_j(x) + \sum_{j=0}^{N} a_j \int_{-1}^{1} k(x, y)\, T_j(y)\, dy - g(x) \right\} W(x)\, T_i(x)\, dx = 0$$

$$\sum_{j=0}^{N} a_j \left[\int_{-1}^{1} W(x)\, T_i(x)\, T_j(x)\, dx + \int_{-1}^{1} W(x)\, T_i(x) \int_{-1}^{1} k(x, y)\, T_j(y)\, dy\, dx \right]$$

$$= \int_{-1}^{1} W(x)\, T_i(x)\, g(x)\, dx \quad ; \; i=0(1)N$$

We can write this in a matrix form as:

$$(D + B)\, \underline{a} = \underline{g} \tag{2.3.5}$$

where D is the diagonal matrix (2.3.4) and

$$B_{i,j} = \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{-1}^{1} k(x, y)\, T_j(y)\, dy \qquad i, j = 0(1)\, N \tag{2.3.6}$$

$$g_i = \int_{-1}^{1} W(x)\, T_i(x)\, g(x)\, dx \quad ; \quad i = 0(1)\, N \tag{2.3.7}$$

2.4 NUMERICAL IMPLEMENTATION

We need to perform the integrals in equations (2.3.6), (2.3.7) numerically.

Adiscrete Galerkin calculation uses an appropriate quadrature rule to approximate the integrals in (2.3.6), (2.3.7). A suitable rule for the integration over x is clearly the Gauss-Chebyshev (P + 1) point quadrature rule. With weights

$$W_k = \frac{\pi}{P}, \; k \neq 0, P$$

$$W_0 = W_P = \frac{\pi}{2P} \tag{2.4.1}$$

and points $x_k = \text{Cos}\ (\frac{\pi k}{P})$, $k = 0(1).....$

hence we approximate the integral in (2.3.7) as:

$$g_i \sim \bar{g}_i = \frac{\pi}{P} \sum_{k=0}^{P''} g(\text{Cos}(\tfrac{\pi k}{P}))\ T_i(\text{Cos}(\tfrac{\pi k}{P}))$$

$$\bar{g}_i = \frac{\pi}{P} \sum_{k=0}^{P''} g(\text{Cos}(\tfrac{\pi k}{P}))\ \text{Cos}(\tfrac{\pi i k}{P}) \qquad (2.4.2)$$

Where the symbol Σ'' implies that the first and last term are halved, and we have used the relation

$$T_i(x) = \text{Cos}\ i\ (\text{Cos}^{-1} x) \qquad (2.4.3)$$

An attempt to use a product form of this rule for the integral over y to approximate the integral in (2.3.6) would lead to a large numerical error; because of the "missing" weight function $(1 - y^2)^{-\frac{1}{2}}$.

The difficulty is overcome in practice (Elliott, [20]; Scraton, [56]) by introducing the matrix $\bar{K}$ with elements

$$\bar{K}_{i,j} = \int_{-1}^{1} W(x)\ T_i(x)\ dx \int_{-1}^{1} W(y)\ T_j(y)\ k(x, y)\ dy \qquad (2.4.4)$$

then $\bar{K}$ can be efficiently approximated by K :

$$K_{i,j} = (\frac{\pi}{P})^2 \sum_{r=0}^{P''} \sum_{s=0}^{P''} k(\text{Cos}\ (\tfrac{r\pi}{P}),\ \text{Cos}(\tfrac{s\pi}{P}))\ \text{Cos}(\tfrac{i\pi r}{P})\ \text{Cos}(\tfrac{js\pi}{P}) \qquad (2.4.5)$$

$$i, j = 0(1)N$$

Now apart from a constant factor, we can identify $\bar{K}_{i,j}$ as the $(i, j)^{th}$ coefficient in the double Chebyshev expansion of the function $k(x, y)$

that is

$$k(x, y) = \frac{4}{\pi^2} \sum_{i=0}^{\infty}{}' \sum_{j=0}^{\infty}{}' \bar{K}_{i,j} T_i(x)\, T_j(y) \qquad (2.4.6)$$

While apart from the factor $\frac{4}{\pi^2}$(2.3.6) identifies $B_{i,j}$ as the $(i,j)^{th}$ coefficient in an expansion of the function $k(x, y)\,(1-y^2)^{\frac{1}{2}}$. From (2.3.6) and (2.4.4) we have:

$$\bar{K}_{i,j}\,(1-y^2)^{\frac{1}{2}} = \int_{-1}^{1} W(x)\, T_i(x) \int_{-1}^{1} T_j(y)\, k(x,y)\, dy\, dx \qquad (2.4.7)$$

by multiplying both sides of (2.4.6) by the function $(1 - y^2)^{\frac{1}{2}}$ we get:

$$k(x, y)\,(1 - y^2)^{\frac{1}{2}} = \frac{4}{\pi^2} \sum_{i=0}^{\infty}{}' \sum_{j=0}^{\infty}{}' \bar{K}_{ij}(1 - y^2)^{\frac{1}{2}}\, T_i(x)\, T_j(y) \qquad (2.4.8)$$

therefore we can obtain the coefficients $B_{i,j}$ from $\bar{K}_{i,j}$ by using the Chebyshev expansion of $(1 - y^2)^{\frac{1}{2}}$ and multiplying the two series together Using the identity

$$2T_i(x)\, T_j(x) = T_{i+j}(x) + T_{|i-j|}(x) \qquad (2.4.9)$$

where the Chebyshev expansion of $(1 - y^2)^{\frac{1}{2}}$ is:

$$(1 - y^2)^{\frac{1}{2}} = \frac{2}{\pi} - \frac{4}{\pi} \sum_{r=1}^{\infty} \frac{T_{2r}(y)}{4r^2-1} \qquad (2.4.10)$$

hence we have the identity:

$$\frac{\pi}{2} B_{i,j} = \bar{K}_{i,j} - \sum_{r=1}^{\infty} \frac{1}{4r^2-1} (\bar{K}_{i,j+2r} + \bar{K}_{i,|j-2r|}) \qquad (2.4.11)$$

$$i, j = 0(1).....$$

Therefore we evaluate an approximation K to $\bar{K}$ using a product $(P+1)$ point Gauss-Chebyshev rule, and an approximation $\bar{B}$ to B using the identity (2.4.11) with the convention that $K_{i,j} = o$, $i, j > N$

The standard cost for performing the sum on (2.4.5) the process suggests an N^2P^2 dependence but the simple two stages of (Scraton, [56]).

$$S_{r,j} = \frac{\pi}{P} \sum_{q=o}^{P''} k(\mathrm{Cos}(\tfrac{r\pi}{P}), \mathrm{Cos}(\tfrac{q\pi}{P}))\mathrm{Cos}(\tfrac{jq\pi}{P}) \qquad (2.4.12)$$

$$K_{i,j} = \frac{\pi}{P} \sum_{k=o}^{P''} S_{k,j}\mathrm{Cos}(\tfrac{ik\pi}{P}) \qquad (2.4.13)$$

reduces the cost to

(a) $NP(P+N)$ operations for evaluating K

(b) N^3 operations for producing B from (2.4.11)

(c) $\frac{1}{3}N^3$ operations for solving the system (2.3.5) by Gauss-Elimination

In the paper of Delves [14] he reduced the operation count to $O(N^2 \mathrm{Log} N)$ operations overall, and we now show how this is done.

2.5 SETTING UP AND SOLVING THE SYSTEM

Evaluation of k from (2.4.12) and (2.4.13) can be achieved in about $NP(P+N)$ operations.

If we set $P=N$ we can identify (2.4.5) as representing a discrete Fourier Cosine Transform of the function $k(\mathrm{Cos}\phi, \mathrm{Cos}\phi)$. This transform can be carried out using the FFT (Fast Fourier Transform) technique which produces K in $O(N^2 \mathrm{Log}\ N)$ operations. Also the FFT procedure can be used

to evaluate (2.4.2) in O(NlogN) operations, where this count is almost a factor of N better than the standard Galerkin order count. The choice P<N is not sensible, because it guarantees very large errors in the matrix and the solution.

If the kernel and the driving term g each have awkward features which make them hard to integrate in that way, it is sensible to set P>N. So the FFT procedure yields a (PXP) matrix for K (and then B) which can be truncated and using only the leading (NXN) submatrix and this makes a cancel in the solution vector $\underline{a}$ which converges more rapidly than $\underline{g}$ or K.

Evaluation of B:

The direct use of (2.4.11) takes $O(N^3)$ operations, however, two one dimensional chebyshev series each of O(N) term can be multiplied together in O(NlogN) operation using the FFT procedure.

The algorithm is applicable to (2.4.11) which represents the explicit result of multiplying the Chebyshev expansion of k(x, y) see (2.4.6), by that of $(1 - y^2)^{\frac{1}{2}}$ which is a function of one variable, so that (see(2.4.11)) we can multiply the double series for K a row at a time, taking successively i = O(1)N and yielding an overall operations count of $O(N^2 \log N)$ as shown in [16] .

Hence the (N+1)X(N+1) Galerkin matrix equations can be set up in $O(N^2 \log N)$ operations.

A complete error analysis and comparative timings can be found in [14] which shows that first the method will converge very rapidly provided that the kernel k(x, y) and the driving term g(x) are smooth, and second that it is possible to provide cheaply computable error estimates which take into account both truncation errors (those of N terms in the

series) and quadrature error due to (2.4.2), (2.4.5). This paper also shows that there exists a simple iterative scheme for solving the equation (2.3.5) with an overall $O(N^2)$ operations count.

2.6 ERROR ESTIMATES

The error in the numerical solution obtained by the Fast Galerkin method has three components:

(1) The truncation error which stems from cutting off the exact expansion of the exact solution

$$f(x) = \sum_{j=0}^{\infty}{}' \; b_j \, T_j(x) \quad ; \quad -1 \leq x \leq 1 \tag{2.6.1}$$

at the Nth term. The truncated expansion is given by

$$f_N(x) = \sum_{j=0}^{N} a_j \, T_j(x) \quad ; \quad -1 \leq x \leq 1$$

(2) The discretization or quadrature error due to the numerical estimation of the matrix B and vector $\underline{g}$, equation (2.3.5).

(3) There are in principle errors arising from the solution of the linear system (2.3.5).

The computed solution $f_N(x)$ has error $e_N(x)$ that is

$$|e_N(x)| = |f(x) - f_N(x)| \leq \sum_{j=0}^{N}{}' |a_j - b_j| + \sum_{j=N+1}^{\infty} |b_j|$$

$$\leq \sum_{j=0}^{N} |\bar{a}_j - b_j| + \sum_{j=N+1}^{\infty} |b_j| + \sum_{j=0}^{N} |a_j - \bar{a}_j|$$

$$= E_1 + E_2 + E_3 \tag{2.6.2}$$

where $\bar{a}_j$; j=0(1)N are the exact Chebyshev coefficients which satisfy the equation (2.3.5).

I. Truncation error estimates

Since g(x) and k(x,y) are bounded functions on [-1,1] then (see [14]) $B_{i,j}$, g_i satisfy bounds of the form

$$|g_i| \leq C_g\, i^{-p} \quad ; \quad i \geq 1 \tag{2.6.3}$$

$$|B_{i,j}| \leq C_B\, i^{-q}\, j^{-r} \quad ; \quad i,j \geq 1$$

where C_g, C_B are some positive constants and p, q, r are values which depend on the differentiability of g(x) and k(x,y).

Delves [14] gave the following estimates for the truncation error $E_1 + E_2$:

A posteriori: $$E_1 + E_2 \sim E_2 \sim \frac{C_b}{s-1} N^{-(s-1)} \sim Na_N \tag{2.6.4a}$$

A priori: $$E_1 + E_2 \sim \frac{C_b}{s-1} N^{-(s-1)} \sim NC_d \max\{g_N, K_{N,1}\} \tag{2.6.4b}$$

where C_b is some known constant (independent of i, and all i > 0) and s = min {q, p}. The bound (2.6.4a) is standard; the bound (2.6.4b) contains the unknown C_d but gives a useful way of estimating the required value of N before the bulk of the calculation need be performed.

II. Quadrature error estimates

We in fact solve not the system (2.3.5) but the perturbed system

$$(D+B+\delta B)\,(\underline{a}+\delta\underline{a}) = \underline{g} + \delta\underline{g} \tag{2.6.5}$$

where $\delta\underline{a} = \bar{\underline{a}} - \underline{a}$; in any norm

$$||\delta \underline{a}|| \leq Q \{||\delta \underline{g}|| - ||\delta B|| ||\underline{a}||\} \Big| (1 + Q ||\delta B||) \qquad (2.6.6)$$

where $Q = ||(D+B)^{-1}||$.

Delves [14] estimates the quantities $||\delta \underline{g}||$, $||\delta B||$ and Q as:

$$\delta g_r = \sum_{j=1}^{\infty} (g_{2pj-r} + g_{2pj+r}) \quad , r \leq p$$

(p is the number of points in the quadrature rule).

Then we estimate $|\delta g_r| \sim |g_p|$, which can be used as the basis of a numerical estimate of $||\delta \underline{g}||$. Similarly $||\delta B||$ can be estimated as

$$||\delta B||_\infty \sim \frac{4}{3} \sum_{j=1}^{p} |K_{pj}|$$

and a rough estimate for Q is given by

$Q \sim ||\underline{g}|| \Big| ||\underline{a}||$. Hence equation (2.6.6) gives an estimate for the quadrature error E_3.

Throughout this work we do not include the error estimates in our results. For a complete error analysis and details on how the method returns the estimate error to the user, the reader is referred to [14, 17].

Finally, the total cost for the Fast Galerkin method is $O(N^2 \log N)$, and it is this lower count, together with the rapid convergence and error estimate, which makes the scheme attractive.

CHAPTER 3

NUMERICAL INVERSION OF THE LAPLACE TRANSFORM

3.1 INTRODUCTION

Consider the following integral equation

$$\int_0^\infty k(s,t)\,\phi(t)\,dt = \psi(s) \qquad 0 \leq s < \infty \qquad (3.1.1)$$

Where the kernel k depends only on the product of the variables s,t. This general class of integral equation includes the Laplace transform, the Fourier sine and cosine transforms and many other integral equations of importance in physics.

We consider in this chapter the numerical solution of the Laplace inversion where the main difficulty in applying Laplace-transform techniques is the determination of the original function $\phi(r)$ from its transform.

$$\int_0^\infty e^{-pr}\,\phi(r)\,dr = \psi(p) \qquad 0 \leq p < \infty \qquad (3.1.2)$$

In many cases, analytical methods fail and numerical methods must be used. The best known numerical methods for the inversion of the Laplace transform are based on the expansion of the original function in a series of orthogonal functions. A special case of one of these methods is discussed in [52]. The principal reason for the importance of orthogonal exponential functions is that only real values of $\psi(p)$ are required for calculating the coefficients of the series expansion of $\phi(r)$. However the computation of $\phi(r)$ from values of $\psi(p)$ on the real axis is numerically unstable, Bellman and Kalaba [8].

Therefore, if a high degree of accuracy is desired, the calculation must be carried out in multiple precision or methods must be used which determine the original function from values of the transform in the complex-plane as described in [26] in which one of his approach is to derive from an expansion of the transform $\psi(p)$ in some region of the p-plane a rational approximant to $\psi(p)$ which can then be inverted using rational functions; this approach has been used by Longman [39, 40] who generated a rational function approximations to $\psi(p)$ by means of a non-linear sequence to sequence transformation from the Maclaurin expansion of the transform $\psi(p)$. These rational functions are then inverted analytically to yield a sequence of approximations $\phi_n(r)$ to the inverse $\phi(r)$ of $\psi(p)$. Where [52] based his method on the approximation of the transform by truncated series of orthogonal functions, related to the Jacobi or Laguerre polynomials, i.e.

$$\psi(p) = p^{-a} \sum_{k=o}^{\infty} C_k P_k^{(\alpha,\beta)}(1-bp^{-1}) \qquad (3.1.3)$$

where $P_r^{(\alpha,\beta)}$ is the Jacobi polynomial of degree r and α, β, a and b are free parameters and the coefficient C_k is given by:

$$C_k = h_k^{-1} \int_{-1}^{1} (1-x)^{\alpha} (1+x)^{\beta} P_k^{(\alpha,\beta)}(x)\, y(x)\, dx \qquad (3.1.4)$$

where

$$y(x) = \left(\frac{b}{1-x}\right)^a \psi\left(\frac{b}{1-x}\right) \quad , \qquad (3.1.5)$$

$$h_k = \int_{-1}^{1} (1-x)^{\alpha} (1+x)^{\beta} \left[P_k^{(\alpha,\beta)}(x) \right]^2 dx \qquad (3.1.6)$$

Inverting the series (3.1.3) term by term, we obtain

$$\phi(r) = \frac{r^{a-1}}{\Gamma(a)} \sum_{k=0}^{\infty} C_k \frac{(\alpha+1)_k}{k!} U_k \left[\frac{br}{2}\right] \qquad ** \qquad (3.1.7)$$

where $U_k(x)$ is a polynomial of degree k

$$U_k(x) = {}_2F_2 \left[\begin{matrix} -k, & k+\alpha+\beta+1 \\ \alpha+1, & a \end{matrix} ; x \right] = \sum_{r=0}^{\infty} \frac{(-k)_r (k+\alpha+\beta+1)_r}{(\alpha+1)_r (a)_r} \frac{x^r}{r!} \qquad (3.1.8)$$

then with $\alpha = \beta = -\frac{1}{2}$, the computation can be simplified. Indeed, the truncated series formulas for the inversion of the Laplace transform is

$$\phi(r) \simeq \frac{r^{a-1}}{\Gamma(a)} \sum_{k=0}^{N}{}' C_k U_k \left(\frac{br}{2}\right) \text{ where } U_k(x) = {}_2F_2 \left[\begin{matrix} -k, & k \\ \frac{1}{2}, & a \end{matrix} ; x \right] \qquad (3.1.9)$$

and the coefficients C_k can be calculated by Clenshaw's method for the computation of Chebyshev coefficients.

$$C_k \simeq \frac{2}{N} \sum_{t=0}^{N}{}'' y\left(\cos \frac{t\pi}{N}\right) \cos \left(\frac{tk\pi}{N}\right) \qquad (3.1.10)$$

and the polynomials $U_k(x)$ can be generated from the recurrence relation:

$$U_n(x) = (A+Bx) U_{n-1}(x) + (C+Dx) U_{n-2}(x) + E U_{n-3}(x)$$

n = 3(1)....

** $(a)_r = \frac{\Gamma(a+r)}{\Gamma(a)}$

where A, B, C and D,E are relationsgiven in terms of the free parameters α, β, a and b [52] .

The method requires the values of the Laplace transform for non-equidistant real values of the argument.

Another approach [65] based on expand the original inverse as a Fourier expansion of $\phi(r)$.

Let $\phi_N(t)$ denote the Fourier-expansion of $\phi(t)$ interms of $Q_k(t)$, so that

$$\phi_N(t) = \sum_{k=0}^{N} Q_k(t)\ C_k \qquad (3.1.11)$$

where

$$C_k = \int_0^{\infty} Q_k(r)\ \phi_N(r)\, W(r)\ dr \qquad (3.1.12)$$

and $Q_k(t)$ is the orthonormal functions such that

$$\int_0^{\infty} Q_m(r)\ Q_n(r)\ W(r)\ dr = \delta_{mn} \qquad (3.1.13)$$

Where δ_{mn} is the kronecker delta, W(r) is the non-negative weight function over [0, ∞). Equation (3.1.11) may be regarded as a formula for numerical inversion of the Laplace transform, only if C_k can be conveniently expressed in terms of $\psi(t_i)$, where t_i are points in the p-Plane. Also equation (3.1.11) is an inversion formula if $Q_k(r)$ and W(r) are linear combinations of exponential functions.

We describe in this chapter a new method of approximating the exact inverse of the Laplace transform (3.1.2) as the solution of an integral

equation of the first kind. We based the method on mapping the variables onto the finite space $[0, 1]$ and applying the Galerkin techniques on the resulting integral equation using a truncated Chebyshev expansion for the exact mapped inverse. Then using the Augmented Galerkin method [1] for solving the linear system of Galerkin equations, where the matrix elements of the system (Laplace-matrix) are calculated analytically.

3.2 ILL-CONDITIONING AND NATURE OF THE PROBLEM

The general Fredholm integral equation of the first kind is defined by

$$\int_a^b k(x,y)\, f(y)\, dy = g(x) \qquad a \leq x \leq b \qquad (3.2.1)$$

which in operator form is

$$K f = g \qquad (3.2.2)$$

the solution of such equations has been studied by many authors (Phillips [50]; Tikonov [60]; Baker, Fox, Mayers and Wright [4]) Turchin, Kozlov and Malkevich [62]; Hanson, [28] and a useful review is given by Miller ([45]); and recently B.A. Lewis ([36]) and a nice paper by Babolian and Delves ([1]).

Their ill-conditioned nature may be illustrated in a rather simple manner as follows: the problem has much in common with that of solving a system of algebraic equations

$$B \underline{a} = \underline{b}$$

in which the matrix B is severely ill-conditioned (or even singular) the most troublesome feature from a computational point of view is that the problem is ill-posed that is the solution f of (3.2.2) does not depend continuously

on the driving term g, that is, a very small perturbation on the driving term g can give rise to arbitrarily large perturbation in the solution f. To show this according to the Riemann-Lebesgue theorem, that is for any integrable kernel, note that

$$R_n(x) = \int_a^b k(x,y) \operatorname{Sin}(ny)\, dy \to 0 \qquad (3.2.3)$$

as $n \to \infty$

and so it is possible to make $R_n(x)$ arbitrarily small by choosing a sufficiently large value of n that is for given ε however small,

$$||R_n(x)||_\infty = \underset{a \leq x \leq b}{\text{maximum}} |R_n(x)| \leq \varepsilon \qquad (3.2.4)$$

Now suppose that equation (3.2.2) has a unique solution f, and let g be given a small perturbation, say

$$\delta g(x) = C\, R_n(x)$$

where C is an arbitrary constant.

The corresponding change in f is $\delta f = C \operatorname{Sin}(nx)$, it is clear by assigning a value to C and choosing a sufficiently large n we can make the ratio

$$||\delta f||_\infty \Big/ ||\delta g||_\infty$$

as large as we like.

In section (3.3)we shall apply the Fast Galerkin method for the solution of the integral equation of the first kind (3.1.2) and solve the obtained system of linear equations using the ill-posed solver of [1]

3.3 THE GALERKIN ALGORITHM

The transform function $\psi(p)$ is infinite at $p = o$ for many commonly occurring problems (e.g. $\phi(r)$ = constant), so in order to weaken the necessary condition for a finite transform let us multiply both sides of the transform (3.1.2) by the parameter p which leads to

$$p \int_o^\infty e^{-pr}\, \phi(r)\, dr = h(p) \qquad (3.3.1)$$

We consider the exact inverse $\phi(r)$ of the transform (3.3.1) such that the following conditions hold:

(a) The Laplace transform h(p) exists (and is known) in some region $[o, m]$ of the semi infinite space $[o, \infty)$.

(b) The integral $\int_o^\infty |\phi(r)|^2\, dr$ exists.

Now let $p = mq$, then from (3.3.1) we have

$$\int_o^\infty mq\, e^{-mqr}\, \phi(r)\, dr = h(mq) \qquad o \leq q \leq 1 \qquad (3.3.2)$$

Let $z = e^{-mr} \rightarrow mdr = -z^{-1}dz$ hence

$$\int_0^1 qz^{q-1} f(z)\, dz = h(mq) \qquad 0 \leq q \leq 1 \tag{3.3.3}$$

where

$$f(z) = \phi(-\text{Log}_e\, |z|/m)$$

Let $q = \frac{s+1}{2}$, $z = \frac{t+1}{2}$; $s,t \in [-1,1]$

then from (3.3.3) we have:

$$\int_{-1}^{1} \left(\frac{s+1}{2}\right) \left(\frac{t+1}{2}\right)^{\left(\frac{s+1}{2}\right)-1} f\left(\frac{t+1}{2}\right) \frac{dt}{2} = g(s) \; ; \quad -1 \leq s \leq 1 \tag{3.3.4}$$

where $g(s) = h(m\frac{s+1}{2})$.

By weighted Galerkin technique using the truncated Chebyshev expansion for $f(\frac{t+1}{2})$:

$$f\left(\frac{t+1}{2}\right) \simeq f_N\left(\frac{t+1}{2}\right) = \sum_{j=0}^{N} a_j\, T_j(t) \qquad -1 \leq t \leq 1 \tag{3.3.5}$$

we end up with the linear system

$$B\, \underline{a} = \underline{g} \tag{3.3.6}$$

where

$$B_{i,j} = \int_{-1}^{1} \left(\frac{s+1}{2}\right) \frac{T_i(s)}{\sqrt{1-s^2}} \int_{-1}^{1} \left(\frac{t+1}{2}\right)^{\left(\frac{s+1}{2}\right)-1} T_j(t)\, \frac{dt}{2}\, ds \qquad (3.3.7)$$

$$g_i = \int_{-1}^{1} g(s) \frac{T_i(s)}{\sqrt{1-s^2}}\, ds \qquad (3.3.8)$$

The integrals appearing in equations (3.3.7), (3.3.8) must now be evaluated. For the integral in equation (3.3.8) the technique of chapter (2) can be applied by relating g_i to the Chebyshev coefficients in expansion of $g(s)$, and then evaluating these coefficients numerically using Fast Fourier Transform technique.

For the integral in equation (3.3.7) we evaluate it analytically as follows.

Let us change the variables as:

$x = \frac{s+1}{2}$; $y = \frac{t+1}{2}$; $x, y \in [0,1]$ then

$$B_{i,j} = 2 \int_{0}^{1} x T_i^*(x)\, W(x) \int_{0}^{1} y^{x-1} T_j^*(y)\, dy\, dx \qquad 0 \leq x \leq 1 \qquad (3.3.9)$$

where T^* is the shifted Chebyshev polynomial on $[0,1]$

$$W(x) = 2^{-1} (x-x^2)^{-\frac{1}{2}} \qquad (3.3.10)$$

Lemma (3.3.1)

$$\int_{0}^{1} \frac{W(x)}{mx+r}\, dx = \frac{\pi}{2\sqrt{mr+r^2}} \quad , r > 0$$

Proof:

$$\int_{-1}^{1} (1-x^2)^{-\frac{1}{2}} (mx+m+2r)^{-1}\, dx = \frac{-1}{\sqrt{mr+r^2}} \tan^{-1}\left[\sqrt{\frac{r}{m+r}}\ \sqrt{\frac{1-x}{1+x}}\right]_{-1}^{1}$$

$$= \frac{\pi}{2\sqrt{mr+r^2}}$$

Lemma (3.3.2)

$$\int_0^1 \frac{xW(x)}{mx+r}\,dx = \frac{\pi}{2m}\begin{cases} 1 & r=0 \\ 1-\frac{r}{\sqrt{mr+r^2}} & r>0 \end{cases}$$

Proof: (see Appendix A)

Lemma (3.3.3)

$$\int_0^1 y^x\, T^*_{j-1}(y)\,dy = \tfrac{1}{4}\left[\frac{2}{j(2-j)} - \frac{1}{j}\int_0^1 x\,y^{x-1}\,T^*_j(y)\,dy + \frac{x}{j-2}\int_0^1 y^{x-1}\,T^*_{j-2}(y)\,dy\right]$$

Proof: (see Appendix A)

$$T^*_j(y) = 2\,(2y-1)\,T^*_{j-1}(y) - T^*_{j-2}(y) \qquad j \geq 2 \qquad (3.3.11)$$

$$B_{i,j} = 2(4)\int_0^1 xT^*_i(x)\,W(x)\int_0^1 y^x\,T^*_{j-1}(y)\,dydx - 2\,B_{i,j-1} - B_{i,j-2}$$

Using Lemma (3.3.3) then

$$B_{i,j} = \frac{4}{j(2-j)}\int_0^1 x\,T^*_i(x)\,W(x)\,dx - \frac{1}{4j}\,B_{i+1,j} - \frac{1}{2j}\,B_{i,j} - \frac{1}{4j}\,B_{i-1,j}$$

$$+ \frac{1}{4(j-2)}\,B_{i+1,j-2} + \frac{1}{2(j-2)}\,B_{i,j-2} + \frac{1}{4(j-2)}\,B_{i-1,j-2} - 2\,B_{i,j-1}$$

$$- B_{i,j-2} \qquad (i \geq 1, \quad j \geq 3)$$

since

$$\int_0^1 x\, T_i^*(x)\, W(x)\, dx = \pi \begin{Bmatrix} \frac{1}{4} & i = 0 \\ \frac{1}{8} & i = 1 \\ 0 & i \geq 2 \end{Bmatrix} \qquad (*)$$

we have:

$$B_{i,j} = \frac{4\pi}{j(2-j)} \begin{bmatrix} \frac{1}{8} & i = 1 \\ 0 & i > 1 \end{bmatrix} - \frac{1}{4j} B_{i+1,j} + \frac{1}{4(j-2)} B_{i+1,j-2} - 2 B_{i,j-1}$$

$$+ \left(\frac{1}{2(j-2)} - 1\right) B_{i,j-2} - \frac{1}{2j} B_{i,j} - \frac{1}{4j} B_{i-1,j} + \frac{1}{4(j-2)} B_{i-1,j-2}$$

$$(i \geq 1, \; j \geq 3)$$

$$B_{i+1,j} = \frac{16\pi}{(2-j)} \begin{bmatrix} \frac{1}{8} & i = 1 \\ 0 & i > 1 \end{bmatrix} - (4j+2) B_{i,j} - 8j B_{i,j-1} + \left(\frac{2j}{j-2} - 4j\right) B_{i,j-2} - B_{i-1,j}$$

$$+ \frac{j}{j-2} B_{i-1,j-2} + \frac{j}{j-2} B_{i+1,j-2} \quad (i \geq 1, \; j \geq 3) \qquad (3.3.12)$$

$$B_{i,o} = 2 \int_0^1 W(x)\, T_i^*(x)\, dx = \pi \begin{bmatrix} 1 & i = 0 \\ 0 & i > 0 \end{bmatrix} \qquad (3.3.13)$$

$$B_{o,j} = 8 \int_0^1 x\, W(x) \int_0^1 y^x\, T_{j-1}^*(y)\, dy\, dx - 2 B_{o,j-1} - B_{o,j-2}$$

Using Lemma (3.3.3) and (*) we have:

$$B_{1,j} = \frac{2\pi}{(2-j)} - (2j+1)\, B_{o,j} - 4j\ B_{o,j-1} + \left(\frac{j}{j-2} - 2j\right) B_{o,j-2} + \frac{j}{j-2}\, B_{1,j-2}$$

$$j \geq 3 \qquad (3.3.14)$$

Let

$$T_j^*(y) = \sum_{r=o}^{j} C_{j,r}\, y^r \qquad \text{then} \qquad (3.3.15)$$

$$B_{o,j} = \sum_{r=o}^{j} C_{j,r}\, A_{o,r} \quad ; \qquad (3.3.16)$$

where

$$A_{o,r} = 2\int_o^1 x\, W(x) \int_o^1 y^{x+r-1}\, dy\, dx = 2\int_o^1 \frac{x\, W(x)}{x+r}\, dx$$

From Lemma (3.3.2) then

$$A_{o,r} = \pi\left\{ \begin{array}{l} 1 \qquad\quad ;\ r = o \\ \\ 1 - \frac{r}{\sqrt{r+r^2}} ;\ r > o \end{array} \right\} \qquad (3.3.17)$$

Substituting (3.3.15) in (3.3.11) we have

$$C_{j,r} = 4\, C_{j-1,r-1} - 2\, C_{j-1,r} - C_{j-2,r} \qquad (3.3.18)$$

$$r=o(1)j \qquad r=1(1)j \qquad r=o(1)j-1 \qquad r=o(1)j-2$$

$$B_{i,1} = 2\int_o^1 x\, T_i^*(x)\, W(x) \int_o^1 y^{x-1}\, T_1^*(y)\, dy\, dx\ .$$

$$B_{i,1} = 4\int_0^1 \frac{xT_i^*(x)\,W(x)}{x+1}\,dx - B_{i,0} \quad , \quad i \geq 2 \qquad (**)$$

$$B_{i,1} = 16\int_0^1 \frac{x^2W(x)}{x+1}\,T_{i-1}^*(x)\,dx - 2B_{i-1,1} - B_{i-2,1} \; ; \; B_{i,0} = 0 \quad (i \geq 2)$$

$$B_{i,1} = 16\int_0^1 x\,W(x)\,T_{i-1}^*(x)\,dx - 6B_{i-1,1} - B_{i-2,1}$$

$$B_{i,1} = 2\pi\begin{cases} 1 & i = 2 \\ 0 & i > 2 \end{cases} - 6B_{i-1,1} - B_{i-2,1} \quad ; \quad i \geq 2 \qquad (3.3.19)$$

$$B_{i,2} = 16\int_0^1 \frac{xT_i^*(x)\,W(x)}{x+2}\,dx - 8\int_0^1 \frac{xT_i^*(x)\,W(x)}{x+1}\,dx - 2B_{i,1} - B_{i,0} \; ; \; i \geq 2$$

From (**) then

$$B_{i,2} = 16\int_0^1 \frac{xT_i^*(x)\,W(x)}{x+2}\,dx - 4B_{i,1} \quad ; \quad i \geq 2$$

Define

$$V_i = 16\int_0^1 \frac{xT_i^*(x)\,W(x)}{x+2}\,dx$$

Using (3.3.11) we have

$$V_i = 64\pi\begin{cases} \frac{1}{8} & i = 2 \\ 0 & i > 2 \end{cases} - 10V_{i-1} - V_{i-2} \quad ; \; i \geq 2 \qquad (3.3.20)$$

where

$$V_o = 8\pi(1 - \frac{2}{\sqrt{6}}) \quad ; \quad V_1 = 16\pi(\frac{5}{\sqrt{6}} - 2) \tag{3.3.20a}$$

$$B_{i,2} = V_i - 4\, B_{i,1} \quad ; \quad i \geq 2 \tag{3.3.21}$$

These equations constitute a set of recurrence relations from which the matrix B can be computed.

We can summarize the algorithm for evaluating the integral of equation (3.3.7) after scaling the matrix by a constant factor $\frac{4}{\pi 2}$ as follows:

Step (1)

$$B_{i,o} = \frac{4}{\pi} \begin{bmatrix} 1 & ; & i = o \\ o & ; & i > o \end{bmatrix}$$

Step (2)

$$B_{1,1} = \frac{4}{\pi}(\frac{6}{\sqrt{2}} - 4) \; ; \qquad B_{0,1} = \frac{4}{\pi}(1 - \sqrt{2}) \; ;$$

$$B_{i,1} = \frac{8}{\pi} \begin{bmatrix} 1 & i = 2 \\ o & i \neq 2 \end{bmatrix} - 6\, B_{i-1,1} - B_{i-2,1} \quad ; \quad i \geq 2$$

Step (3)

$$V_o = \frac{4}{\pi}(8 - \frac{16}{\sqrt{6}}) \; ; \; V_1 = \frac{4}{\pi}(\frac{80}{\sqrt{6}} - 32) \quad ;$$

$$B_{1,2} = \frac{4}{\pi}(\frac{80}{\sqrt{6}} - \frac{24}{\sqrt{2}} - 16) \quad ;$$

$$V_i = \frac{4}{\pi} \begin{bmatrix} 8 & ; & i = 2 \\ o & ; & i > 2 \end{bmatrix} - 10V_{i-1} - V_{i-2} \quad ; \quad i \geq 2$$

$$B_{i,2} = V_i - 4\, B_{i,1} \quad ; \quad i \geq 2$$

Step (4)

$$C_{o,o} = 1,\ C_{1,o} = -1 \quad ;\quad C_{1,1} = 2 \quad ;$$

$$A_{o,r} = \frac{4}{\pi} \begin{cases} 1 & ; r = o \\ 1 - \frac{r}{\sqrt{r+r^2}} & ; r > o \end{cases}$$

$$C_{j,r} = 4\, C_{j-1,r-1} - 2\, C_{j-1,r} - C_{j-2,r} \quad ; \quad j \geq 2$$

$$B_{o,j} = \sum_{r=o}^{j} C_{j,r}\, A_{o,r} \quad ; \quad j \geq 2$$

Step (5)

$$B_{1,j} = \frac{4}{\pi}\left(\frac{2}{2-j}\right) - (2j+1)\, B_{o,j} - 4j\, B_{o,j-1}$$

$$+ \left(\frac{j}{j-2} - 2j\right) B_{o,j-2} + \frac{j}{j-2}\, B_{1,j-2} \quad ; j \geq 3$$

Step (6)

$$B_{i+1,j} = \frac{64}{\pi(2-j)} \begin{pmatrix} \frac{1}{8} & i = 1 \\ 0 & i > 1 \end{pmatrix} - (4j+2)\, B_{i,j} - 8j\, B_{i,j-1}$$

$$+ \frac{j}{j-2}\, B_{i+1,j-2} - B_{i-1,j} + \left(\frac{2j}{j-2} - 4j\right) B_{i,j-2}$$

$$+ \frac{j}{j-2}\, B_{i-1,j-2} \qquad (i \geq 1, \quad j \geq 3)$$

3.4 EFFECTIVENESS OF THE METHOD

The choice of any method should be made in terms of: accuracy, cost and stability which we shall consider.

(I) COST: The cost of the method has three components: (a) The time taken in evaluating the integral in equation using the described algorithm and the driving term equation. (b) The manipulations needed to set up the relevant algebraic equations. (c) The solution of the system represented by equation (3.3.6).

We would usually classify (a) as "useful" time, (b) and (c) as "overhead operations", the cost for evaluating the Laplace matrix using a set of recurrence relations as described in the algorithm is of order $O(N^2)$ and the driving term can be evaluated in $O(N\text{Log } N)$ operation using "Fast Fourier Transform" technique described in chapter (2) and $O(N^2\text{Log } N)$ operations for set up equations, an explicit use is made of the structure of the equations to yield an $O(N^2)$ iterative technique for their solution. Hence, in terms of cost wise this looks an attractive algorithm, but it has instability problems in practice which we describe.

(II) STABILITY: The stability of the method depend on the stability of the set of recurrence relations used in evaluating the Laplace matrix and also on the stability of the ill-posed solver which we used for solving the linear system (see next section). We would like to give a brief discussion about forward and backward recurrence relations (RR) before we study the stability of the recurrence relations used in the algorithm.

Considerable study has been made of relations with constant coefficients of the general form:

$$a_n y(i) + a_{n-1} y(i+1) + a_{n-2} y(i+2) + \ldots\ldots + a_o y(i+n) = 0 \qquad (3.4.1)$$

which are traditionally described as unstable when one or more of the roots $\lambda_1, \lambda_2, \ldots\ldots, \lambda_n$ of the characteristic equation

$$a_o \lambda^n + a_1 \lambda^{n-1} + \ldots\ldots + a_n = 0 \qquad (3.4.2)$$

be outside the unit circle. Thus if (3.4.1) is solved recursively in the direction of increasing i, any propagated error will be a linear combination of the fundamental solution $\lambda^i_1, \lambda^i_2, \ldots\ldots \lambda^i_n$ of the homogeneous form of (3.4.1), provided the roots of (3.4.2) are distinct. If one or more of these roots exceeds unity in magnitude some or all of the propagated error will eventually increase unboundedly in absolute value, causing the number of correct decimal places in the solution to diminish as n increases. Computation of the solution of the homogeneous form of (3.4.1) by forward recurrence is usually impractical owing to strong instability; on the other hand, backward application of the homogeneous form of (3.4.1) provides a stable way of computing the solution, since rounding error grow no faster than the wanted solution, as a rule [49].

A considerable number of algorithms have been used extensively in the computation of the solutions of difference equations by backward recurrence when forward recurrence is strong unstable [49], which we do so.

It proceeds as follows: for a suitably chosen large integer N, a "trial" solution $y_r^{(N)}$ of the homogeneous form of (3.4.1) is generated recursively for r = N, N-1,....., 0 beginning with $y_N^{(N)} = 0$ and $y_{N-1}^{(N)} = \varepsilon$ then the solution say f_r is found by multiplying $y_r^{(N)}$ by a normalizing factor μ_N. For example if the value f_o is given then $\mu_N = f_o / y_o^{(N)}$, more

generally if f_r satisfies a condition of the form

$$m_0 f_0 + m_1 f_1 + \ldots\ldots m_{N-1} f_{N-1} = 1$$

with given coefficients m_r, then

$$\mu_N = 1 \Big/ (m_0 y_0^{(N)} + m_1 y_1^{(N)} + \ldots + m_{N-1} y_{N-1}^{(N)})$$

where the value of N can be estimated by testing with a higher value and comparing results.

Hence let us study the stability of the recurrence relations in our algorithm starting with the recurrence relations of steps (2) and (3), where the related homogeneous recurrence relations are:

$$B_{i,1} + 6 B_{i-1,1} + B_{i-2,1} = 0 \qquad ; \qquad i > 2$$

$$V_i + 10 V_{i-1} + V_{i-2} = 0 \qquad ; \qquad i > 2 \tag{3.4.3}$$

respectively. Since the relations are of one dimensional form then we seek a solution of the form,

$$B_{i,1} = \alpha^i$$

$$V_i = \beta^i \tag{3.4.4}$$

substituting from (3.4.4) in (3.4.3) and solving the resulting quadratic equation we get

$$\alpha_{1,2} = -3 \pm 2\sqrt{2}$$

$$\beta_{1,2} = -5 \pm 2\sqrt{6}$$

which shows that both the recurrence relations are unstable in the direction of increasing i. We achieved a good accuracy for computing the second and third columns of the Laplace matrix by running the recurrence relations in steps (2), (3) backward. ($\varepsilon = 0.0$)

The recurrence relation in step (4) has two problems, the first provided by the recurrence relations of computing the Chebyshev coefficients:

$$C_{j,r} = 4\, C_{j-1,r-1} - 2\, C_{j-1,r} - C_{j-2,r}$$

which is also unstable. It is possible in principle to avoid round-off problems since all coefficients are integer, but instead overflow problems result when we use an integer arithmetic. Secondlywe lose a large number of significant figures during the summation (3.3.16)

$$\sum_{r=o}^{j} C_{j,r}\, A_{o,r}$$

Since the sign of the Chebyshev coefficients alternate, it is possible to overcome this problem by using arbitrary precision arithmetic [55]. We have ascertained experimentally the numerical instability of the recurrence relations in steps (5), (6). For the recurrence relation of step (5) we achieved a good accuracy by using the arbitrary precision (up to 72 digit) arithmetic. However, for the recurrence relation of step (6) we could not achieve a good accuracy because the recurrence relation in step (6) is very unstable.

Hence, we essentially abandoned this approach altogether, and evaluated the Laplace matrix by using an alternative algorithm which we now describe.

An alternative algorithm for evaluating the integral:

$$B_{i,j} = 2 \int_0^1 x\, T_i^*(x)\, W(x)\, dx \int_0^1 y^{x-1}\, T_j^*(y)\, dy$$

Using (3.3.15) we have:

$$B_{i,j} = \sum_{r=0}^{j} C_{j,r}\, A_{i,r} \qquad (3.4.5)$$

where $C_{j,r}$ as (3.3.18),

$$A_{i,r} = 2 \int_0^1 \frac{x\, T_i^*(x)\, W(x)}{x+r}\, dx \qquad (3.4.6)$$

Using (3.3.11) we have

$$A_{i,r} = 8 \int_0^1 x\, T_{i-1}^*(x)\, W(x)\, dx - (4r+2)\, A_{i-1,r} - A_{i-2,r}$$

from (*) gives the Recurrence formula.

$$A_{i,r} = \frac{4}{\pi} \begin{cases} 1 & i = 2 \\ 0 & i > 2 \end{cases} - (4r+2)\, A_{i-1,r} - A_{i-2,r} \quad ; \quad i \geq 2 \qquad (3.4.7)$$

$$A_{o,r} = \frac{4}{\pi} \begin{cases} 1 & r = 0 \\ 1 - \frac{r}{\sqrt{r+r^2}} & r > 0 \end{cases} \quad ; \qquad (3.4.8)$$

$$A_{1,r} = \frac{4}{\pi} - (2r+1)\, A_{o,r} \qquad (3.4.9)$$

where $B_{i,o}$, $B_{i,1}$ and the initial values as before.

So we can summarize the algorithm as follows:

Step (1)

$$B_{i,o} = \frac{4}{\pi} \begin{cases} 1 & i = 0 \\ 0 & i > 0 \end{cases}$$

Step (2)

$$B_{i,1} = \frac{4}{\pi} \begin{cases} 1 & i = 2 \\ 0 & i > 2 \end{cases} - 6\, B_{i-1,1} - B_{i-2,1} \quad ; \quad i \geq 2$$

where $B_{0,1}$; $B_{1,1}$ as shown in step (2) page (38)

Step (3)

$$B_{i,j} = \sum_{r=0}^{j} C_{j,r}\, A_{i,r} \quad ; \quad j \geq 2$$

$$\cdot\ C_{j,r} = 4C_{j-1,r-1} - 2\, C_{j-1,r} - C_{j-2,r}$$

$$\cdot\ A_{o,r} = \frac{4}{\pi} \begin{cases} 1 & r = 0 \\ 1 - \frac{r}{\sqrt{r^2+r}} & r > 0 \end{cases} \quad ; A_{1,r} = \frac{4}{\pi} - (2r+1)\, A_{o,r}$$

$$\cdot\ A_{i,r} = \frac{4}{\pi} \begin{cases} 1 & i = 2 \\ 0 & i > 2 \end{cases} - (4r+2)\, A_{i-1,r} - A_{i-2,r} \qquad i \geq 2$$

where the initial values $B_{0,1}$, $B_{1,1}$; $C_{o,o} = 1$, $C_{1,o} = -1$, $C_{1,1} = 2$ are given.

We achieved a good accuracy for computing the matrix elements of the Laplace matrix i.e. the integral in equation (3.3.7), where we displayed the matrix elements in appendix (B) which shows that the integral of equation (3.3.7) converges to zero as $i \to \infty$ for fixed j while the previous algorithm diverges because of the instability of its (RR) for large N. We carried

out the calculation on ICL 1906S computer using Arbitrary precision arithmetic in Algol 68 [55]; we used in fact 72 decimal digit of precision. The cost of such precision is of course very high; but we note that the matrix B is independent of the parameter m in (3.3.3) and hence B need be computed only once and then stored for later use.

(III) ACCURACY: No simple method for inverting Laplace transform has been successful for all types of functions $\psi(p)$; we therefore consider first the class of functions for which we might hope the current method could be successful.

First of course, we are applying this scheme with the assumption that the equation under consideration has square integrable solution in the mapped interval of integration which implies that the expanded solution (3.3.5) is convergent with respect to the weight function $W(x) = (1-x^2)^{-\frac{1}{2}}$ in $L_2(W)$, hence we expect:

(a) A rapidly convergent (good results) with smooth square integrable solution in the mapped range of integration.

(b) Bad results slow (even divergence) convergent with non-square integrable solution.

The condition that the solution be "smooth" in the mapped region excludes, for example, transforms which have an infinite number of oscillations on $[0, \infty)$, since these oscillations are all mapped into $[-1, 1]$ yielding a decidedly non-smooth mapped solution. However, transforms $\psi(p)$ which have (at most) a simple pole at $p = 0$ are handled by the algorithm, since this pole is suppressed by the factor p introduced into (3.3.1).

Hence, generally the accuracy of the method depends on the assumption that the equation under consideration must have a square integrable defining function (smooth) in the mapped region and also on the exact estimate of the parameters of the ill-posed solver (see next section).

3.5 SOLUTION OF THE LINEAR SYSTEM

Since our problem is an ill-posed problem, that is the matrix involved in the linear system (3.3.6) is in general quite ill-conditioned, any attempt to solve the system directly will lead to numerical nonsense for large value of N; the solution of the system (3.3.6) defines an approximate solution f_N of the equation (3.3.4) which for large N oscillates wildly in the interval of the solution. We chose the ill-posed solver described in [1] for solving the linear system (3.3.6), which exercises direct control on an expansion of the solution where the Galerkin equations for the chosen expansion are augmented by a set of regularity conditions on the solution, and the augmented (and overdetermined) equations solved in L_1 and L_∞ norms.

Since the Chebyshev polynomials are orthogonal with respect to the weight function W(t) on $[-1, 1]$, then the assumption that (3.3.4) has an $L_2(W)$ solution implies that the representation

$$f(t) = \sum_{i=0}^{\infty} a_i T_i(t) \qquad (3.5.1)$$

is convergent in $L_2(W)$ and that hence

$$\sum_{i=0}^{\infty} a_i^2 < \infty \quad ; \quad |a_N| = O(N^{-\frac{1}{2}})$$

Hence there exist constants $C_f > 0$, $R > \frac{1}{2}$ such that

$$|a_i| \leq C_f \hat{i}^{-R} \quad ; \quad i = 0,1.... \tag{3.5.2}$$

where

$$\hat{i} = i \qquad i > 0$$
$$= 1 \qquad i = 0$$

Babolian and Delves [1] regularize the solution with given constants C_f, R by imposing (3.5.2) as a constraint on the computed solution vector $\underline{a}$, that is, rather solving (3.3.6) [1] solve the problem:

$$\text{minimize} ||B\underline{a} - \underline{g}||$$

subject to

$$|a_i| \leq C_f \hat{i}^{-R} \quad , \quad i = 0(1)N$$

Estimation of the parameters C_f and R:-

With our experience with the ill-posed solver for the numerical inversion of the Laplace transforms we found that the ill-posed solver is sensitive to the values C_f, R. However we achieved a good result for equations with smooth square integrable functions, We state here the three methods suggested by the authors for estimating these parameters.

Method (1)

Choose R arbitrary and compute C_f from the relation $C_f = \lambda(||g||/||B||)$ where λ must set heuristically.

Method (2)

Since for a wide class of kernels matrix B in (3.3.6) is well-conditioned for small values of N, hence we choose N_o small and solve the system (3.3.6) directly to obtain $\underline{a}$. Now taking the logarithms of the constraints equation (3.5.2) we obtain

$$\text{Log}|a_i| \lesssim \text{Log } C_f - R\text{Log}\hat{i} \qquad i = O(1)N_o$$

We may estimate the parameters from a least-square solution to the over-determined system:

$$\text{Log}|a_i| = \text{Log } C_f - R\text{Log}i \quad , \quad i = 1(1)N_o$$

where the sequence $\{|a_i|\}$ should be decreasing to ensure positive R, and finally multiply the computed C_f by a safety factor (μ).

Method (3)

Assign a value to R and calculate C_f from:

$$C_f = \max\ (|a_i|\hat{i}^R)$$

$$o \lesssim i \lesssim N_o$$

Finally with our experience with the ill-posed solver suggests it has satisfactory stability properties. That is as N increases, the error reduces initially and then finally stabilizes; increasing N further neither gains nor loses accuracy. However, this comment assumes that suitable values of C_f,R have been chosen, for poor choice of those values, low accuracy results were obtained, and we often had more difficulty in finding suitable values. In our experience with the method, and additional numerical results (not included in this chapter), "method (3)" above was most successful, but it would seem that further work in this area might well be worthwhile.

3.6 NUMERICAL EXAMPLES

Finally we present some numerical examples to demonstrate how the method works in general. The examples displayed here have been chosen to show how essential are some of the restrictions made on the functions under consideration and to demonstrate the way in which changes in the exact inverse affect the quality of the results.

The computed error shown in this work is in terms of MS-ERROR (Mean square error) as:

$$e_N(s_i) = \phi(s_i) - \phi_N(s_i)$$

$$\bar{e}_N = \sum_{i=1}^{N} e_N^2(s_i) \; ; \; \text{MS-ERROR} = \frac{1}{N}\sqrt{\bar{e}_N}$$

Problem (3.6.1)

The equation to be solved is

$$p \int_0^{\infty} e^{-pr} \phi(r)\, dr = h(p) \quad ; \quad o \leq p \leq m$$

where

$$\phi(r) = \text{Exp}(-ar) \quad ; \quad a > o \quad ; \quad r \in [o, \infty)$$

$$h(p) = \frac{p}{a+p} \quad ; \quad p \in [o, m]$$

$$\text{Exact inverse} = \left(\frac{t+1}{2}\right)^{a/m} \quad ; \quad t \in [-1, 1]$$

$$g(s) = \frac{m(s+1)}{m(s+1)+2a} \quad ; \quad s \in [-1, 1]$$

Example (3.6.1.1)

$a = 1$; $m = 1$

The computed results (MS-ERROR) of this example displayed in table (3.6.1) with the methods described in section (3.5). Note that with a=m (>1) we achieved the same results as in table (3.6.1)

TABLE (3.6.1)

Computed MS-ERROR

for example (3.6.1.1)

N	Method (1)				Method (3) $N_o = 4$			
	R = 3		R = 11		R = 5		R = 11	
	$\lambda = 2$		$\lambda = 6$		$C_f = \frac{1}{2}$		$C_f = \frac{1}{2}$	
3	6.516	$X10^{-2}$	2.657	$X10^{-3}$	2.858	$X10^{-3}$	3.165	$X10^{-3}$
4	9.758	-2	8.768	-5	2.693	-3	4.579	-5
5	1.162	-1	1.902	-5	1.212	-4	5.858	-5
6	1.333	-1	1.952	-6	1.938	-4	2.132	-6
7	1.459	-1	1.239	-6	2.311	-4	3.841	-8
8	1.569	-1	8.090	-8	4.175	-5	5.745	-8
9	1.632	-1	1.214	-7	5.395	-5	9.796	-8
10	-		6.073	-8	6.664	-5	5.293	-8
11	-		5.301	-8	3.426	-5	3.835	-8
12	-		4.133	-8	3.351	-5	4.084	-8

Example (3.6.1.2)

$a = 2,\ m = 1$

The computed results (MS-ERROR) shown in table (3.6.2), also $a = 2m$ gives the same accuracy.

Example (3.6.1.3)

$a = 1.0,\ m = 2.0$; $(a/m) \neq$ integer,

hence the mapped exact inverse is:

$$\sqrt{\frac{t+1}{2}} \quad ; \quad t \in [-1,1]$$

which we expect a very slow convergence because of the square-root behaviour of the exact inverse (as shown in table (3.6.3)).

TABLE (3.6.2)

Computed MS-ERROR

for example (3.6.1.2)

$N_o = 4$

N	Method (1) R = 11 λ = 6		Method (2) μ = 24 R=5 $C_f=\frac{1}{2}$		Method (3) R = 5 $C_f=2^{R-3}$		R = 8 $C_f=2^{R-3}$		R = 11 $C_f=2^{R-3}$	
3	3.013	-2 X10	4.244	-2 X10	4.244	-2 X10	4.244	-2 X10	4.244	-2 X10
4	3.038	-2	1.410	-2	3.224	-4	3.224	-2	3.224	-4
5	3.039	-2	3.948	-3	1.163	-3	1.376	-4	1.544	-5
6	3.040	-2	1.478	-3	4.615	-4	2.289	-4	3.044	-5
7	3.040	-2	7.637	-4	2.451	-4	7.187	-6	5.593	-6
8	3.041	-2	1.199	-3	3.464	-4	1.563	-5	1.067	-5
9	3.041	-2	7.666	-4	2.588	-4	1.214	-1	1.214	-1
10	3.041	-2	8.235	-4	2.745	-4	-		-	
11	3.082	-2	8.160	-4	2.720	-4	-		-	
12	3.338	-2	8.051	-4	2.683	-4	-		-	

TABLE (3.6.3)

Computed MS-ERROR

for example (3.6.1.3)

N	Method (3) $N_o = 4$						Method (1)			
	R = 4		R = 5		R = 6		R = 5		R = 11	
	$C_f=(0.43)(3)^R$		$C_f=(0.43)(3)^R$		$C_f=(0.43)(3)^R$		$\lambda = 4.0$		$\lambda = 6.0$	
3	1.998	$X10^{-1}$	1.998	$X10^{-1}$	1.998	$X10^{-1}$	3.203	$X10^{-2}$	3.770	$X10^{-2}$
4	7.007	-2	7.007	-2	7.007	-2	2.596	-2	3.189	-2
5	7.412	-2	8.115	-2	8.693	-2	2.607	-2	3.150	-2
6	2.898	-2	3.637	-2	4.748	-2	2.657	-2	3.148	-2
7	2.888	-2	2.758	-2	4.133	-2	2.699	-2	3.149	-2
8	3.478	-2	2.418	-2	3.923	-2	1.704	-2	3.150	-2
9	3.954	-2	2.252	-2	3.827	-2	1.740	-2	3.150	-2
10	4.317	-2	2.163	-2	3.776	-2	2.836	-2	3.151	-2
11	4.583	-2	2.110	-2	3.747	-2	3.978	-2	3.151	-2
12	4.776	-2	2.077	-2	3.730	-2	5.053	-2	3.152	-2

Problem (3.6.2)

The problem to be solved is:

$$p \int_0^\infty e^{-pr}\, \phi(r)\, dr = h(p) \quad ; \quad 0 \leq p \leq m$$

where

$$\phi(r) = r^{-1}(e^{-\alpha r} - e^{-\beta r}) \quad ; \quad \alpha \neq \beta$$

$$= f(t) = \frac{m}{\text{Log}_e \left|\frac{t+1}{2}\right|} \left(\left(\frac{t+1}{2}\right)^{\beta/m} - \left(\frac{t+1}{2}\right)^{\alpha/m} \right) \quad ; \; t \in [-1,1] \; ;$$

$$h(p) = p \, \text{Log}_e \left|\frac{p+\beta}{p+\alpha}\right| = g(s) = \frac{m}{2}(s+1) \, \text{Log}_e \left|\frac{m(s+1)+2\beta}{m(s+1)+2\alpha}\right|$$

$$p \in [0,m] \quad ; \quad s \in [-1,1]$$

TABLE (3.6.4)

Computed MS-ERROR

for problem 'example' (3.6.2)

m=1 $N_0 = 4$

N	$\alpha = 1, \beta = 2$			$\alpha = 2, \beta = 5$	arbitrary choice
	Method (2)	Method (3)		Method (3)	
	$\mu = 6$ R=3, $C_f = \frac{1}{2}$	R = 5 $C_f = (0.02)(3)^R$	R = 9 $C_f = (0.02)(3)^R$	R = 7 $C_f = (0.1047)(3)^R$	R = 8 $C_f = 1000$
3	6.884 X10^{-2}	5.832 X10^{-2}	6.884 X10^{-2}	7.2415 X10^{-2}	7.2415 X10^{-2}
4	3.583 -2	7.982 -3	7.982 -3	4.6625 -3	7.4825 -3
5	1.782 -2	2.374 -3	1.251 -3	8.8578 -3	9.0747 -3
6	8.250 -3	1.584 -3	1.561 -3	4.3318 -3	3.7489 -4
7	5.921 -3	1.297 -3	1.117 -3	4.4098 -3	1.9058 -4
8	1.119 -2	4.624 -4	1.517 -1	4.4100 -3	4.5218 -4
9	5.882 -3	4.448 -4	1.517 -1	4.4167 -3	2.2339 -1
10	6.781 -3	5.418 -4	-	4.4188 -3	-
11	8.708 -3	6.384 -4	-	4.4199 -3	-
12	7.642 -3	7.165 -4	-	2.2339 -1	-

Problem (3.6.3)

$$p\int_0^{\infty} e^{-pr}\,\phi(r)\,dr = h(p) \quad ; \quad o \le p \le m \qquad (3.6.1)$$

where: $h(p) = \frac{p}{p^2+p+1}$

(p of the L.H.S. is that introduced in (3.3.1));

$\phi(r) = \frac{2}{\sqrt{3}} e^{-\frac{r}{2}} \operatorname{Sin}(\frac{\sqrt{3}}{2}r)$ which belong to $L_2[o, \infty)$.

Where the exact inverse in the mapped interval $[-1, 1]$ is also square integrable function

$$\phi(r) = f(t) = \frac{2}{\sqrt{3}} \operatorname{Exp}(\frac{1}{2m} \log|\frac{t+1}{2}|) \operatorname{Sin}(\frac{-\sqrt{3}}{2m} \log|\frac{t+1}{2}|) \; ; \; t \in [-1, 1]$$

We expect a very slow convergence for this problem because of the infinite number of oscillations in the solution on $[o, \infty)$, since these oscillations are all mapped into $[-1, 1]$ yielding a decidedly non-smooth mapped solution, as shown in table (3.6.5) with $\alpha = 0.0$, $N = 10$, where α is a constant parameter. We have:

$$p\int_0^{\infty} e^{-(p-\alpha)r}\,\theta(r)\,dr = h(p) \quad ; \quad \theta(r) = e^{-\alpha r}\,\phi(r)$$

let $p-\alpha = q$; $-\alpha \le q \le m - \alpha$ then we have

$$\int_0^{\infty} e^{-qr}\,\theta(r)\,dr = h(q+\alpha)(q+\alpha)^{-1} \qquad (3.6.2)$$

where

$$h(q+\alpha) = ((q+\alpha)^2 + (q+\alpha) + 1)^{-1}(q+\alpha)$$

Hence instead of solving (3.6.1) we solve (3.6.2) where $\theta(r)$ is a square integrable function in the mapped interval, which is smoother than $\phi(r)$, we obtain the original solution from

$\phi(r) = e^{\alpha r}\,\theta(r)$, table (3.6.5) with $\alpha = 0.8$, $N = 10$ shows a better approximation to the exact inverse.

Problem (3.6.4)

The problem to be solved is

$$p\int_0^{\infty} e^{-pr}\,\phi(r)\,dr = h(p) \qquad 0 \leq p \leq m$$

where

$h(p) = \text{Exp}\,(-p)$; $\phi(r) = H(r-1)$.

H denoting the Heaviside unit function that is:

$$H(r-1) = \begin{cases} 0 & ; \quad r < 1.0 \\ \frac{1}{2} & ; \quad r = 1.0 \\ 1 & ; \quad r > 1.0 \end{cases}$$

$$= f(t-1) = \begin{cases} 0 & ; \quad t < 2e^{-m} - 1 \\ \frac{1}{2} & ; \quad t = 2e^{-m} - 1 \\ 1 & ; \quad t > 2e^{-m} - 1 \end{cases} \qquad ; \quad t \in [-1,1]$$

The function $\phi(r)$ possesses a discontinuity at $r=1.0$, (and at $t = 2e^{-m} - 1$ on $[-1,1]$). We expect a very slow convergence if we solve the problem straightforward, but if we use the same technique as described in problem (3.6.3) we gain a satisfactory accuracy as shown in table (3.6.6b) while straightforward solution shows a very slow convergence as shown in table (3.6.6a).

TABLE (3.6.5)

Computed results for problem (3.6.3)

Using Method (3) of Section (3.5).

α=0.0 we set R=3 ; C_f=0.29538 but for α=0.8 we set R=6 ; $C_f=(0.53)(3)^R$

$N_0 = 4$

	α = 0.8 ; N = 10						α=0.0 ;N=10	
r	Exact $\phi(r)$		Computed $\theta(r)$		Computed $\phi_N(r)$		Computed $\phi_N(r)$	
1	5.335071	$X10^{-1}$	2.40825	$X10^{-1}$	5.35966	$X10^{-1}$	2.52080	$X10^{-1}$
2	4.19279	-1	8.34946	-2	4.13551	-1	1.75177	-1
3	1.33243	-1	9,82855	-3	1.08341	-1	1.46816	-1
4	-4.95298	-2	-8.71115	-4	-2.13706	-2	1.36373	-1
5	-8.79424	-2	-1.02095	-3	-5.57422	-2	1.32530	-1
6	-5.08923	-2	-4.75068	-4	-5.77257	-2	1.31116	-1
7	-7.64371	-3	-1.88329	-4	-5.09292	-2	1.30596	-1
8	1.271509	-2	-7.09858	-5	-4.27221	-2	1.30405	-1
9	1.28047	-2	-2.62010	-5	-3.50944	-2	1.30334	-1
10	5.38548	-3	-9.50630	-6	-2.83378	-2	1.30308	-1

TABLE (3.6.6a)

Computed results for problem (3.6.4)

Using Method (3) of Section (3.5) with R=5 ; C_f=4.86

α = 0.0 ; N = 9 ; N_0 = 4					
r	Exact	Computed $\phi_N(r)$	r	Exact	Computed $\phi_N(r)$
0.1	0.0	-3.924515 X10^{-2}	1.1	1.0	6.894168 X10^{-1}
0.2	0.0	-8.250561 -2	1.2	1.0	8.047141 -1
0.3	0.0	-6.448603 -2	1.3	1.0	8.918664 -1
0.4	0.0	-6.952970 -2	1.4	1.0	9.531676 -1
0.5	0.0	-6.828635 -2	1.5	1.0	9.929666 -1
0.6	0.0	-2.213843 -2	1.6	1.0	1.016223 +0
0.7	0.0	7.977687 -2	1.7	1.0	1.027617 +0
0.8	0.0	2.240937 -1	1.8	1.0	1.031096 +0
0.9	0.0	3.874392 -1	1.9	1.0	1.029736 +0
1.0	0.5	5.476987 -1	2.0	1.0	1.025773 +0

TABLE (3.6.6b)

Computed results for problem (3.6.4)

Using Method (3) of Section (3.5) with R=4 ; C_f=41.842012725

α = 0.8 ; N = 9 ; N_0 = 4					
r	Exact	Computed $\phi_N(r)$	r	Exact	Computed $\phi_N(r)$
0.1	0.0	6.320459 X10^{-2}	1.1	1.0	7.199864 X10^{-1}
0.2	0.0	7.245875 -2	1.2	1.0	8.799887 -1
0.3	0.0	-1.550782 -2	1.3	1.0	1.003926 +0
0.4	0.0	-1.104226 -1	1.4	1.0	1.087749 +0
0.5	0.0	-1.492069 -1	1.5	1.0	1.132604 +0
0.6	0.0	-1.127365 -1	1.6	1.0	1.143496 +0
0.7	0.0	-7.466148 -3	1.7	1.0	1.127743 +0
0.8	0.0	1.494862 -1	1.8	1.0	1.093539 +0
0.9	0.0	3.371318 -1	1.9	1.0	1.048828 +0
1.0	0.5	5.338595 -1	2.0	1.0	1.000543 +0

Comparison with other methods

We compare our method with other methods which are described below.

(a) Problem (3.6.5): A problem in Viscoelasticity.

We consider here the numerical solution of the Laplace transform inversion of the function

$$\psi(p) = p^{-1}\exp(-p/(1+\delta p)^{\frac{1}{2}}) \; ; \; p \in [0, m]$$

which arises prominently in problems of pulse propagation in viscoelastic media. Here δ is a positive parameter; note that $\delta = 0$ gives the Heaviside unit function (problem (3.6.4)). The exact inverse of this problem is $\phi(r) \to 1$ as $r \to \infty$ in which we expect a good accuracy because of the smoothness of the exact inverse in the mapped interval.

We display our results with different values of (r) in tables (3.6.7-9) where we set the transformed interval to $[0, 1]$ that is (m=1), following in the last row of each table the results obtained by (Longman [40]) "LM", who used the Maclaurin expansion of a function $\psi(p)$ of the Laplace transform operator p, rational function approximations to $\psi(p)$ are generated by means of a new nonlinear sequence to sequence transformation. These rational functions are then inverted analytically to yield a sequence of approximations $\phi_N(r)$ to the inverse $\phi(r)$. Longman's results, as he stated, are given to the number of places of decimals (till eight) warranted by the degree of convergence. The convergence in the results is good, yielding 2-3 figures of accuracy for our method.

(b) Zakian and Littlewood [65] used a weighted least-squares approximation to the exact inverse, using Legendre polynomials by inverting the transform $\psi(p) = (p+1)^{-1}$ (see example (3.6.1.1)), using 15-decimal floating point arithmetic.

The numerical results are presented in table (3.6.10). Our results were obtained by using method (3) of section (3.5) with R = 11, C_f = 0.5 and N_0 = 4. The results of table (3.6.10) shows that the computed error of Zakian et al is increased as N increases while our computed error is decreased as N increases, but still our method is not superior to that of Zakian et al.

(c) Simon et al [57] calculated the inversion $\phi(r)$ over some interval $0 < r < r\ (max)$ given an explicit expression for $\psi(p)$. Then $\phi(r)$ can be approximately represented by

$$\phi(r) = \frac{\exp(ar)}{2r}\left[\text{Re}\ \psi(a) + 2\sum_{j=1}^{\infty}(-1)^j\ \psi\left(a + \frac{j\pi i}{r}\right)\right]$$

where the parameter 'a' is a number greater than the abscissa of any singularity of $\psi(p)$.

The numerical results presented in table (3.6.11) show the comparison of this method with our method for the transform $\psi(p) = (p^2+p+1)^{-1}$ (see problem (3.6.3)). The column marked NN indicates the number of terms used for the series transformation stated above. We solve the problem using the modification described in problem (3.6.3) using method (3) of section (3.5) with R = 6, $C_f = (0.53)\ (3)^R$ and N_0 = 4.

The results in table (3.6.11) show that the technique of Simon et al performed marginally better than our method.

Conclusion

There is no simple method for inverting Laplace transform which has been successful for all types of $\psi(p)$; some methods are successful for a particular function $\psi(p)$ but fail for other functions. We would like to mention here some remarks on the results obtained by our method.

(1) For square integrable (smooth) functions in the mapped interval we expect the method to work with a very rapidly convergent expansion which it did in example (3.6.1.1) with a=m. While we did not obtain a good accuracy when we set m=2, a=1 as in example (3.6.1.3) in which the exact inverse (non-smooth) in the mapped interval where we expect a slow convergence and the results, as shown in table (3.6.3), were indeed bad.

(2) In problem (3.6.5) we tried the method on an important problem in visco-elasticity with different δ(0.1, 0.5, 1.0) where we expect to get a good accuracy, but as shown in tables (3.6.7-9) the results for a particular value of (r) with different values of (N) it only settled down to about 2-3 significant figures, a comparison with the results of (Longman [40]) shows that the method did not work particularly well for this problem.

Hence from the results obtained and the comparison with the other different methods, it seems that the method is unsuccessful, even for square integrable functions (see for example problems (3.6.1.3), (3.6.5)). The results in tables (3.6.7-11) showed that the method is <u>not</u> superior to the other methods considered.

In general we can say it is a worthwhile attempt to construct a reliable method for the inversion of the Laplace transform. The problem with our method is that our mapping from the semi-infinite space $[0,\infty)$ to the finite space $[0, 1]$ is not a wholly satisfactory mapping. Its most unsatisfactory feature is the way in which the smoothness of the exact inverse in the mapped coordinates (which affects the rate of convergence expected) depends initially on the interval $[0, \overline{m}]$ over which the right hand side is assumed to be known (compare example 3.6.1.1 with a=1, m=1, and 3.6.1.3 with a=1, m=2). It would seem that further work in this area might well be worthwhile. We like to mention that also the difficulty we met for estimating the parameters C_f,R of the ill-posed solver; in [1] it was claimed that the results obtained were insensitive to these parameters, but this has been very far from our experience with the Laplace transform.

TABLE (3.6.7)

The computed results forproblem (3.6.5) with $\delta = 0.1$

Using Method (2) of Section (3.5) with R=3; C_f=4.2
$N_0 = 4$

$\delta = 0.1$

N	r=0.9	r=1.4	r= 1.9	r=2.4	r=2.9
3	0.2952303	0.5074115	0.6920145	0.8245496	0.9125024
4	0.7280290	0.7980660	0.8466989	0.8928830	0.9302520
5	0.4126560	0.8216475	0.9541834	0.9977782	1.0090524
6	0.4857554	0.8519203	0.9727749	1.0011713	1.0050095
7	0.4672528	0.8625233	0.9844455	1.0047672	1.0036439
8	0.3615866	0.8703771	0.9848880	1.0026916	1.0023690
9	0.4402151	0.8751125	0.9902401	1.0022430	1.0008755
10	0.4286342	0.8759346	0.9946708	1.0024023	0.9997975
11	0.4256299	0.8772646	0.9947661	1.0020596	1.0000294
12	0.4282447	0.8760694	0.9939611	1.0026942	0.9999502
13	0.4274081	0.8709588	1.0008845	1.0017003	0.9987199
14	0.4363679	0.8655848	0.9992457	1.0037097	0.9991084
15	0.4454988	0.8575710	1.0056525	1.0029593	0.9974353
16	0.4256683	0.8655528	1.0143513	0.9970883	0.9939145
17	0.4238757	0.8678255	1.0028325	1.0034426	0.9960778
LM	0.4175	0.890211	0.9902597	0.99945478	0.99997752

TABLE (3.6.8)

The computed results of problem (3.6.5) with $\delta = 0.5$

Using Method (3) of Section (3.5) with R=4; C_f=6.4
$N_0 = 4$
$\delta = 0.5$

N	r=0.9	r=1.4	r=1.9	r=2.4	r=2.9
3	0.2729017	0.4692297	0.6605347	0.8031376	0.8994054
4	0.6554698	0.7692293	0.8433814	0.8976704	0.9356796
5	0.5530341	0.7871973	0.8989787	0.9503800	0.9745003
6	0.5348616	0.7776247	0.8996438	0.9544070	0.9784827
7	0.5439892	0.7706502	0.8944892	0.9533385	0.9793690
8	0.5608906	0.7725605	0.8901168	0.9505740	0.9786495
9	0.5482404	0.7818190	0.8925024	0.9490444	0.9772838
10	0.5428246	0.7797423	0.8945157	0.9495910	0.9770642
11	0.5450661	0.7764917	0.8955542	0.9502802	0.9770298
12	0.5493717	0.7727498	0.8961966	0.9511705	0.9770848
13	0.5497815	0.7723863	0.8962304	0.9512718	0.9770976
14	0.5503047	0.7718897	0.8965379	0.9512676	0.9770298
15	0.5506191	0.7714610	0.8966627	0.9513472	0.9769828
16	0.5509457	0.7708074	0.8968666	0.9515246	0.9769009
17	0.5511615	0.7704142	0.8969408	0.9517583	0.9768182
LM	0.5507	0.77599	0.908416	0.9507248	0.9777283

TABLE (3.6.9)

The computed results for problem (3.6.5) with $\delta = 1.0$

Using Method (3) of Section (3.5) with R=4 ; C_f=40.59

$N_o = 4$

$\delta = 1.0$

N	r = 0.9	r = 1.4	r = 1.9	r = 2.4	r = 2.9
3	0.2732438	0.4543898	0.6461753	0.7926339	0.8925516
4	0.9633236	0.7212630	0.6792743	0.7388341	0.8166059
5	0.6919014	0.8572111	0.8774184	0.8955504	0.9212049
6	0.5592892	0.7962174	0.8815688	0.9162214	0.9399729
7	0.5048298	0.7605095	0.8858068	0.9271177	0.9471401
8	0.6241397	0.7405671	0.8575066	0.9176163	0.9472768
9	0.6528437	0.7436857	0.8503268	0.9160312	0.9480227
10	0.6579224	0.7601117	0.8437033	0.9125937	0.9480733
11	0.6388311	0.7779599	0.8393110	0.9088569	0.9480723
12	0.6160995	0.7956143	0.8372533	0.9044086	0.9476416
13	0.6149969	0.7964441	0.8372797	0.9041422	0.9475913
14	0.6108443	0.7997197	0.8358269	0.9039489	0.9479023
15	0.5647476	0.7873242	0.8637628	0.8999654	0.9422768
16	0.5691884	0.7799118	0.8705849	0.8993614	0.9403102
17	0.5768589	0.7690135	0.8758376	0.9012632	0.9386946
LM	0.6124	0.766354	0.856236	0.910691	0.944246

TABLE (3.6.10)

The Computed error = $|\phi(r) - \phi_N(r)|$

r	$\phi(r)$	N = 5		N = 10		N = 15	
		[65]	Our method	[65]	Our method	[65]	Our method
0.0	1.00	2×10^{-13}	6.3×10^{-4}	3×10^{-10}	9.1×10^{-7}	6×10^{-7}	1.2×10^{-7}
0.5	0.61	6 -14	6.0 -5	5 -11	1.3 -7	9 -8	1.5 -8
1.0	0.37	7 -14	1.9 -4	6 -11	8.0 -8	4 -8	9.1 -8
2.0	0.14	8 -14	1.1 -4	3 -11	6.2 -8	3 -8	9.9 -8
5.0	0.01	2 -13	3.0 -5	5 -11	1.8 -8	2 -9	7.1 -10

TABLE (3.6.11)

The computed error = $|\phi(r) - \phi_N(r)|$

r	$\phi(r)$	[57]	NN	N = 8, α = 0.8 Our method
1.0	0.533507	5×10^{-5}	13	2.4×10^{-3}
2.0	0.419280	2 -5	13	5.7 -3
3.0	0.133243	4 -6	13	2.4 -2
4.0	-0.049530	0.0	13	2.8 -2
5.0	-0.087942	2 -6	13	3.2 -2
6.0	-0.050892	1 -6	14	6.8 -3
7.0	-0.007644	5 -6	15	4.3 -2
8.0	0.012715	1 -6	15	5.5 -2
9.0	0.012805	1 -6	15	4.7 -2
10.0	0.005385	1 -6	16	3.3 -2

CHAPTER 4

THE NUMERICAL SOLUTION OF CAUCHY-TYPE SINGULAR INTEGRAL EQUATIONS

4.1 INTRODUCTION

Cauchy-type singular integral equations are often encountered in problems of mathematical physics and their mathematical properties have been well investigated (see [43]). For the numerical solution it is possible to reduce them to an equivalent Fredholm integral equation of the second kind, and solving the result by any numerical technique (see for example [32] [33]). Considered direct methods for the solution of singular integral equations, where, after separating the dominant parts, these equations may be expressed in the form:

$$A\,\phi(s) + \frac{B}{\pi}\int_a^b \frac{\phi(t)}{t-s}\,dt + \int_a^b k(s,t)\,\phi(t)\,dt = h(s) \qquad (4.1.1)$$

where $h(s)$ and $k(s,t)$ are known Hölder-condition functions in the interval $[a,b]$, and A, B are real constants. Two methods have been devised [22, 23; 34, 35]; they both set $[a, b]$ to the standard interval $[-1, 1]$ and approximate $\phi(s)$ by the truncated series of the form:

$$\phi(s) = W(s)\,\psi(s) \simeq W(s)\sum_{i=0}^{N-1} a_i\,P_i(s) \qquad (4.1.2)$$

where $W(s) = (1-s)^{\alpha}\,(1+s)^{\beta}$; $-1 < \alpha,\beta < 1$ is the weight function and accordingly $P_i(s)$ are the Jacobi polynomials denoted by $P_i^{(\alpha,\beta)}$. The methods are different in principle although they yield results of the same form. One of the methods [34] is based on expanding $\psi(s)$ as an infinite series of the form

$$\psi(s) = \sum_{i=0}^{\infty} b_i \, P_i^{(\alpha,\beta)}(s) \tag{4.1.3}$$

and using the orthogonality relation

$$\int_{-1}^{1} P_N^{(\alpha,\beta)}(s) \; P_M^{(\alpha,\beta)}(s) \; W(s) \, ds = \delta_{N,M} \, R_N \quad ; \tag{4.1.4}$$

$$R_N = \frac{2^{(\alpha+\beta+1)}}{2N + \alpha + \beta + 1} \cdot \frac{\Gamma(N+\alpha+1) \; \Gamma(N+\beta+1)}{N! \; \Gamma(N + \alpha + \beta + 1)} \tag{4.1.5}$$

The coefficients b_i are determined by truncating the obtained infinite system of linear equations and then an approximate system of equations is obtained for the determination of the coefficients b_i, i=o(1)N-1. The other method starts with an approximating polynomial of finite degree:

$$\psi(s) \sum_{i=0}^{N-1} C_i \, P_i^{(\alpha,\beta)}(s) \tag{4.1.6}$$

and by applying the quadrature formulae given in [22] [35] using the points S_i determined by $P_N^{(\alpha,\beta)}(S_i) = o$; i = 1(1)N, for $\alpha = \beta = \pm \frac{1}{2}$ to approximate values of $\psi(s)$. In the paper of S. Krenk [34] he follows the second method provided he gave a simple summation formulae for the determination of the coefficients C_i in (4.1.6) and special formulae are derived in terms of Chebyshev polynomials for the cases $(\alpha,\beta) = (\pm\frac{1}{2}, \pm\frac{1}{2})$ also he included special formulas for $\psi(1)$ and $\psi(-1)$.

These methods are clearly related both to each other, and to the Fast Galerkin expansion method considered in this thesis. Note however the weight factor W(s) which appears in the approximate solution (4.1.2). This

weight factor is introduced for numerical convenience. However, it forces the approximate solution to be zero at the end point of the interval, and hence will in general lead to only slow convergence of the numerical solution to the exact solution at interior points.

We seek to avoid this difficulty, and present a numerical solution for Cauchy-type singular integral equation of the form:

$$\phi(s) + \int_a^b \frac{\phi(t)}{t-s}\, dt = h(s) \qquad a \lesssim s \lesssim b \tag{4.1.7}$$

Using an expansion method, which a generalization of Fast Galerkin method for second kind integral equations, we consider Fredholm, Volterra and inverse-Volterra singular integral equations of Cauchy-type; the extension of the method to equations of the more general form (4.1.1) is trivial.

4.2 THE FINITE PART OF AN INFINITE INTEGRAL

The integral in (4.1.7) does not exist in the Riemann sense; for Fredholm equation (a,b fixed) we interpret it here in the usual Cauchy sense (see chapter (1), section (1.3)). However, the Fast Galerkin method as extended in [14] implemented as described in [17], treats Greens-function type operators (having kernels with a discontinuous derivative along the line t=s) as the sum of "Volterra" and "inverse-Volterra" operators.

For Volterra and inverse-Volterra equations with Cauchy kernel, the singularities appears at the end of the range of integration, and the principal value integral does not exist. We give a meaning to these

integral equations in terms of the Hadamard finite part integral. We state the definition given by Hadamard [27] for the integral

$$\int_a^s \frac{A(t)}{(s-t)}\,dt \quad , \quad a \leq s \leq b$$

where he gave the sign "⌜" to denote the finite part of an infinite integral.

Hadamard definition

The finite part of an infinite integral:

$$\int_a^s \frac{A(t)}{(s-t)}\,dt \qquad a \leq s \leq b$$

is defined by adding a term

$B(t)\ \mathrm{Log}_e\ |s-t|$ and taking the limit as $t \to s$.

That is:

$$\overline{\int_a^s \frac{A(z)}{(s-z)}\,dz} = \lim_{t \to s} \left[\int_a^t \frac{A(z)}{(s-z)}\,dz + B(t)\ \mathrm{Log}_e|s-t| \right]$$

where the function $A(z)$ is assumed to satisfy the Lipschitz condition, and $B(t)$ is a function satisfying the conditions:

(a) the limit must exist

(b) $B(t)$ has a continuous first derivative at least in the vicinity of $t=s$.

Note that the finite part of the integral will in general depend on the exact choice of the function $B(t)$. Here we make the consistent choice:

$B(t)$ = constant. The following results then hold:

Lemma (4.2.1)

(i) $$\overline{\int_a^s} \frac{dz}{(z-s)} = -\text{Log } |s-a| \; ; \quad a \lesssim s \lesssim b$$

(ii) $$\overline{\int_s^b} \frac{dz}{(z-s)} = \text{Log } |s-b| \; ; \quad a \lesssim s \lesssim b$$

Proof: (i) from Hadamard definition with $A(z) = 1$

We have:

$$\overline{\int_a^s} \frac{dz}{s-z} = \lim_{t \to s} \left[\int_a^t \frac{dz}{s-z} + B(t) \log |s-t| \right]$$

$$= \lim_{t \to s} \left[- \text{Log } |s-t| + \text{Log } |s-a| + B(t) \text{ Log } |s-t| \right]$$

We choose here to set $B(t) = +1$; then

$$\overline{\int_a^s} \frac{dz}{z-s} = - \text{Log } |s-a|$$

with the same processes (ii) follows.

We shall require within the formalism to carry out a change of variables; the Hadamard integral (of integral order) is not invariant under such a transformation (see [27] p.137-138).

Carrying out a linear transformation explicitly we find:

Lemma (4.2.2)

(a) $$\int_{-1}^{x} \frac{dy}{y-x} = -\operatorname{Log}\left|\frac{b-a}{2}(x+1)\right| \qquad -1 \leq x \leq 1$$

(b) $$\int_{x}^{1} \frac{dy}{y-x} = \operatorname{Log}\left|\frac{b-a}{2}(x-1)\right| \qquad -1 \leq x \leq 1$$

Proof: By mapping the variables in Lemma (4.2.1) onto the standard interval $[-1, 1]$ gives (a), (b) respectively i.e. (set $z = \frac{b-a}{2} y + \frac{b+a}{2}$; $s = \frac{b-a}{2} x + \frac{b+a}{2}$).

Lemma (4.2.3)

(a) $$\int_{-1}^{1} T_n(x)\, dx = \begin{cases} \frac{2}{1-n^2} & , n \text{ even} \\ 0 & , n \text{ odd} \end{cases}$$

(b) $$\int_{-1}^{1} T_n(x)\, T_m(x)\, dx = \begin{cases} \frac{1}{1-(n+m)^2} + \frac{1}{1-(n-m)^2} & \text{both } (n,m) \text{ even or odd} \\ 0 \quad ; \quad \text{otherwise} \end{cases}$$

(c) $$\int_{-1}^{1} T_n(x)\, U_{m-1}(x)\, dx = \begin{cases} \frac{-2m}{n^2 - m^2} & ; |n-m| \text{ odd} \\ 0 & ; \text{otherwise} \end{cases}$$

(d) $$\int_{-1}^{1} \frac{T_n(y)}{\sqrt{1-y^2}} \operatorname{Log}|x-y|\, dy = -\pi \begin{cases} \operatorname{Log} 2 & ; \quad n = 0 \\ \frac{1}{n} T_n(x) & ; \quad n > 0 \end{cases}$$

Where $T_n(x)$, $U_m(x)$ are the first and second kind of Chebyshev polynomials respectively.

4.3 THE FRAMEWORK

We describe the numerical solution of the Cauchy type singular integral equations over a finite interval $[a, b]$ of the form:

$$\text{(a)} \qquad \phi(s) + \int_a^b \frac{\phi(t)}{t-s}\, dt = h(s) \quad ; \quad a \leq s \leq b$$

$$\text{(b)} \qquad \Phi(s) + \int_a^s \frac{\phi(t)}{t-s}\, dt = h(s) \quad ; \quad a \leq s \leq b \qquad (4.3.1)$$

$$\text{(c)} \qquad \phi(s) + \int_s^b \frac{\phi(t)}{t-s}\, dt = h(s) \quad ; \quad a \leq s \leq b$$

Where (a), (b) and (c) known as Fredholm, Volterra and inverse-Volterra Cauchy type singular integral equations of second kind, these equations have important application, for example in aerodynamics.

In the next three sections, we shall consider the numerical solution of the three types of singular integral equations above. To use Fast Galerkin scheme for second kind integral equations [14], [16] we map the variables (t, s) onto the finite interval $[-1, 1]$ by setting

$$t = \frac{b-a}{2} y + \frac{b+a}{2}$$

$$x, y \quad [-1, 1]$$

$$s = \frac{b-a}{2} x + \frac{b+a}{2}$$

Substituting these in (4.3.1) we have:

$$(a^1)\quad f(x) + \mu \int_{-1}^{1} \frac{f(y)}{y-x}\, dy = g(x) \quad ; \quad -1 \le x \le 1$$

$$(b^1)\quad f(x) + \mu \int_{-1}^{x} \frac{f(y)}{y-x}\, dy = g(x) \quad ; \quad -1 \le x \le 1 \qquad (4.3.2)$$

$$(c^1)\quad f(x) + \mu \int_{x}^{1} \frac{f(y)}{y-x}\, dy = g(x) \quad ; \quad -1 \le x \le 1$$

$$\mu = \left(\frac{b-a}{2}\right)\left(\frac{2}{b-a}\right)$$

We approximate the assumed L_2- solution f by the truncated Chebyshev expansion for f(x):

$$f \simeq f_N = \sum_{i=o}^{N} a_i\, T_i(x) \qquad -1 \le x \le 1 \qquad (4.3.3)$$

By applying the weighted Galerkin method on the equations in (4.3.2) as described in chapter (2) we end up with the linear system of equations where the unknowns are the Chebyshev coefficients a_i:

$$(D + B)\, \underline{a} = \underline{g} \qquad (4.3.4)$$

where:

$$D_{i,j} = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\, T_j(x)\, dx = \pi \begin{cases} 1 & i = j = o \\ \frac{1}{2} & i = j > o \\ 0 & i \ne j \end{cases} \qquad (4.3.5)$$

$$g_i = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\, g(x)\, dx \qquad (4.3.6)$$

All the three equations in (4.3.2) have the same structure of the right hand side of the system (4.3.4) as in equation (4.3.6), hence for evaluating the integral in equation (4.3.6) we use the technique described in chapter (2).

Now for evaluating the matrix B in the system (4.3.4) that is the integrals:

$$B_{i,j}\,(\text{Fredholm}) = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\,dx \int_{-1}^{1} \frac{T_j(y)}{y-x}\,dy \;;\quad -1 \le x \le 1 \qquad (4.3.7a)$$

$$B_{i,j}\,(\text{Volterra}) = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\,dx \int_{-1}^{x} \frac{T_j(y)}{y-x}\,dy \;;\quad -1 \le x \le 1 \qquad (4.3.7b)$$

$$B_{i,j}\,(\text{inverse-Volterra}) = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\,dx \int_{x}^{1} \frac{T_j(y)}{y-x}\,dy \;;\quad -1 \le x \le 1 \qquad (4.3.7c)$$

It is essential for accuracy to use analytic methods. For simplicity let us call the integrals in equations (4.3.7a-c) as Fredholm matrix, Volterra matrix and inverse-Volterra matrix respectively. We shall produce recurrence relations to evaluate the Volterra and inverse-Volterra matrices.

We could calculate the Fredholm matrix as the sum of Volterra matrix and inverse-Volterra matrix, i.e.

Fredholm matrix = Volterra matrix + inverse-Volterra matrix

but it proves possible (and more accurate) to compute the Fredholm matrix directly.

4.4 FREDHOLM MATRIX

From equation (4.3.7a) we have

$$B_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\, dx \int_{-1}^{1} \frac{T_j(y)}{y-x}\, dy$$

where $\frac{4}{\pi^2}$ is a scaling factor.

Using Lemma (1.1) we can write the above integral as:

$$B_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} dx \int_{-1}^{1} dy\ \frac{T_i(x)\ T_j(y)}{y-x}\ W(x) \quad ; \quad W(x) = (1-x^2)^{-\frac{1}{2}}$$

$$= \frac{4}{\pi^2} \int_{-1}^{1} dy \int_{-1}^{1} dx\ \frac{T_i(x)\ T_j(y)}{y-x}\ W(x)$$

$$= -\frac{4}{\pi^2} \int_{-1}^{1} T_j(x)\, dx \int_{-1}^{1} \frac{T_i(y)}{y-x}\ W(y)\, dy$$

$$= -\frac{4}{\pi^2} \int_{-1}^{1} \left(\int_{-1}^{1} \frac{W(y)T_i(y)}{y-x}\, dy \right) T_j(x)\, dx$$

Since (see for example [61] p.180)

$$\int_{-1}^{1} \frac{W(y)T_i(y)}{y-x}\, dy = \pi \begin{cases} U_{i-1}(x) & ; \ i > 0 \\ 0 & ; \ i = 0 \end{cases} \qquad |x| < 1$$

Where $U_i(x)$ is the second kind of Chebyshev polynomials.

Hence:

$$B_{i,j} = -\frac{4}{\pi} \int_{-1}^{1} T_j(x) \begin{bmatrix} U_{i-1}(x) & i > 0 \\ 0 & i = 0 \end{bmatrix} dx$$

From Lemma (4.2.3c) we have:

$$B_{i,j} = \frac{8}{\pi} \left\{ \begin{array}{ll} \dfrac{i}{j^2 - i^2} & , \; |j - i| \text{ odd} \\ 0 & \text{otherwise} \end{array} \right\}$$

4.5 VOLTERRA MATRIX

We describe in this section the way to evaluate the integral in equation (4.3.7b) as a set of recurrence relations that is from (4.3.7b). We have:

$$B_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{-1}^{x} \frac{T_j(y)}{y-x}\, dy \quad ; \quad W(x) = (1-x^2)^{-\frac{1}{2}}$$

from the identity

$$T_j(y) = 2yT_{j-1}(y) - T_{j-2}(y) \qquad j \geq 2 \qquad (4.5.1)$$

then

$$B_{i,j} = \left(\frac{4}{\pi^2}\right) 2 \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{-1}^{x} T_{j-1}(y)\, dy + B_{i+1,\,j-1} + B_{i-1,\,j-1}$$

$$- B_{i,\,j-2} \quad (i \geq 1, \quad j \geq 2)$$

evaluating the integral value of

$$\int_{-1}^{x} T_{j-1}(y)\, dy = \tfrac{1}{2} \left\{ \frac{T_j(y)}{j} - \frac{T_{j-2}(y)}{j-2} \right\}_{-1}^{x} \qquad (4.5.2)$$

then we have

$$B_{i,j} = \left(\frac{4}{\pi^2}\right) \frac{1}{j} \int_{-1}^{1} W(x)\, T_i(x)\, T_j(x)\, dx - \left(\frac{4}{\pi^2}\right) \frac{1}{j-2} \int_{-1}^{1} W(x)\, T_i(x)\, T_{j-2}(x)\, dx$$

$$- \frac{4}{\pi^2} \frac{(-1)^j}{j} \int_{-1}^{1} W(x)\, T_i(x)\, dx + \frac{4}{\pi^2} \frac{(-1)^{j-2}}{j-2} \int_{-1}^{1} W(x)\, T_i(x)\, dx$$

$$+ B_{i+1,\, j-1} + B_{i-1,\, j-1} - B_{i,\, j-2} \quad (i \geq 1,\ j \geq 3)$$

Using the identity

$$\int_{-1}^{1} W(x)\, T_i(x)\, dx = \left\{ \begin{array}{ll} \pi & i = o \\ 0 & i > o \end{array} \right\} \qquad (4.5.3)$$

We end up with general recurrence relation

$$B_{i,j} = \frac{1}{j} D_{i,j} - \frac{1}{j-2} D_{i,j-2} + B_{i+1,j-1} + B_{i-1,j-1} - B_{i,j-2} \qquad (4.5.4)$$

$$(i \geq 1,\ j \geq 3)$$

where $D_{i,j}$ and $D_{i,j-2}$ are diagonial matrices as in (4.3.5), that is

$$D_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, T_j(x)\, dx = \frac{4}{\pi} \begin{cases} 1 & i = j = 0 \\ \frac{1}{2} & i = j > 0 \\ 0 & i \neq j \end{cases} \qquad (4.5.5a)$$

$$D_{i,j-2} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, T_{j-2}(x)\, dx = \frac{4}{\pi} \begin{cases} 1 & i = j-2 = 0 \\ \frac{1}{2} & i = j-2 > 0 \\ 0 & i \neq j-2 \end{cases} \qquad (4.5.5b)$$

$$B_{o,j} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, dx \int_{-1}^{x} \frac{T_j(y)}{y-x} dy$$

equations (4.5.1), (4.5.2) and (4.5.3) lead to the recurrence relation for computing the first row of the matrix, that is

$$B_{o,j} = \frac{8}{\pi} \left(\frac{(-1)^{j-2}}{j(j-2)} \right) + 2B_{1,j-1} - B_{o,\,j-2} \quad (j \geq 3) \qquad (4.5.6)$$

Now for computing the first column of Volterra matrix we have to use Hadamard definition for evaluating the finite part of the infinite integral.

$$B_{i,o} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{-1}^{x} \frac{dy}{y-x}$$

by applying Lemma (4.2.2) then we have:

$$B_{i,o} = \frac{4}{\pi^2} \int_{-1}^{1} \left(-\text{Log} \left|\frac{b-a}{2}(x+1)\right|\right) W(x)\, T_i(x)\, dx$$

$$= \frac{4}{\pi^2} \left(-\text{Log} \left|\frac{b-a}{2}\right|\right) \int_{-1}^{1} W(x)\, T_i(x)\, dx - \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, \text{Log}\,|x+1|\, dx$$

Using the identity (4.5.3) and Lemma (4.2.3d) we have:

$$B_{i,o} = \frac{4}{\pi^2}\left\{\begin{array}{ll} -\text{Log}\,|\frac{b-a}{2}|\pi & ; i = o \\ 0 & ; i > o \end{array}\right\} - \frac{4}{\pi^2}\left\{\begin{array}{ll} -\pi\text{Log}2 & i = o \\ -\frac{\pi}{i}T_i(-1) & i > o \end{array}\right\}$$

$$= \frac{4}{\pi}\left\{\begin{array}{ll} -\text{Log}\,|\frac{b-a}{2}| + \text{Log}\,2 & i = o \\ \frac{(-1)^i}{i} & i > 0 \end{array}\right\}$$

(where a, b is the finite interval in equations (4.3.1))

$$B_{i,o} = \frac{4}{\pi}\left\{\begin{array}{ll} \log\,|\frac{4}{b-a}| & i = o \\ \frac{(-1)^i}{i} & i > 0 \end{array}\right\} \qquad (4.5.7)$$

Now for calculating the second column of the Volterra matrix:

$$B_{i,1} = \frac{4}{\pi^2}\int_{-1}^{1} W(x)\,T_i(x)\,dx \int_{-1}^{x} \frac{y}{y-x}\,dy$$

by using equations (4.5.1), (4.5.3) we have:

$$B_{i,1} = \frac{1}{2}\,(B_{i+1,o} + B_{i-1,o}) + \frac{2}{\pi}\left\{\begin{array}{ll} 1 & i = 1 \\ 0 & i > 1 \end{array}\right\};\quad i \geq 1 \qquad (4.5.8)$$

the same with equations (4.5.1), (4.5.3) and the relation

$$x^2T_i(x) = \frac{1}{4}\,(T_{i+2}(x) + 2\,T_i(x) + T_{i-2}(x)) \qquad (4.5.9)$$

We can calculate the third column of Volterra matrix from the recurrence relation:

$$B_{i,2} = B_{i+1,1} + B_{i-1,1} - B_{i,o} + \frac{1}{\pi} \begin{cases} 1 & i = 2 \\ 0 & i \neq 2 \end{cases} ; \; i \geq 1 \qquad (4.5.10)$$

Hence we can summarize the algorithm for computing the elements of Volterra matrix (4.3.7b).

$$B_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\, dx \int_{-1}^{x} \frac{T_j(y)}{y-x}\, dy \quad \text{as follows}$$

Step (1) $$B_{i,o} = \frac{4}{\pi} \begin{cases} \text{Log} \left|\frac{4}{b-a}\right| ; & i = o \\ \frac{(-1)^i}{i} ; & i > o \end{cases}$$

Step (2)
$$B_{o,1} = \frac{4}{\pi} + B_{1,o},$$
$$B_{i,1} = \frac{1}{2} (B_{i+1,o} + B_{i-1,o}) + \frac{2}{\pi} \begin{cases} 1 & i = 1 \\ 0 & i > 1 \end{cases} ; \; i \geq 1$$

Step (3)
$$B_{o,2} = 2 B_{1,1} - B_{o,o} - \frac{2}{\pi} ;$$
$$B_{i,2} = B_{i+1,1} + B_{i-1,1} - B_{i,o} + \frac{1}{\pi} \begin{cases} 1 & i = 2 \\ 0 & i > 2 \end{cases} ; \; i \geq 1$$

Step (4) $$B_{o,j} = \frac{8}{\pi} \left[\frac{(-1)^{j-2}}{j(j-2)}\right] + 2B_{1,j-1} - B_{o,j-2} \quad ; \; j \geq 3$$

Step (5) $$B_{i,j} = \frac{1}{j} D_{i,j} - \frac{1}{(j-2)} D_{i,j-2} + B_{i+1,\,j-1}$$

$$+ B_{i-1,\,j-1} - B_{i,\,j-2} \quad (i \geq 1,\; j \geq 3)$$

where $D_{i,j}$ and $D_{i,\,j-2}$ can be computed from (4.5.5a-b) respectively.

4.6 INVERSE-VOLTERRA MATRIX

We follow the same procedures in the previous section for computing the elements of the matrix:

$$B_{i,j} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{x}^{1} \frac{T_j(y)}{y-x}\, dy \quad ; \quad W(x) = (1-x^2)^{-\frac{1}{2}}$$

From the equations (4.5.1), (4.5.2) and (4.5.3) we get the general recurrence relation

$$B_{i,j} = B_{i+1,\, j-1} + B_{i-1,\, j-1} - B_{i,\, j-2} - \frac{1}{j} D_{i,j} + \frac{1}{j-2} D_{i,j-2}$$

$$(i \geq 1, \quad j \geq 3) \qquad (4.6.1)$$

Where $D_{i,j}$ and $D_{i,\, j-2}$ as in (4.5.5a), (4.5.5b) respectively.

From equation (4.5.1) and the identity

$$\int_{x}^{1} T_{j-1}(y)\, dy = \frac{1}{2} \left[\frac{T_i(y)}{j} - \frac{T_{j-2}(y)}{j-2} \right]_{x}^{1} \qquad (4.6.2)$$

We can compute the first row of inverse-Volterra matrix from the recurrence relation:

$$B_{o,j} = 2B_{1,\, j-1} - B_{o,j-2} - \frac{8}{\pi} \left[\frac{1}{j(j-2)} \right] \qquad j \geq 3 \qquad (4.6.3)$$

By applying Hadamard definition on the infinite integral for computing the first column of inverse-Volterra matrix

$$B_{i,o} = \frac{4}{\pi^2} \int_{-1}^{1} W(x)\, T_i(x)\, dx \int_{x}^{1} \frac{dy}{y-x}$$

Using Lemma (4.2.2) and the identity (4.5.3) we end up with

$$B_{i,o} = -\frac{4}{\pi} \begin{cases} \text{Log}\left|\frac{4}{b-a}\right| & i = o \\ \frac{1}{i} & i > o \end{cases} \qquad (4.6.4)$$

(Where a, b are the finite interval in equation (4.3.1)).

From equations (4.5.1), (4.5.3) and the relation (4.5.9) we compute the second and the third columns of the inverse-Volterra matrix from the recurrence relations:

$$B_{i,1} = \frac{1}{2}\left(B_{i+1,o} + B_{i-1,o}\right) - \frac{2}{\pi} \begin{cases} 1 & i = 1 \\ o & i > 1 \end{cases} ; \quad i \geq 1 \qquad (4.6.5)$$

$$B_{i,2} = B_{i+1,1} + B_{i-1,1} - B_{i,o} - \frac{1}{\pi} \begin{cases} 1 & i = 2 \\ o & i \neq 2 \end{cases} ; \quad i \geq 1 \qquad (4.6.6)$$

We can summarize the algorithm for computing the inverse-Volterra matrix (4.3.7c) as follows:

Step (1) $$B_{i,o} = -\frac{4}{\pi} \begin{cases} \text{Log}\left|\frac{4}{b-a}\right| ; i = o \\ \frac{1}{i} \quad ; \quad i > o \end{cases}$$

Step (2) $$B_{o,1} = \frac{4}{\pi} + B_{1,o},$$

$$B_{i,1} = \frac{1}{2}\left(B_{i+1,o} + B_{i-1,o}\right) - \frac{2}{\pi} \begin{cases} 1 & i = 1 \\ 0 & i > 1 \end{cases} ; \; i \geq 1$$

Step (3) $B_{o,2} = 2B_{1,1} - B_{o,o} - \frac{2}{\pi}$,

$$B_{i,2} = B_{i+1,1} + B_{i-1,1} - B_{i,o} - \frac{1}{\pi}\begin{cases} 1 & i = 2 \\ 0 & i \neq 2 \end{cases} \; ; \quad i \geq 1$$

Step (4) $$B_{o,j} = 2B_{1,j-1} - B_{o,j-2} - \frac{8}{\pi}\left[\frac{1}{j(j-2)}\right] \; ; \quad (j \geq 3)$$

Step (5) $$B_{i,j} = B_{i+1,j-1} + B_{i-1,j-1} - B_{i,j-2} - \frac{1}{j} D_{i,j} + \frac{1}{j-2} D_{i,j-2}$$

$$(i \geq 1,\ j \geq 3)$$

where

$D_{i,j}$ and $D_{i,j-2}$

can be computed from (4.5.5a-b) respectively.

4.7 NUMERICAL EXAMPLES

We present here a number of numerical examples to demonstrate how the method works in general, the test examples were run on IBM 4341 computer. The examples displayed here have been chosen of two types:

- Equations whose exact solution have a very rapidly convergent Chebyshev expansion (smooth-solution);
- Equations whose exact solution have singularity near the finite interval of integration, where we expect the solution behaviour to be represented badly near the singular point as in problem 2.

Problem (4.7.1)

Exact solution : $f(x) = x$

(1.a) Volterra-type

$$f(x) + \int_{-1}^{x} \frac{f(y)}{y-x}\, dy = g(x) \qquad -1 \leq x \leq 1$$

$$g(x) = x + \int_{-1}^{x} dy + x \int_{-1}^{x} \frac{dy}{y-x}$$

From Lemma (4.2.2) we have

$$g(x) = x + (x+1) + x(-\log|x+1|)$$

$$= (2x+1) - x \operatorname{Log} |x+1|$$

(1.b) Inverse-Volterra-type

$$f(x) + \int_{x}^{1} \frac{f(y)}{y-x}\, dx = g(x) \qquad -1 \leq x \leq 1$$

$$g(x) = x + \int_{x}^{1} dy + x \int_{x}^{1} \frac{dy}{y-x}$$

Lemma (4.2.2) gives:

$$g(x) = 1 + x \operatorname{Log} |x-1|$$

(1.c) Fredholm-type

$$f(x) + \int_{-1}^{1} \frac{f(y)}{y-x}\, dy = g(x) \qquad -1 \leq x \leq 1$$

$$g(x) = (x+2) + x \operatorname{Log} \left|\frac{1-x}{1+x}\right|$$

Problem (4.7.2)

Exact solution $f(x) = (x+\alpha)^{-1}$; $\alpha \notin [-1,1]$

(2.a) Volterra-type

$$g(x) = f(x) + \int_{-1}^{x} \frac{f(y)}{y-x}\, dy \quad ; \quad -1 \le x \le 1$$

$$= \frac{1}{(x+\alpha)} + \int_{-1}^{x} \frac{dy}{(y+\alpha)(y-x)}$$

$$= \frac{1}{(x+\alpha)} + \frac{1}{(x+\alpha)} \int_{-1}^{x} \frac{dy}{y-x} - \frac{1}{(x+\alpha)} \int_{-1}^{x} \frac{dy}{(y+\alpha)}$$

From Lemma (4.2.2) we have:

$g(x) = g_1(x) - g_2(x)$ where

$$g_1(x) = \frac{1}{(x+\alpha)} \left(1 + \text{Log } \left|\frac{\alpha-1}{\alpha+x}\right|\right) ;$$

$$g_2(x) = \frac{1}{(x+\alpha)} \text{ Log } |x+1|$$

(2.b) Inverse-Volterra-type.

$$g(x) = f(x) + \int_{x}^{1} \frac{f(y)}{y-x}\, dy \quad ; \quad -1 \le x \le 1$$

$$= \frac{1}{(x+\alpha)} + \frac{1}{(x+\alpha)} \int_x^1 \frac{dy}{y-x} - \frac{1}{(x+\alpha)} \int_x^1 \frac{dy}{y+\alpha}$$

from Lemma (4.2.2) we have:

$g(x) = g_1(x) + g_2(x)$ where

$$g_1(x) = \frac{1}{(x+\alpha)} \text{ Log } \left|\frac{x+\alpha}{\alpha+1}\right| + \frac{1}{x+\alpha}$$

$$g_2(x) = \text{Log } |1-x| \, (x+\alpha)^{-1}$$

(2.c) Fredholm-type

$$f(x) + \int_{-1}^{1} \frac{f(y)}{y-x} \, dy = g(x) \qquad ; \quad -1 \leq x \leq 1$$

hence

$$g(x) = \frac{1}{(x+\alpha)} + \int_{-1}^{1} \frac{dy}{(y-x)(y+\alpha)}$$

$$= \frac{1}{(x+\alpha)} + \frac{1}{(x+\alpha)} \int_{-1}^{1} \frac{dy}{y-x} - \frac{1}{(x+\alpha)} \int_{-1}^{1} \frac{dy}{y+\alpha}$$

$$= \frac{1}{(x+\alpha)} \left\{ 1 - \text{Log } \left|\frac{\alpha+1}{\alpha-1}\right| \right\} + \frac{1}{(x+\alpha)} \text{ Log } \left|\frac{1-x}{1+x}\right|$$

Problem (4.7.3)

$$f(x) + \int_{-2}^{\frac{1}{2}} \frac{f(y)}{y-x}\, dy = g(x) \qquad -2 \leq x \leq \tfrac{1}{2}$$

$$f(x) = \frac{1}{x+\alpha} \quad , \quad \alpha \notin [\underline{-2}, \overline{\tfrac{1}{2}}]$$

$$g(x) = \frac{1}{x+\alpha} \left\{ 1 - \text{Log } \left|\frac{\alpha+\frac{1}{2}}{\alpha-2}\right| \right\} + \frac{1}{x+\alpha} \text{ Log } \left|\frac{\frac{1}{2}-x}{2+x}\right|$$

Problem (4.7.4)

$$f(x) + \int_{0}^{1} \frac{f(y)}{y-x} dy = g(x) \qquad 0 \leq x \leq 1$$

$$f(x) = \frac{1}{x+\alpha} \quad , \qquad \alpha \notin [\underline{0}, \overline{1}]$$

$$g(x) = \frac{1}{x+\alpha} \left\{ 1 - \text{Log } \left|\frac{1+\alpha}{\alpha}\right| \right\} + \frac{1}{x+\alpha} \text{ Log } \left|\frac{1-x}{x}\right|$$

Problem (4.7.5)

$$f(x) + \int_{-1}^{x} \frac{f(y)}{y-x}\, dy = g(x) \qquad -1 \leq x \leq \tfrac{1}{2}$$

$$f(x) = \frac{1}{x+\alpha} \quad ; \quad \alpha \notin [-1, \tfrac{1}{2}] \quad ; \quad \alpha = 7.0$$

$$g(x) = \frac{1}{x+\alpha}\left\{1 + \log\left|\frac{\alpha-1}{\alpha+x}\right|\right\} - \frac{1}{x+\alpha}\,\mathrm{Log}\,|x+1|$$

Problem (4.7.6)

$$f(x) + \int_{x}^{2} \frac{f(y)}{y-x}\, dy = g(x) \qquad -4 \leq x \leq 2$$

$$f(x) = x$$

$$g(x) = 2 + x\log_e |x-2|$$

TABLE (4.7.1)

Computed MS-ERROR

for problem (4.7.1)

N	Problem (4.7.1a)		Problem (4.7.1b)		Problem (4.7.1c)	
3	3.852185	$X10^{-17}$	3.349465	$X10^{-17}$	2.223085	$X10^{-16}$
4	8.446443	-17	3.915261	-17	7.614716	-16
5	1.114849	-16	8.355797	-17	4.574660	-16
6	1.097062	-16	5.816721	-17	7.517792	-16
7	1.390668	-16	6.115154	-17	8.972819	-16
8	1.359696	-16	1.026158	-16	1.462702	-15
9	1.546358	-16	1.462325	-16	1.132914	-15
10	1.648023	-16	2.432101	-16	1.908959	-15
11	1.601857	-16	4.633822	-16	2.033869	-15
12	1.684939	-16	6.422655	-16	1.986623	-15
13	1.813073	-16	7.213128	-16	1.408878	-15
14	1.896252	-16	7.816584	-16	1.582351	-15
15	1.977299	-16	6.450235	-16	2.263904	-15
16	1.954503	-16	4.614272	-16	2.703725	-15
17	2.505227	-16	6.829688	-16	1.777726	-15
18	2.511134	-16	5.572537	-16	2.456309	-15

TABLE (4.7.2)

Computed MS-ERROR

for problem (4.7.2a)

N	α = 1.005	α =1.1	α = 1.5	α = 4.0	α = 9.0
3	1.9011 $X10^{+1}$	5.9653 $X10^{-1}$	4.4788 $X10^{-2}$	1.2382 $X10^{-3}$	1.0869 $X10^{-4}$
4	1.1433 +1	2.8339 -1	1.5788 -2	1.5772 -4	5.8588 -6
5	7.4126 +0	1.5522 -1	6.1139 -3	2.0173 -5	3.2772 -7
6	4.9964 +0	9.2746 -2	2.3484 -3	2.5648 -6	1.8281 -8
7	3.4128 +0	5.7821 -2	9.0536 -4	3.2911 -7	1.0289 -9
8	2.4587 +0	3.7042 -2	3.4872 -4	4.2064 -8	5.7632 -11
9	2.0056 +0	2.4293 -2	1.3440 -4	5.3468 -9	3.2075 -12
10	1.8797 +0	1.6209 -2	5.1653 -5	6.7298 -10	1.7659 -13
11	1.8544 +0	1.0831 -2	1.9700 -5	8.3786 -11	9.6153 -15
12	1.7635 +0	7.1049 -3	7.4473 -6	1.0403 -11	5.1773 -16
13	1.5564 +0	4.4763 -3	2.7889 -6	1.3053 -12	4.9573 -17
14	1.2425 +0	2.6889 -3	1.0471 -6	1.6862 -13	5.0032 -17
15	9.1481 -1	1.5775 -3	4.0016 -7	2.2305 -14	4.3709 -17
16	7.3667 -1	9.9127 -4	1.5747 -7	2.9373 -15	5.3985 -17
17	7.9853 -1	7.0585 -4	6.2628 -8	3.8659 -16	5.3017 -17
18	9.0893 -1	5.1507 -4	2.4283 -8	1.1878 -16	5.3524 -17

TABLE (4.7.3)

Computed MS-ERROR
for problem (4.7.2b)

N	α = 1.005	α = 1.1	α = 1.5	α = 4.0	α = 9.0
3	5.4251 $X10^{+1}$	1.2049 $X10^{+0}$	9.7235 $X10^{-2}$	1.8466 $X10^{-3}$	1.2362 $X10^{-4}$
4	2.7452 +1	5.0728 -1	2.6016 -2	2.0987 -4	7.2536 -6
5	1.7557 +1	3.0037 -1	1.0056 -2	2.5224 -5	3.7859 -7
6	1.2001 +1	1.6880 -1	3.4699 -3	2.9771 -6	1.9611 -8
7	9.5852 +0	1.0889 -1	1.3256 -3	3.7411 -7	1.0795 -9
8	7.6051 +0	6.6750 -2	4.9308 -4	4.6891 -8	5.9491 -11
9	6.6437 +0	4.3014 -2	1.8644 -4	4.8934 -9	3.2849 -12
10	5.6448 +0	2.6899 -2	6.9542 -5	7.3457 -10	1.8008 -13
11	4.9124 +0	1.7012 -2	2.5812 -5	9.0162 -11	9.7131 -15
12	4.0894 +0	1.0556 -2	9.5932 -6	1.1138 -11	5.2504 -16
13	3.3689 +0	6.5579 -3	3.6049 -6	1.3895 -12	8.1345 -17
14	2.7806 +0	4.1566 -3	1.4043 -6	1.8201 -13	1.0111 -16
15	2.4564 +0	2.7112 -3	5.5656 -7	2.4267 -14	9.9951 -17
16	2.3813 +0	1.8258 -3	2.2050 -7	3.2286 -15	9.7127 -17
17	2.3056 +0	1.2006 -3	8.3037 -8	4.1537 -16	9.9488 -17
18	2.0994 +0	7.5163 -4	2.9594 -8	1.4765 -16	1.0828 -16

TABLE (4.7.4)

Computed MS-ERROR

for problem (4.7.2c) :

N	α = 1.005	α = 1.1	α = 1.5	α = 4.0	α = 9.0
3	1.9552 $\times 10^{+1}$	4.2454 $\times 10^{-1}$	5.2237 $\times 10^{-2}$	1.7567 $\times 10^{-3}$	1.5301 $\times 10^{-4}$
4	1.3394 +1	8.6905 -1	5.5445 -2	4.3296 -4	1.4767 -5
5	5.6894 +0	1.5407 -1	8.0337 -3	2.7522 -5	4.2583 -7
6	9.6062 +0	2.6715 -1	5.7642 -3	5.1605 -6	3.3714 -8
7	3.3229 +0	6.5764 -2	1.2489 -3	4.3013 -7	1.2597 -9
8	7.5776 +0	9.4669 -2	7.3707 -4	7.3622 -8	9.1997 -11
9	2.3036 +0	2.7520 -2	1.8242 -4	6.7502 -9	3.7911 -12
10	5.7896 +0	3.3335 -2	9.6646 -5	1.0797 -9	2.5951 -13
11	1.5549 +0	1.0218 -2	2.4772 -5	1.0268 -10	1.1091 -14
12	3.9409 +0	1.1011 -2	1.2195 -5	1.5411 -11	7.5054 -16
13	9.7684 -1	4.0572 -3	3.4358 -6	1.5622 -12	7.2732 -17
14	2.7514 +0	4.3179 -3	1.7194 -6	2.3489 -13	1.2561 -16
15	7.1684 -1	2.0899 -3	5.5695 -7	2.6405 -14	6.9327 -17
16	2.5496 +0	2.0045 -3	2.7276 -7	3.9824 -15	1.2638 -16
17	6.8466 -1	8.5512 -4	8.0070 -8	4.1191 -16	1.1924 -16
18	1.9232 +0	6.6942 -4	3.3766 -8	2.8412 -16	1.2675 -16

TABLE (4.7.5)

Computed MS-ERROR

for problems (4.7.3), (4.7.4)

N	Problem (4.7.3) $\alpha = 2.1$		$\alpha = 9.0$		Problem (4.7.4) $\alpha = 1.1$		$\alpha = 9.0$	
3	4.3208	$X10^{-1}$	3.0979	$X10^{-4}$	6.9934	$X10^{-3}$	3.2784	$X10^{-5}$
4	9.2606	-1	4.2196	-5	2.2973	-3	1.4320	-6
5	1.6922	-1	1.6536	-6	1.8003	-4	1.9748	-8
6	3.1608	-1	1.8061	-7	4.3507	-5	7.2544	-10
7	7.7263	-2	9.1731	-9	4.5240	-6	1.2917	-11
8	1.2345	-1	9.2333	-10	9.9007	-7	4.4056	-13
9	3.4902	-2	5.1559	-11	1.1327	-7	8.6217	-15
10	4.7574	-2	4.8700	-12	2.3119	-8	3.3961	-16
11	1.4038	-2	2.8295	-13	2.7411	-9	5.5485	-17
12	1.7069	-2	2.5037	-14	5.2492	-10	8.0938	-17
13	6.1197	-3	1.4839	-15	6.6564	-11	9.5895	-17
14	7.2625	-3	2.2365	-16	1.2784	-11	1.2167	-16
15	3.5254	-3	8.6017	-17	1.7909	-12	7.1339	-17
16	3.7019	-3	1.1297	-16	3.4573	-13	1.2329	-16
17	1.5954	-3	1.1207	-16	4.5651	-14	1.2993	-16
18	1.3569	-3	1.3438	-16	8.6908	-15	1.6655	-16

TABLE (4.7.6)

Computed MS-ERROR

for problems (4.7.5), (4.7.6)

N	Problem (4.7.5)		Problem (4.7.6)	
3	1.45124932247	X10^{-4}	6.2103132180321	X10^{-16}
4	7.83435416156	-6	3.8432043013599	-15
5	4.38670022829	-7	4.1334674924832	-15
6	2.44985453637	-8	3.0868807974163	-15
7	1.37855562044	-9	1.6646507502957	-15
8	7.72058212519	-11	4.1277545249464	-15
9	4.29567099030	-12	8.6872901773451	-15
10	2.36441357113	-13	9.2017114443120	-15
11	1.28760665549	-14	6.2320777889542	-15
12	6.94124232209	-16	6.1652850877294	-15
13	6.34368238391	-17	9.5863403044256	-15
14	5.78222654645	-17	7.7446629139675	-15
15	5.28951295178	-17	5.6771849963233	-15
16	6.24924351001	-17	9.6334816564371	-15
17	6.25694341867	-17	1.1423274023229	-14
18	6.49185351098	-17	1.1090375291897	-14

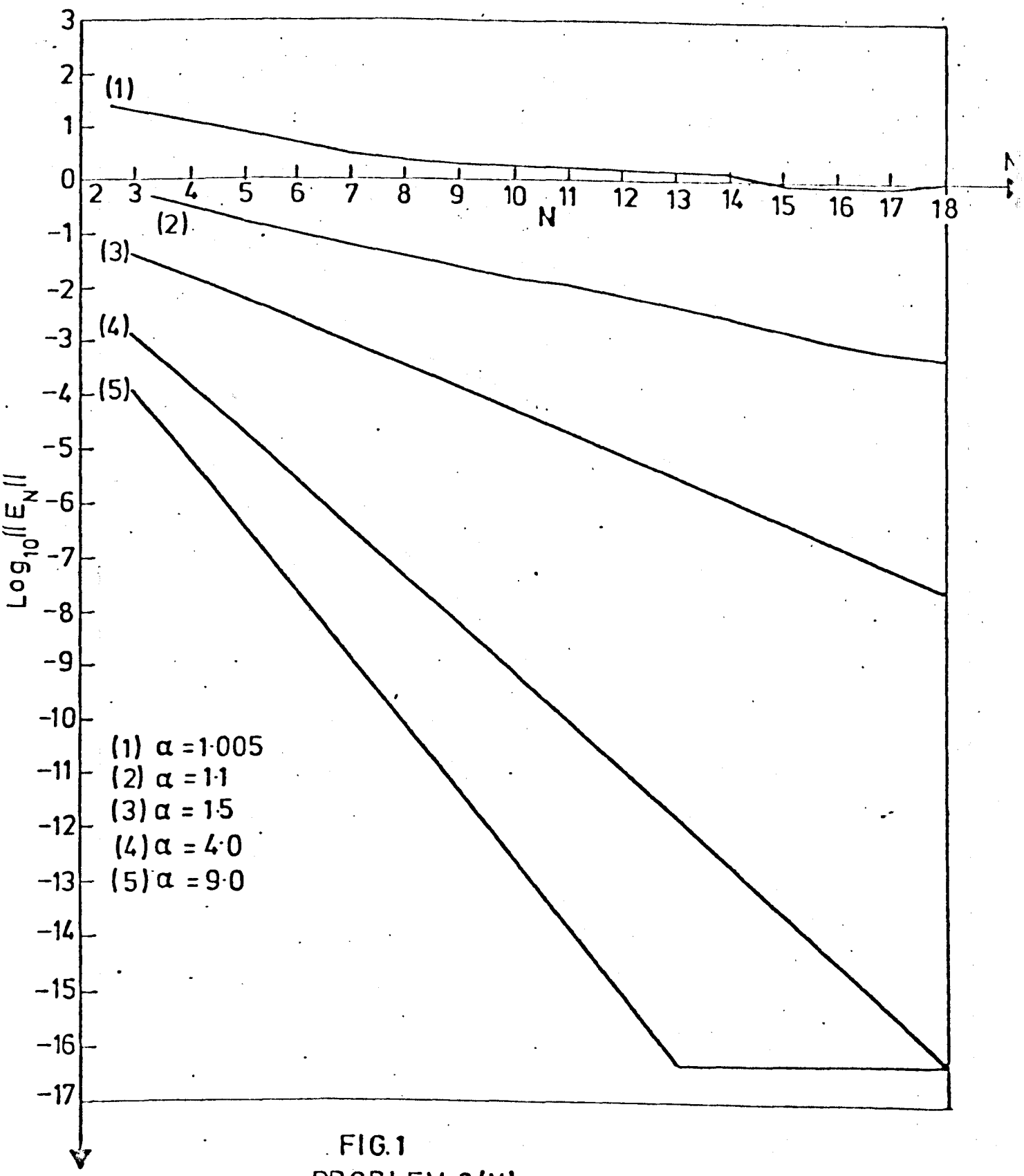

FIG. 1
PROBLEM 2(V)

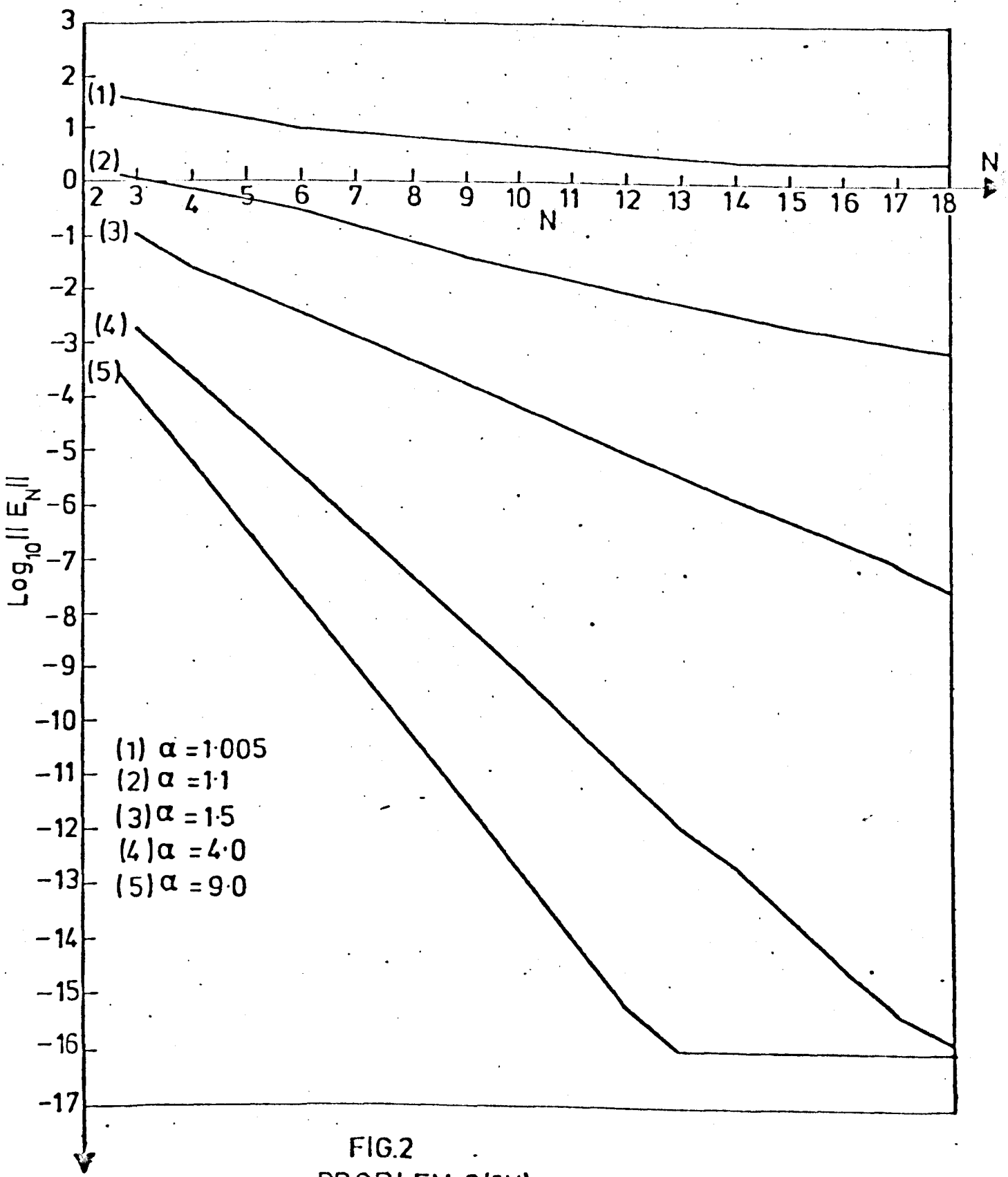

FIG.2
PROBLEM 2(IV)

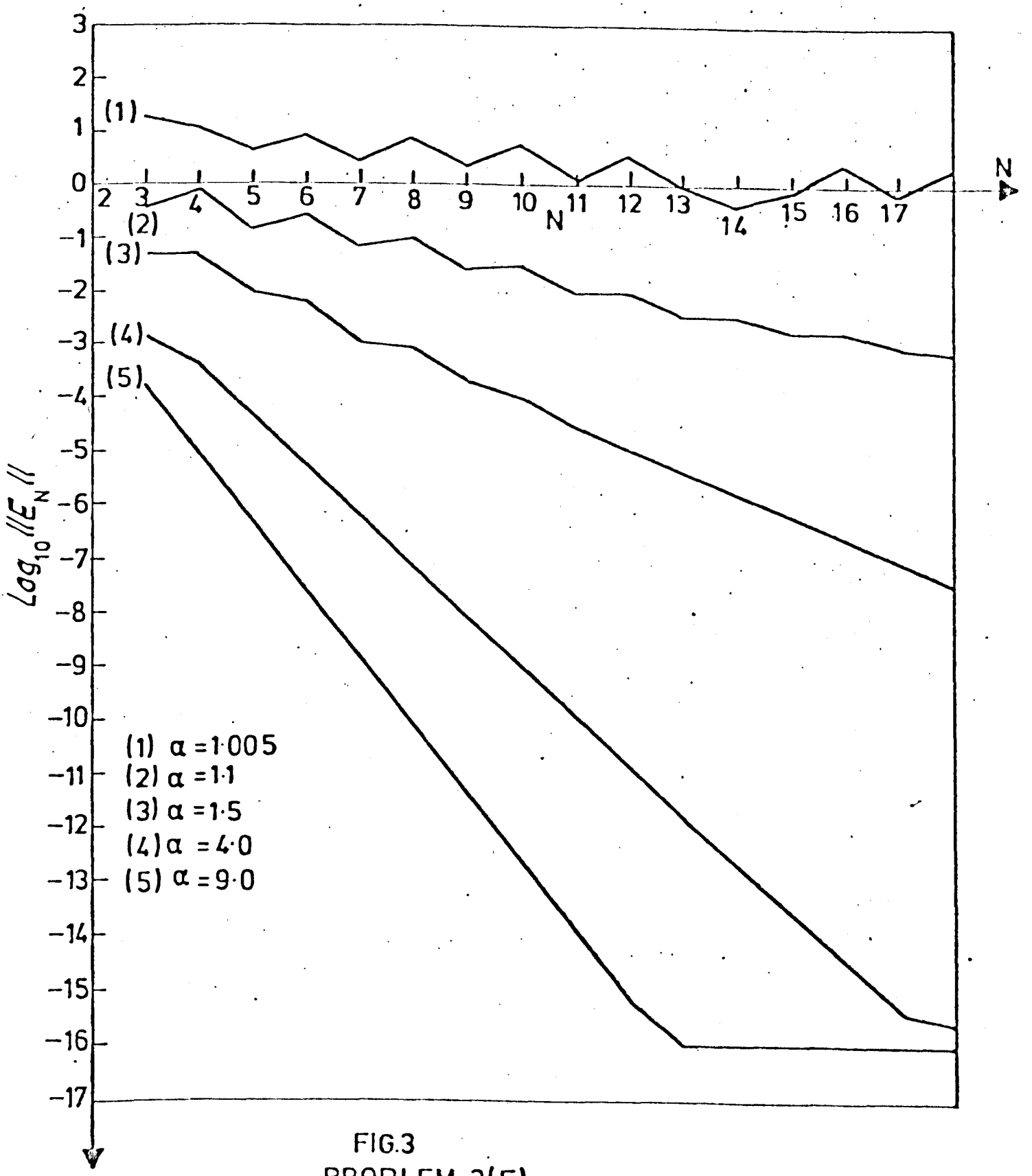

FIG.3
PROBLEM 2(F)

4.8 CONCLUSION

The algorithm described above yields a direct generalization of that given in [14].

The following points can be extracted from the above tables:

(a) Numerical results for problem (1) shows that good accuracy have been achieved as we expect from our choosing the exact solution (smooth).

(b) In problem (2) we chose the exact solution $(x+\alpha)^{-1}$ to be L_2-solution inside the range of integration but singular outside. Since in the beginning we assumed that the exact solution is square integrable in the interval of integration, hence our expectation for a good accuracy depends on our choice of the constant "α". We expect fast convergence with "α" far outside the interval, and slow convergence with "α" near to the interval. Figures (1-3) shows the convergence obtained graphically for problems (4.7.2a-c) with various values of the constant "α".

On the Log-linear scale used, the results are well fitted (apart from a possible odd-even effect in N) by a relation of the form

$$||f-f_N|| = C\,A^N$$

That is, "exponentially rapid" convergence. We expect this form of convergence, since the solution is in fact smooth everywhere inside the range of integration. However, whether the convergence is in fact rapid depends on the value of "α"; it is clear from the figures that the singularity approaches the interval of integration, $|\alpha|$ approaches 1 and convergence becomes very slow.

(c) The method works with a good accuracy when we take the interval of integration different from the standard interval $[-1, 1]$ as in problems (4.7.3-6).

The method which we have described has not been compared with other methods, at the time of preparing this work. The methods described at the beginning of this chapter are used to solve equations of the first kind, and they calculate the values of $\psi(1)$ of equation (4.1.2); the approximate solution is forced to be zero at the end point of the interval, which in general leads to only slow convergence to the exact solution at the interior points. The Fast Galerkin method avoids this difficulty, and the accuracy achieved is independent of the strength of the singularities in the equation, and depends only on the smoothness of the solution.

We conclude from the results above, and from our experience with the method that it allows the efficient solution of a wide class of Cauchy-singular integral equations, the achievable accuracy is quite clear from the results which reflects the stability of the method.

An advantage of the Galerkin approach is that cheap and rather effective estimates can be made available, see [14], [16].

CHAPTER 5

THE FAST GALERKIN TECHNIQUE FOR LINEAR INTEGRO-DIFFERENTIAL EQUATIONS

5.1 INTRODUCTION

We describe in this chapter the Fast Galerkin scheme for computing the coefficients in the Chebyshev expansion of the solution of first and second order linear integro-differential equations.

The method is valid under the "usual conditions" (the solution required is bounded and possesses a finite number of maxima and minima in the finite range of integration). The essence of the method is that an expansion in Chebyshev polynomials is assumed for the highest derivative occurring in the equation. Other methods which are based on expansion techniques include those of:

(a) El-Gendi [18] who reduced a first order integro-differential equation to an equivalent integral equation by a preliminary integration.

(b) Wolfe [63] who used a generalization of the Clenshaw and Norton [11] technique to solve the first derivative $f'(x)$ recovering $f(x)$ at the end by final integration.

The method described here is a generalization of the Fast Galerkin scheme for second kind integral equations described in chapter (2) which retains $O(N^2 \text{Log}_e N)$ operations count of that scheme. We remark that the method shares that of Wolfe the feature that we solve first for the highest occurring derivative of $f(x)$, recovering the function itself in a subsidiary operation; however the approach is quite different in other respects from that of Wolfe. The method described here has the following advantages:

(a) The rapid convergence and low operations ($O(N^2 LogN)$) count of the Fast Galerkin scheme are retained.

(b) Integro-differential equations of Fredholm, Volterra and inverse-Volterra type are handled.

(c)The method is capable to handle singular integro-differential equations.

(d) The defining equations for the method are well conditioned.

Throughout this chapter we consider only Fredholm integral operators; however, the method applies equally well to equations containing Volterra operators.

In the following two sections we expand the function (solution) and the occurring derivatives in terms of Chebyshev polynomials, then we apply Galerkin scheme to obtain a linear system which contains the coefficients of the derivatives expansion and the coefficients of the function expansion. We then write the equations which relates the function (solution) coefficients and the derivatives coefficients.

Finally, we apply the method to a system of singular integro-differential equations containing a Cauchy kernel in which we use the technique described in chapter (4), for computing the singular part. The method aims to retain a very rapid convergence and also a stable structure of the Fast Galerkin equations; numerical examples indicate that these aims are met.

5.2 LINEAR FIRST ORDER INTEGRO-DIFFERENTIAL EQUATIONS

We consider here the following linear first order integro-differential equation with linear boundary condition.

$$Q(x)\, f'(x) + R(x)\, f(x) + \lambda \int_{-1}^{1} k(x,y)\, f(y)\, dy = g(x) \qquad -1 \leq x \leq 1 \qquad (5.2.1a)$$

$$\underline{c}^t\, f(\underline{b}) + \underline{d}^t\, f'(\underline{b}) = e \qquad (5.2.1b)$$

where

$\underline{b} = (b_1,\ldots,b_m)^t$; $-1 \leq b_i \leq 1$, m is the number of the boundary points,

$$f(\underline{b}) = (f(b_1),\ldots,f(b_m))^t \ ; \ f'(\underline{b}) = (f'(b_1),\ldots,f'(b_m))^t \qquad (5.2.1c)$$

$\underline{c} = (c_1,\ldots,c_m)^t$; $\underline{d} = (d_1,\ldots,d_m)^t$

Let us introduce the Chebyshev polynomial expansion of

$f(x)$, $f'(x)$, $Q(x)$ and $R(x)$, $g(x)$:

$$f(x) = \sum_{j=0}^{\infty}{}' a_j\, T_j(x) \qquad ; \qquad f'(x) = \sum_{j=0}^{\infty}{}' a'_j\, T_j(x) \qquad (5.2.2)$$

$$Q(x) = \sum_{s=0}^{\infty}{}' q_s\, T_s(x) \ ; \ R(x) = \sum_{s=0}^{\infty}{}' r_s\, T_s(x) \ ; \ g(x) = \sum_{s=0}^{\infty}{}' g_s\, T_s(x)$$

Applying the Galerkin scheme to (5.2.1a) with the expansions in (5.2.2) we end up with the infinite system of equations:

$$\sum_{j=0}^{\infty}{}' a_j \left[\sum_{s=0}^{\infty}{}' q_s \int_{-1}^{1} T_s\, T_j\, T_i\, W(x)\, dx \right] + \sum_{j=0}^{\infty}{}' a_j \left\{ \sum_{s=0}^{\infty}{}' r_s \int_{-1}^{1} T_s\, T_j\, T_i\, W(x)\, dx \right.$$

$$\left. + \lambda \int_{-1}^{1} \int_{-1}^{1} k(x,y)\, T_j(y)\, T_i(x)\, W(x)\, dx\, dy \right\} = \frac{\pi}{2} g_i \qquad (5.2.3)$$

Since $T_i(x)\, T_j(x) = \frac{1}{2}(T_{i+j}(x) + T_{|i-j|}(x))$ then

$$\int_{-1}^{1} \frac{T_s(x)\, T_j(x)\, T_i(x)}{\sqrt{1-x^2}}\, dx = \frac{\pi}{2} \begin{cases} 2 , & i = j = s = o \\ \delta_{ij} , & i + j > o \text{ and } s = o \\ \frac{1}{2}(\delta_{s,i+j} + \delta_{s,|i-j|}) ; & s > o \end{cases}$$

where δ_{ij} is the Kronecker delta, whence we may write (5.2.3) in the form:

$$Q\, \underline{a}' + (R + \lambda B)\, \underline{a} = \underline{g} \qquad (5.2.4)$$

where

$$Q_{i,j} = \tfrac{1}{2}(q_{i+j} + q_{|i-j|}) ;$$

$$R_{i,j} = \tfrac{1}{2}(r_{i+j} + r_{|i-j|}) ; \qquad (5.2.4a)$$

$$B_{i,j} = \frac{2}{\pi} \int_{-1}^{1} \int_{-1}^{1} k(x,y)\, T_i(x)\, T_j(y)\, W(x)\, dy\, dx \quad , \quad j \geq 1 \; ; \; i = 0(1) \ldots .$$

and for $j = o$, $i \geq o$

$$Q_{i,o} = \frac{q_i}{2} \quad ; \quad R_{i,o} = \frac{r_i}{2} \quad ; \qquad (5.2.4b)$$

$$B_{i,o} = \frac{1}{\pi} \int_{-1}^{1} \int_{-1}^{1} k(x,y)\, T_i(x)\, W(x)\, dy\, dx \;;$$

$$W(x) = (1-x^2)^{-\frac{1}{2}}$$

Now the solution of the problem requires the evaluation of the coefficients a_j in the expansion:

$$f(x) = \tfrac{1}{2} a_o + a_1 T_1(x) + \ldots\ldots + a_n T_n(x) + \ldots \qquad (5.2.5a)$$

where $f(x)$ satisfies a linear integro-differential equation in the variable x with polynomial coefficients provided that the 'usual conditions' are satisfied, the series converges uniformly to $f(x)$ in the range but the series for the derivatives of $f(x)$ are not readily obtainable from (5.2.5a); we express formally the first derivative of $f(x)$ as:

$$f'(x) = \tfrac{1}{2} a'_o + a'_1 T_1(x) + \ldots\ldots + a'_N T_N(x) \ldots \qquad (5.2.5b)$$

(see for example [9])

then from the relation

$$2 \int T_j(x)\, dx = \frac{T_{j+1}(x)}{j+1} - \frac{T_{j-1}(x)}{j-1}$$

We deduce

$$2j\, a_j = a'_{j-1} - a'_{j+1} \qquad (5.2.5c)$$

Now to solve the system (5.2.4). For $\underline{a}'$ we have to find first $\underline{a}$, to do this we have to write $\underline{a}$ in terms of $\underline{a}'$. For introducing the ideas involved let us consider the solution of the first order ordinary differential equation

$$f'(x) = h(x) \qquad -1 \leq x \leq 1 \qquad (5.2.6a)$$

subject to the boundary condition (5.2.1b).

We introduce also the Chebyshev expansion of h(x)

$$h(x) = \sum_{j=0}^{\infty} a'_j T_j(x) \qquad (5.2.6b)$$

We can compute the coefficients a'_j using (FFT) technique described in chapter (2); here we assume that a'_j are known. Now we can write (from (5.2.5c)):

$$a_j = \frac{1}{2_j} (a'_{j-1} - a'_{j+1}) \qquad (5.2.7a)$$

in the matrix form:

$$\underline{a}^{(1)} = A\, \underline{a}' \qquad (5.2.7b)$$

where

$$A = (a_{ij}) = \begin{cases} \dfrac{1}{2(j+1)} & ; \quad j = i \geq 0 \\ \dfrac{-1}{2(j-1)} & ; \quad j = i + 2 \\ 0 & \text{otherwise} \end{cases} \qquad (5.2.7c)$$

$$\underline{a}^{(1)} = (a_1, a_2, a_3, \ldots, a_N, \ldots)^t$$

hence from (5.2.7), (5.2.6b) using the expansion of $f(x)$ in (5.2.2) equation (5.2.6a) can be written in the form

$$\begin{cases} \underline{a}^{(1)} = A\,\underline{a}' \\ \\ \underline{c}^t T\underline{a} + \underline{d}^t T\underline{a}' = e \end{cases} \qquad (5.2.8a)$$

where

$$T_{i,j} = T_j(b_i) \quad , \quad T_{i,o} = \frac{1}{2} \qquad i = 1(1)m \quad , \quad j = 1(1)\ldots. \qquad (5.2.8b)$$

Now from (5.2.8a) we obtain

$$(\underline{c}^t\underline{r})\, a_o = e - (\underline{d}^t T + \underline{c}^{t(1)} T A)\, \underline{a}' \quad ; \qquad (5.2.8c)$$

$$\underline{r} = (\tfrac{1}{2},\ldots,\tfrac{1}{2})^t \quad ; \quad T^{(1)}_{i,j} = T_{i(j+1)} \qquad i = 1(1)m \quad ; \quad j = 0(1)\ldots$$

hence we deduce that for equation (5.2.6a) to have a unique solution it is necessary that

$$\Delta_1 = \underline{c}^t\underline{r} = \tfrac{1}{2} \sum_{i=1}^{m} c_i \neq 0 \qquad (5.2.9a)$$

therefore assuming (5.2.9a) is valid, then

$$a_o = \underline{k}^t\,\underline{a}' + \mu \qquad (5.2.9b)$$

$$\underline{k}^t = -\frac{1}{\Delta_1}\left(\underline{d}^t T + \underline{c}^{t(1)} T A\right) \; ; \quad \mu = \frac{e}{\Delta_1} \tag{5.2.9c}$$

hence from (5.2.8),(5.2.9) we obtain

$$\underline{a} = A' \, \underline{a}' + \underline{\mu} \tag{5.2.10}$$

where

$$A' = (\underline{k}^t A)^t \quad ; \quad \underline{\mu} = (\mu, 0, 0, \ldots\ldots, 0)^t$$

Equation (5.2.10) represents an infinite set of equations for the expansion coefficients $\underline{a}$ and it gives a canonical relation between the expansion of a function and of its first derivative, subject to the boundary conditions, in practice we shall truncate all infinite expansions uniformally after the $(N+1)^{st}$ term and solve (5.2.10) as (N+1) x (N+1) system for the coefficients $a_o, \ldots\ldots, a_N$; we shall use equation (5.2.10) for the solution of first order and second order integro-differential equations to rewrite these in terms of defining equations for the highest occurring derivative. These defining equations then already incorporate the boundary conditions, and can be solved directly.

Now for the solution of (5.2.1), if we substitute $\underline{a}$ from (5.2.10) into (5.2.4) then we obtain

$$(Q + (R + \lambda B)A') \; \underline{a}'' = \underline{g}_1 \tag{5.2.11}$$

where $\underline{g}_1 = \underline{g} - (R + \lambda B) \, \underline{\mu}$ (5.2.12)

that is

$$(\underline{g}_1)_i = g_i - \mu\left(\frac{r_i}{2} + \lambda B_{i,0}\right) \qquad (5.2.13)$$

$$\underline{g}_1 = \underline{g} - \mu\underline{x} \qquad (5.2.14)$$

where $\underline{x}$ is the first column of $(R + \lambda B)$

So now the infinite system (5.2.11) is the representation of (5.2.1). So having found $\underline{a}'$ from (5.2.11) we can compute $\underline{a}$ from (5.2.10) to proceed numerically (5.2.11) must be truncated.

5.3 LINEAR SECOND ORDER INTEGRO-DIFFERENTIAL EQUATIONS

We consider the following linear integro differential system

$$P(x)f''(x) + Q(x)\,f'(x) + R(x)\,f(x) + \lambda \int_{-1}^{1} k(x,y)\,f(y)\,dy = g(x) \qquad (5.3.1)$$

$$Cf(\underline{b}) + Df'(\underline{b}) = \underline{e} \qquad -1 \leq x \leq 1$$

where $\underline{b}$, $f(\underline{b})$, $f'(\underline{b})$ are as defined in (5.2.1c) and C, D are 2 x m matrices; $\underline{e}$ is 2x1 vector.

To ensure that the system (5.3.1) has a unique solution, we assume that:

$$\Delta_2 = \det\left[\underline{Cr}, \underline{Cb} + 2\underline{Dr}\right] \neq 0 \ ; \text{ (see (5.3.11C))} \qquad (5.3.2)$$

If we introduce the expansions

$$P(x) = \sum_{s=0}^{\infty}{}' p_s T_s(x) \quad , \quad f''(x) = \sum_{j=0}^{\infty}{}' a''_j T_j(x) \tag{5.3.3}$$

and the same expansions introduced in (5.2.2); then by applying the weighted Galerkin technqiue to (5.3.1) we end up with the infinite system

$$P\underline{a}'' + Q\underline{a}' + (R + \lambda B)\,\underline{a} = \underline{g} \tag{5.3.4}$$

where the matrices Q, R, B are as defined in (5.2.4), and

$$P_{i,j} = \tfrac{1}{2}(p_{i+j} + p_{|i-j|})$$

$$P_{i,o} = \frac{p_i}{2} \qquad \begin{matrix} i = 0(1)\ldots. \\ j = 1(1)\ldots. \end{matrix} \tag{5.3.5}$$

Now with the same idea as for the first order we have to write $\underline{a}$, $\underline{a}'$ in terms of $\underline{a}''$. To do this let us consider the solution of the linear second order differential equation of the form

$$f''(x) = \sum_{j=0}^{\infty}{}' a''_j T_j(x) \qquad -1 \leq x \leq 1 \tag{5.3.6a}$$

$$Cf(\underline{b}) + Df'(\underline{b}) = \underline{e} \tag{5.3.6b}$$

where C, D, $f(\underline{b})$ and $f'(\underline{b})$ as defined before.

Equation (5.2.10) holds for (5.3.6) (considering one of the equations in (5.3.6b)) and we now are looking for a similar equation relating to $\underline{a}'$ and $\underline{a}''$ that is, similar to what we did for the first order. We try to write a_0' in terms of $\underline{a}''$.

Now from (5.2.5c) and a similar form we have:

$$\begin{cases} \overset{(1)}{\underline{a}} = A\,\underline{a}' \\ \overset{(1)}{\underline{a}'} = A\,\underline{a}'' \end{cases} \tag{5.3.7}$$

if we drop the first row and column of A we obtain another matrix, say A_1, so we can write

$$\overset{(2)}{\underline{a}} = A_1 \overset{(1)}{\underline{a}'} = A_1\, A\, \underline{a}'' = A_2\, \underline{a}'' \tag{5.3.8}$$

where A_2 is a matrix with elements

$$a_{i,j} = \begin{cases} \dfrac{1}{4(j+1)(j+2)} & j = i \geq o \\ \dfrac{-1}{2(j^2-1)} & j = i+2 \\ \dfrac{1}{4(j-1)(j-2)} & j = i+4 \\ 0 & \text{otherwise} \end{cases} \tag{5.3.9}$$

$$\overset{(1)}{\underline{a}'} = (a_1', a_2', \ldots\ldots, a_N' \ldots)^t$$

$$\overset{(2)}{\underline{a}} = (a_2, a_3, \ldots\ldots, a_N \ldots)^t$$

From (5.2.7a) we have:

$$a_1 = \tfrac{1}{2}(a'_0 - a'_2) \quad ; \quad a'_2 = \tfrac{1}{4}\,(a''_1 - a''_3)$$

$$a'_0 = 2a_1 + a'_2 = 2a_1 + \tfrac{1}{4}(a''_1 - a''_3)$$

$$a'_0 = 2a_1 + \underline{h}^t\,\underline{a}'' \qquad (5.3.10)$$

where

$$\underline{h} = (0,\ \tfrac{1}{4},\ 0,\ -\tfrac{1}{4},\ 0,\ 0\ldots.)^t$$

Now to find a_1 in terms of a''_i we write (5.3.6b) as

$$C\,T\,\underline{a} + D\,T\,\underline{a}' = \underline{e} \qquad (5.3.11a)$$

$$C\,[\underline{r}, \underline{b}]\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} + C\,{}^{(2)}T^{(2)}\,\underline{a} + D\,\underline{r}\,a'_0 + D\,{}^{(1)}T^{(1)}\,\underline{a}' = \underline{e} \qquad (5.3.11b)$$

where

$$\underline{r} = (\tfrac{1}{2},\ldots,\tfrac{1}{2})^t \ ; \ T_{i,j} = T_j(b_i) \ ; \ T_{i,0} = \tfrac{1}{2}$$

$$T^{(1)}_{i,j} = T_{i(j+1)} \ ; \ T^{(2)}_{i,j} = T_{i(j+2)} \quad i = 1(1)m \ ; \ j = 0(1)\ldots$$

Using (5.3.7), (5.3.8) and (5.3.10) from (5.3.11b) we have:

$$[\underline{Cr}\ \underline{Cb} + 2\,D\,\underline{r}]\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \underline{e} - (C\,{}^{(2)}T\,A_2 + D\,{}^{(1)}T\,A + D\,\underline{r}\,h^t)\,\underline{a}'' \qquad (5.3.11c)$$

which has a unique solution if (5.3.2) holds.

Lemma (5.1)

If $\underline{a}''$ is a solution to (5.3.6) then there exist a vector $\underline{L}$ and a number η such that

$$a_o' = \underline{L}^t \underline{a}'' + \eta \qquad (5.3.12)$$

Proof (see [2], and the references stated there).

Therefore from (5.3.7) and (5.3.12) we have:

$$\underline{a}' = A'' \underline{a}'' + \underline{\eta} \; ; \qquad (5.3.13)$$

$$A'' = (\underline{L}^t A)^t \quad , \quad \underline{\eta} = (\eta, 0, 0, \ldots, 0)^t$$

hence equation (5.3.13) represents the relation between the coefficients of the first and second derivatives.

Now from (5.3.4) using (5.2.10), (5.3.13) we obtain:

$$P\underline{a}'' + Q(A''\underline{a}'' + \underline{\eta}) + (R + \lambda B)[A'(A'' \underline{a}'' + \underline{\eta}) + \underline{\mu}] = \underline{g}$$

Or

$$[P + (Q + (R + \lambda B)A')\, A'']\ \underline{a}'' = \underline{g}_2 \qquad (5.3.14)$$

where

$$\underline{g}_2 = \underline{g} - \eta \underline{z} - \mu \underline{x} \; ;$$

$\underline{z}$ is the first column of $[\bar{Q} + (R + \lambda B)A^{\dagger}]$ and

$\underline{x}$ is the first column of $(R + \lambda B)$

hence for the solution of (5.3.1) we solve (5.3.14) for $\underline{a}''$ and evaluate $\underline{a}'$ from (5.3.13) then $\underline{a}$ from (5.2.10).

5.4 NUMERICAL PROCEDURES

We describe in this section an economical procedure for solving the system (5.3.14) numerically, and the same procedure can be applied to the system (5.2.11) of the first order linear integro-differential equation.

The basic decision to be made here is the way to truncate the infinite system (5.3.14).

First let us define

$$\underline{Z}^{(N)} = (z_0, z_1, \ldots\ldots\ldots, z_N)^t \qquad (5.4.1)$$

where we usually take $N \geq 3$;

then we write the infinite system

$$H\,\underline{a}'' = \underline{g}_2 \qquad (5.4.2)$$

where

$$H = P + [\bar{Q} + (R + \lambda B)A^{\dagger}]\,A''$$

bearing in mind that A', A'' are upper Hessenberg matrices.

The simple way is to truncate every expansion after the N^{th} term then all the vectors replaced by their - (N+1) - vectors, and the infinite square matrices by their (N+1) x (N+1) leading sub-matrices.

We can note that according to (5.2.5c) we need to know $\underline{a}''^{(N+2)}$ to evaluate $\underline{a}^{(N+1)}$ from (5.3.13) and then $\overset{(N)}{\underline{a}}$ from (5.2.10) because if we just evaluate $\overset{(N)}{\underline{a}''}$ then $a^{(N)}_{N-1}$, $a^{(N)}_{N}$ will have an error of order

$$\frac{\left|a''^{(N+2)}_{N+1}\right|}{4N(N-1)} \quad , \quad \frac{\left|a''^{(N+2)}_{N+2}\right|}{4N(N+1)}$$

respectively, and also a_0, a_1 will have similar errors, however these errors are small for those values of N. We evaluate also

$\underline{k}^{(N)}$, $\underline{L}^{(N)}$ from (5.2.9c) and Lemma (5.1).

without further approximation by evaluating elements of the m x (N+3), leading sub-matrix of the matrix T defined in (5.2.8b).

First of all let us make the assumption that

$$a''_i = 0 \qquad i > N$$

then the system (5.3.14) reduces to the system

$$\overset{(N)}{H}\,\overset{(N)}{\underline{a}''} = \overset{(N)}{\underline{g}_2} \tag{5.4.3}$$

where

$$\overset{(N)}{H} = P_{N,N} + \left(Q_{N,\,N+1} + \left(R_{N,\,N+2} + \lambda B_{N,\,N+2}\right)A'_{N+2,N+1}\right)\; A''_{N+1,N} \tag{5.4.4}$$

Where we mean by $Z_{r,s}$ the leading sub-matrices of order (r+1) x (s+1) obtained from the corresponding infinite matrix.

For a point of gain in computational convenience, we could replace $H^{(N)}$ by

$$G^{(N)} = P_{N,N} + (Q_{N,N} + (R_{N,N} + \lambda B_{N,N})A'_{N,N})\ A''_{N,N} \qquad (5.4.5)$$

From truncating all the vectors in (5.3.4), after the N^{th} term and replacing the infinite matrices by their (N+1) x (N+1) leading sub-matrices.

So instead we solve:

$$G^{(N)}\ \underline{a}''^{(N)} = \underline{g}_2^{(N)} \qquad (5.4.6)$$

where $G^{(N)}$ as in (5.4.5) which slightly differ from $H^{(N)}$ only in their last two columns, that is

$$H^{(N)}_{i,(N-1)} - G^{(N)}_{i,N-1} = \frac{R_{i,(N+1)} + \lambda\ B_{i,(N+1)}}{4N(N+1)} \qquad (5.4.7)$$

$$H^{(N)}_{i,N} - G^{(N)}_{i,N} = \frac{Q_{i,(N+1)}}{2(N+1)} + \frac{R_{i,(N+2)} + \lambda\ B_{i,(N+2)}}{4(N+1)(N+2)}$$

$$i = O(1)N$$

which as $N \to \infty$ the R.H.S. of (5.4.7) converges very slow.

Hence to speed up the convergence we have to reduce the gap between $\overset{(N)}{H}$, $\overset{(N)}{G}$ in order to find a better approximation to $\overset{(N)}{H}$ and improve the accuracy of the solution vector. To do this it is unwise and computationally inconvenient to evaluate $B_{i,(N+1)}$, $B_{i,(N+2)}$

for $i = 0(1)N$

but since already evaluated $\underline{r}^{(2N)}$ and $\underline{q}^{(2N)}$ we can define a better approximation to $\overset{(N)}{H}$ say $\overset{(N)}{U}$ as:
the first (N-1) columns of $\overset{(N)}{U}$ are the same as that of $\overset{(N)}{G}$ but

$$\overset{(N)}{U}_{i,(N-1)} = \overset{(N)}{G}_{i,(N-1)} + \frac{r_{N+1-i} + r_{N+1+i}}{8N(N+1)} \quad , \quad i = 0(1)N-1$$

$$\overset{(N)}{U}_{i,N} = \overset{(N)}{G}_{i,N} + \frac{q_{N+1-i} + q_{N+1+i}}{4(N+1)} + \frac{r_{N+2-i} + r_{N+2+i}}{8(N+1)(N+2)} \quad i = 0(1)N-2$$

$$\overset{(N)}{U}_{N,(N-1)} = \overset{(N)}{G}_{N,(N-1)} + \frac{r_1}{8N(N+1)}$$

$$\overset{(N)}{U}_{(N-1),N} = \overset{(N)}{G}_{(N-1)N} + \frac{q_2 + q_{2N}}{4(N+1)} + \frac{r_3}{8(N+1)(N+2)}$$

$$\overset{(N)}{U}_{N,N} = \overset{(N)}{G}_{N,N} + \frac{q_1}{4(N+1)} + \frac{r_2}{8(N+1)(N+2)}$$

Hence instead of solving (5.4.6) we may solve:

$$\overset{(N)}{U}\,\underline{a''} = \overset{(N)}{\underline{g}}_2 \qquad (5.4.8)$$

Similarly for the first order integro-differential system in equation (5.2.11) we solve

$$\overset{(N)}{E}\ \underline{a}' = \underline{g}_1 \qquad (5.4.9)$$

where, with the convention $Z_N = Z_{N,N}$

$$\overset{(N)}{E} = Q_N + (R_N + \lambda B_N)A'_N$$

We could solve the system

$$\overset{(N)}{F}\ \underline{a}' = \underline{g}_1 \qquad (5.4.10)$$

instead (5.4.9) by replacing $\overset{(N)}{E}$ by $\overset{(N)}{F}$ for the same reason mentioned before in the second order system, where $\overset{(N)}{E}$ and $\overset{(N)}{F}$ have the same first N columns and

$$\overset{(N)}{F}_{i,N} = \overset{(N)}{E}_{i,N} + \frac{r_{N+1-i} + r_{N+1+i}}{4(N+1)} \quad ; \quad i = 0(1)N-1$$

$$\overset{(N)}{F}_{N,N} = \overset{(N)}{E}_{N,N} + \frac{r_1}{4(N+1)}$$

Problems (5.6.3-6) of section (5.6) shows how this modification works in practice for both first and second order problems.

We evaluate the matrix B_N and the vectors $\overset{(2N)}{\underline{p}}$, $\overset{(2N)}{\underline{q}}$, $\overset{(2N)}{\underline{r}}$ and $\overset{(2N)}{\underline{g}_2}$, $\overset{(2N)}{\underline{g}_1}$ in $O(N^2 \text{Log } N)$ operations, using the technique described in Chapter (2).

We follow the economical technique of [2] for evaluating B_N as:

(1) $B_N \longleftarrow \lambda B_N$

(2) $B_N \longleftarrow (R_N + B_N)$ using only (5.2.4a) and $\underline{r}^{(2N)}$

(3) $B_N \longleftarrow B_N A'$ using $\underline{k}^{(N)}$

(4) $B_N \longleftarrow Q_N + B_N$ using (5.2.4a) and $\underline{q}^{(2N)}$

(5) $B_N \longleftarrow B_N A''$ using only $\underline{L}^{(N)}$

(6) $B_N \longleftarrow (P_N + B_N)$ using (5.2.4a) and $\underline{P}^{(2N)}$

The stages (1)-(6) requires only one matrix of order (N+1) x (N+1) which is B_N and a matrix with size m x (N+3) which is T in (5.2.8a) and 6 vectors, that is ($\underline{P}$, $\underline{q}$, $\underline{r}$, $\underline{g}_2$, $\underline{k}$, $\underline{L}$) each of them containing at most 2N+2 elements. The operation count in stages (3) and (5) is $O(N^2)$ because each column of A' (or A") contains at most three non-zero elements.

We consider the coefficient matrices H and E of equations (5.3.14) and (5.2.11) respectively, that is:

$$H = P + (Q+(R+\lambda B)A')A'' \quad ; \quad E = Q + (R+\lambda B)A'$$

Theorems (1,3) of Babolian and Delves [2] show, under certain smoothness assumptions, that the non-diagonal elements of the coefficient matrix H (and E) are uniformly bounded away from zero, and the necessary condition for the diagonal elements H_{ii} (and E_{ii}) to be uniformly bounded away from zero is that $p_0 \neq 0$ (and $q_0 \neq 0$), where q_0 and p_0 are the first coefficients of the Chebyshev expansion of Q(x) and P(x) respectively.

The above condition is in turn guaranteed if the following statement holds:

P(x) (or Q(x)) does not change sign on [-1, 1] (a)

We now introduce a definition.

Definition (5.1)

Let $E = Q + (R+\lambda B)A'$; $H = P + EA''$

$M = \inf_{i \geq o} |E_{ii}|$; $q_o \neq o$ for first order problems

$M = \inf_{i \geq o} |H_{ii}|$; $p_o \neq o$ for second order problems

Now if $M > o$, the coefficient matrix E (or H) is asymptotically diagonal of type B (see Freeman and Delves [25]) and hence:

(i) The matrix problems (5.4.6) and (5.4.9) are well conditioned.

(ii) The convergence and analysis of [25] is directly applicable and leads to bounds for $||\underline{a} - \underline{a}^{(N)}||$ and $||f-f_N||$.

We now note that a sufficient condition that the problem (5.2.1) (or (5.3.1)) has a unique solution is also given by (a).

If (a) does not hold, then equation (5.2.11) (or (5.3.14)) may be singular or ill-conditioned. Consider the case where (a) does not hold, but (5.3.1) (or (5.2.1)) has a unique solution. Then p_o (or q_o) may be zero (which implies $M = 0$); and even if $p_o \neq o$ (or $q_o \neq o$) we may have $M = 0$. In the next section we discuss these two cases, with the aim of retaining the satisfactory behaviour of the method without requiring P(x) (or Q(x)) to be positive.

5.5 THE CASE THAT P(x) (OR Q(x)) CHANGES SIGN ON $[-1, 1]$

Now before we proceed to a numerical example let us consider the case where the function Q(x) of equation (5.2.1) or the function P(x) of the system (5.3.1) changes their sign on the interval $[-1, 1]$.

Next we study two cases, first $p_o = 0$ (which implies $M = 0$), second if $p_o \neq 0$ we may have $M = 0$; and similar procedures can apply to first order.

(a) The case that $p_o = 0$ (or $q_o = 0$ for first order):-

If $p_o = 0$ in equation (5.3.4) then the corresponding matrix $G^{(N)}$ may be singular for some N and hence the vector solution is inaccurate, that is the numerical calculation is unreliable.

Hence we need to modify the numerical method to allow for this eventuality.

Let us consider when $p_o = 0$ in (5.3.4), the same method applies when $q_o = 0$ in (5.2.4).

Let r be the least positive integer for which

$$p_i = 0 \text{ if } i < r \quad \text{and} \quad p_r \neq 0 \tag{5.5.1}$$

then suppose given r let us apply the Galerkin technique on the following equation:

$$T_r(x)\,P(x)\,f''(x) + T_r(x)\,Q(x)\,f'(x) + T_r(x)\,R(x)\,f(x) + \lambda \int_{-1}^{1} k(x,y)\,T_r(y)\,f(y)\,dy$$

$$= T_r(x)\,g(x)$$

We end up with the system

$$P^*\,\underline{a}'' + Q^*\,\underline{a}' + (R^* + \lambda B^*)\,\underline{a} = \underline{g}^*$$

where

$$B^*_{i,j} = \tfrac{1}{2}\,(B_{i+r,j} + B_{|i-r|,j}) \quad i,j = 0(1).....$$

$$g^*_i = \tfrac{1}{2}(g_{i+r} + g_{|i-r|}) \,, \qquad i = 0(1)....$$

$$P^*_{i,j} = \tfrac{1}{2}(p^*_{i+j} + p^*_{|i-j|}) \qquad i = 0(1)...$$

$$j = 1(1)...$$

$$P^*_{i,o} = \frac{p^*_i}{2} \qquad i = 0(1)...$$

with

$$P^*_i = \tfrac{1}{2}(p_{i+r} + p_{|i-r|}) \,, \quad i = 0(1)...$$

with a similar definition for Q* and R*

if we take j = i then

$$P^*_{i,i} = \tfrac{1}{2}(p^*_{2i} + p^*_o)$$

Now since $p^*_o = p_r \neq 0$ and

$$P^*_{2i} = \tfrac{1}{2}(p_{2i+r} + p_{|2i-r|}) \quad \text{then}$$

$$P^*_{i,i} = \frac{p_r}{2} + \tfrac{1}{4}(p_{2i+r} + p_{|2i-r|})$$

that is $P^*_{i,i} \longrightarrow \frac{p_r}{2}$ as $i \longrightarrow \infty$

It is unlikely for small values of p_i can be treated as zero for choosing r, since $p^*_o = p_r$ remains valid independently of (5.5.1) and we require only $p^*_o \neq o$ and it is not necessary to require that $p_i = 0$, $i < r$ exactly. Problem (5.6.8) of section (5.6) shows how well this modification works in practice.

(b) The case that $p_o \neq 0$ (or $q_o \neq 0$ for first order I-D equations) but M of definition (5.1) is zero:

The solution of the systems (5.2.1) and (5.3.1) depends on the accuracy of the solution to the systems (5.4.9) and (5.4.6) respectively. In practice sometimes if $\lambda = 0$, $M = 0$ and $R(x) = 0$ then according to an "accidental" cancellation on the diagonal of P, (Q for first order I-D equations) lead to the corresponding coefficients matrices $\overset{(N)}{G}$ of (5.4.6) and $\overset{(N)}{E}$ of the system (5.4.9) be singular for some values of $N > 1$, and that is what we met in practice when we applied the method to the problem (5.6.9) of section (5.6), when we computed the vector $\underline{a}'^{(N)}$ from the system $\overset{(N)}{E}\,\underline{a}'^{(N)} = \underline{g}_1^{(N)}$ using Gauss elimination (with complete pivoting)

and back substitution, the results (see table (5.6.5) for some values of N are very bad.

The iterative scheme of Delves [15] treats this difficulty as follows:

If the coefficient matrix of the system proves ill-conditioned the scheme simply uses Gauss elimination, if in this case the matrix is found after N_o elimination stages to be numerically singular, the remaining $N-N_o$ unknowns are set to zero and a solution of the N_o-term approximation will be found. This solution is clearly also a valid result to return for an N-term approximation. Problem (5.6.10) of section (5.6) shows how this modification works in practice.

5.6 NUMERICAL EXAMPLES

Finally, we present some numerical examples to demonstrate how the method works in general. The examples displayed here have been chosen to show how the modification described works in general and how essential this modification required to obtain an accurate solution to the problems under consideration. The numerical calculation were carried out on 4341 computer with (64 bit-real).

Problem (5.6.1)

This problem was discussed by [9]

$$(5+3x)\, f'(x) = \frac{3}{2} f(x) \qquad -1 \leq x \leq 1$$

Solution : $f(x) = (5+3x)^{\frac{1}{2}}$

with boundary condition

$$f(-1) = \sqrt{2}$$

Problem (5.6.2)

$$x^2 f''(x) + \lambda \int_{-1}^{1} k(x,y)\, f(y)\, dy = g(x) \qquad -1 \leq x \leq 1$$

where: $\lambda = 0$, $f(x) = x^2$

$g(x) = 2x^2$; $k(x,y) = xy$

with boundary condition

$f(-1) = 1$; $f'(1) = 2$

Problem (5.6.3)

$$x^2 f'(x) + e^x f(x) + \lambda \int_{-1}^{1} e^{(x+1)y}\, f(y)\, dy = g(x) \qquad -1 \leq x \leq 1$$

where

$$g(x) = (x^2 + e^x)\, e^x + \frac{1}{x+2} (e^{x+2} - e^{-x-2})$$

with boundary condition

$$f(-1) + f(1) = e^{-1} + e^{1}$$

Problem (5.6.4)

As problem (5.6.3) but solved using the modification described in section (5.4) using the system (5.4.10) instead of (5.4.9).

Problem (5.6.5)

$$e^x f''(x) + \text{Cos}(x) f'(x) + \text{Sin}(x) f(x) + \lambda \int_{-1}^{1} e^{(x+1)y} f(y)\, dy = g(x) \; ;$$

$$-1 \leq x \leq 1$$

with boundary conditions

$$f(1) + f(-1) = e^1 + e^{-1}$$

$$f(1) + f(-1) - f'(-1) = e$$

where

$$g(x) = (e^x + \text{Cos}(x) + \text{Sin}(x))e^x + \frac{\lambda}{x+2} (e^{x+2} - e^{-x-2})$$

Solution: $f(x) = e^x$

Problem (5.6.6)

As problem (5.6.5) but solved by the modification described in section (5.4) using the system (5.4.8) instead of (5.4.6).

Problem (5.6.7)

$$xf''(x) + \lambda \int_{-1}^{1} k(x,y) f(y)\, dy = g(x) \qquad -1 \leq x \leq 1$$

where : $f(x) = e^x$

$$g(x) = xe^x + \frac{\lambda}{x+2} (e^{x+2} - e^{-x-2})$$

with boundary condition $f(1) = e^1$; $f'(-1) = e^{-1}$ and $\lambda = 0$.

Problem (5.6.8)

As problem (5.6.7) but solved by using the modification described in section (5.5-(a)).

Problem (5.6.9)

$$Q(x)\, f'(x) + \lambda \int_{-1}^{1} k(x,y)\, f(y)\, dy = g(x) \qquad -1 \leq x \leq 1$$

where : $\lambda = 0$; $f(x) = e^{x}$; $g(x) = Q(x)e^{x}$

$Q(x) = (8x^{4} - 8x^{2} + \frac{1}{2})$; $k(x,y) = e^{(x+1)y}$

with boundary condition

$$f(1) + f(-1) = e + e^{-1}$$

Problem (5.6.10)

As problem (5.6.9) solved by using the modification described in section (5.5-(b)).

TABLE (5.6.1)

Computed MS-ERROR for problems

(5.6.1) and (5.6.2)

N	Problem (5.6.1)	Problem (5.6.2) $\lambda = 0.0$
3	8.48193 $X10^{-4}$	5.03216 $X10^{-16}$
4	1.65783 -4	6.39256 -16
5	3.56508 -5	2.36276 -16
6	7.96143 -6	3.95140 -16
7	2.08315 -6	2.39084 -16
8	5.29486 -7	6.69019 -16
9	1.48465 -7	6.44356 -16
10	4.15952 -8	3.77477 -15
11	1.19146 -8	1.93174 -15
12	3.56840 -9	1.14129 -15
13	1.02064 -9	1.45197 -15
14	2.95717 -10	4.14335 -15
15	8.43011 -11	1.26622 -16

TABLE (5.6.2)

Computed MS-ERROR for problems

(5.6.3) and (5.6.4)

Note that problem (5.6.3) solved using the system (5.4.9) while problem (5.6.4) solved by the modified system (5.4.10)

N	Problem (5.6.3)		Problem (5.6.4)	
	$\lambda = 1.0$			
3	4.63428	$X10^{-2}$	4.08522	$X10^{-2}$
4	1.08485	-2	9.75776	-3
5	2.24532	-3	2.04626	-3
6	3.32007	-4	3.05106	-4
7	3.95864	-5	3.67130	-5
8	4.17286	-6	3.89333	-6
9	4.10493	-7	3.84993	-7
10	3.71320	-8	3.49722	-8
11	3.09503	-9	2.92695	-9
12	2.37593	-10	2.25383	-10
13	1.73325	-11	1.64931	-11
14	1.18443	-12	1.13008	-12
15	7.56255	-14	7.23562	-14

TABLE (5.6.3)

Computed MS-ERROR for problems

(5.6.5) and (5.6.6)

Note that problem (5.6.5) solved using the system (5.4.6) while problem (5.6.6) solved by the modified system (5.4.8)

N	Problem (5.6.5) $\lambda = 1.0$		Problem (5.6.6) $\lambda = 1.0$	
3	6.76339	$X10^{-3}$	6.50952	$X10^{-3}$
4	1.61016	-3	1.60301	-3
5	7.85457	-5	7.67872	-5
6	1.26783	-5	1.25160	-5
7	1.13351	-6	1.12424	-6
8	1.37719	-7	1.36745	-7
9	9.49618	-9	9.42793	-9
10	8.39912	-10	8.34901	-10
11	5.24181	-11	5.21244	-11
12	4.02253	-12	4.00257	-12
13	2.30661	-13	2.29525	-13
14	1.59451	-14	1.58689	-14
15	6.56925	-16	6.55746	-16

TABLE (5.6.4)

Computed MS-ERROR for problems

(5.6.7) and (5.6.8)

Note that problem (5.6.7) has P_o = o and problem (5.6.8) solves the system using the modified defining equations described in section (5.5-(a)).

N	Problem (5.6.7) λ = 0.0		Problem (5.6.8) λ = 0.0	
3	8.99724	X_{10}^{-2}	9.52138	X_{10}^{-2}
4	1.32824	-3	1.33444	-3
5	1.78214	-3	1.81632	-3
6	1.82754	-3	1.48953	-5
7	1.48432	-5	1.50347	-5
8	8.22795	-1	5.76225	-8
9	6.49464	-8	6.54133	-8
10	5.56379	+5	1.77530	-10
11	1.80641	-10	1.81648	-10
12	8.35789	+2	3.11934	-13
13	3.38320	-13	3.38112	-13
14	3.21572	-16	4.13243	-15

TABLE (5.6.5)

Computed MS-ERROR for problems

(5.6.9) and (5.6.10)

Note that problem (5.6.10) solved by using the modified scheme described in section (5.5-(b)).

N	Problem (5.6.9) $\lambda = 0.0$	Problem (5.6.10) $\lambda = 0.0$
3	$4.51970 \times 10^{+14}$	ill-conditioned
4	$5.72462 \times 10^{+14}$	4.22841×10^{-4}
5	$2.14564 \times 10^{+12}$	8.24433×10^{-2}
6	2.25408×10^{-2}	2.25408×10^{-2}
7	4.12674×10^{-3}	4.12675×10^{-3}
8	4.29777×10^{-4}	4.29777×10^{-4}
9	$1.50838 \times 10^{+10}$	4.27986×10^{-3}
10	$2.18103 \times 10^{+8}$	3.85975×10^{-6}
11	3.02961×10^{-7}	3.02961×10^{-7}
12	3.16472×10^{-8}	3.16472×10^{-8}
13	1.12495×10^{-9}	1.12495×10^{-9}
14	$2.33148 \times 10^{+4}$	4.81124×10^{-11}
15	$6.43673 \times 10^{+2}$	1.2249×10^{-12}

5.7 THE APPROXIMATE SOLUTION OF SINGULAR INTEGRO-DIFFERENTIAL EQUATIONS IN ELASTIC CONTACT PROBLEMS

Application of singular integro-differential equations in various branches of mechanics are well known. Among these are elastic contact problems, stresses in composite materials, airfoil problems, etc. The exact solution of these problems is usually not available and in such cases approximate methods have been commonly used. As described in chapter (4) for solving principal value problems which we use that technique and the technique described in this chapter for solving the following singular integro-differential equation:

$$Q(x)\, f'(x) + R(x)\, f(x) + \lambda_1 \int_{-1}^{1} \frac{f(y)}{y-x}\, dy + \lambda_2 \int_{-1}^{1} \frac{f'(y)}{y-x}\, dy$$

$$+ \lambda_3 \int_{-1}^{1} k(x,y)\, f(y)\, dy + \lambda_4 \int_{-1}^{1} k^*(x,y) f'(y)\, dy = g(x) \qquad (5.7.1)$$

$$-1 \leq x \leq 1$$

subject to the boundary condition

$$\underline{c}^t f(\underline{b}) + \underline{d}^t f'(\underline{b}) = e \quad ; \qquad -1 \leq b_i \leq 1$$

where $\lambda_{i;(i=1(1)4)}$ are real values.

Introducing the expansions (5.2.2) described in section (5.2) and applying the weighted Galerkin scheme to the system (5.7.1) we end up with the infinite linear system:

$$(Q + \lambda_2 C + \lambda_4 B^*)\,\underline{a}' + (R + \lambda_1 C + \lambda_3 B)\,\underline{a} = \underline{g} \qquad (5.7.2)$$

where

$$B^*_{i,j} = \frac{2}{\pi} \int_{-1}^{1} \int_{-1}^{1} k^*(x,y) \frac{T_i(x)}{\sqrt{1-x^2}}\, T_j(y)\, dy\, dx \quad ;$$

$$C_{i,j} = \frac{2}{\pi} \int_{-1}^{1} \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}} \frac{T_j(y)}{y-x}\, dy\, dx \quad ;$$

$$B^*_{i,o} = \frac{1}{\pi} \int_{-1}^{1} \int_{-1}^{1} k^*(x,y) \frac{T_i(x)}{\sqrt{1-x^2}}\, dx\, dy \quad ;$$

$$C_{i,o} = \frac{1}{\pi} \int_{-1}^{1} \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}} \frac{dy}{y-x}\, dx$$

while the matrices B, Q and R and the vector $\underline{g}$ are the same as described in section (5.2). We use the technique of chapter (2) to evaluate the matrices B, B* and the vectors $\underline{g}$, $\underline{q}$ and $\underline{r}$ while we use the scheme of chapter (4) to evaluate the matrix C.

We adopt the strategy of section (5.4) for truncating the infinite system (5.7.2) to get the finite linear system:

$$H^{(N)}\, \underline{a}'^{(N)} = \underline{g}^{(N)} \qquad (5.7.3)$$

which is the same as the finite system (5.4.9) only with extra matrices C and B*.

Comparison with other method:-

We present here an example given in Sanker et al [53] who uses a power series formulation instead of orthogonal polynomials and subsequently the collocation method for obtaining the system of equations. We try to compare our method with that method described in [53] for solving the singular integro-differential equation (5.7.1) where:

$f(x) = x^2 + x^3$;

$Q(x) = 1000$;

$R(x) = 1.0$; $\lambda_1 = \lambda_2 = \lambda_3 = 1.0$ and $\lambda_4 = 0.0$;

$g(x) = \frac{14}{3} + 2008.4\ x + 3003\ x^2 + x^3 + (2x + 4x^2 + x^3)\ \text{Log}\left|\frac{1-x}{1+x}\right|$

with the boundary condition:

$f(-1) = 0.0$; $f(1) = 2$

The results of the problem with the results obtained by [53] are displayed in table (5.7.1).

TABLE (5.7.1)

Comparison of the exact and the approximate solution for both our method and the method described in [53]

x	f_a	f_e	The exact solution		Our method; N = 7	
-0.951	0.0443	0.0443	4.4315648999	$x10^{-2}$	4.4315648994	$x10^{-2}$
-0.809	0.125	0.125	1.2500587100	-1	1.2500587100	-1
-0.588	0.142	0.142	1.4244642800	-1	1.4244652800	-1
-0.309	0.066	0.066	6.5977371000	-2	6.5977371004	-2
0.0	0.0	0.0	0.0		1.0056253093	-12
0.309	0.125	0.125	1.2498462900	-1	1.2498462900	-1
0.588	0.549	0.549	5.4904147201	-1	5.4904147201	-1
0.809	1.18	1.18	1.1839561290	+0	1.1839561290	+0
0.951	1.76	1.76	1.7644863510	+0	1.7644863510	+0

Note that we displayed the figures as shown in the paper [53], for f_a (approximate solution) and f_e (exact solution) which is most likely rounded to three figures. We carried out our computation on ICL 1906S computer displayed in the last two columns of table (5.7.1).

COMMENTS ON THE RESULTS

(I) If we use the system (5.4.8) instead of (5.4.6) to produce $\underline{a}''^{(N)}$, and consequently f_N, then MS-ERROR is slightly better (MS-ERROR is smaller than its corresponding value) when we solved problems (5.6.5), (5.6.6), and the same for problems (5.6.3) and (5.6.4) of first order equations (see tables (5.6.2-3)). This suggests that there is not much gain in using the modified systems (5.4.10), (5.5.8) to produce $\underline{a}''^{(N)}$, $\underline{a}'^{(N)}$ respectively.

(II) In problem (5.6.7) when $\lambda = 0$ the maximum error is very large for even values of N. The reason is that for this problem $p_0 = 0$ ($P(x) = x$) and in fact $U^{(N)}$ of the system (5.4.8) is singular for N even (10, 12 for this problem), thus, the condition $p_0 \neq 0$ ($q_0 \neq 0$, for first order) see [2] is essential, and hence the modification of section (5.5(a)) is necessary if this modification is to be relaxed. Problem (5.6.8) in the case $\lambda = 0$, shows that this modification works nicely. Comparing the obtained results with that of problem (5.5.7) (see table (5.6.4)).

(III) In problem (5.6.9) the matrix $F^{(N)}$ of the system (5.4.10) is singular for some values of N, and this has been reflected in the numerical results (see table (5.6.5)). Thus the modification described in section (5.5(b)) is necessary in this situation; and the results for problem (5.6.10) ($\lambda = 0$) show that this modification works well in practice.

(IV) In section (5.7) we applied the method on a singular system of integro-differential equation. The results in table (5.7.1) shows that the method with that scheme described in chapter 4 works very well. The reason is that the coefficient functions $Q(x)$, $R(x)$ and the exact solution $f(x)$ are smooth which gave a well-conditioned system (5.7.3).

CONCLUSION

The above remarks on the numerical results suggest that the Fast Galerkin scheme of chapter 5 is, in general, a stable, fast and straightforward method for solving integro-differential equations (singular or non-singular). Finally if $P(x)$ (or $Q(x)$ in first order equations) changes their sign in the range of the variable x we should use the method with caution.

CHAPTER 6

THE NUMERICAL SOLUTION OF THE EIGENVALUE OF AN INTEGRAL EQUATION

6.1 INTRODUCTION

In this chapter we discuss the application of the Fast Galerkin method described in chapter (2) to the computation of the real simple eigenvalues and their corresponding eigenfunctions of the integral equation

$$\int_a^b k(s,t)\,\phi(t)\,dt = \lambda\phi(s) \quad ; \quad a \leq s \leq b \qquad (6.1.1)$$

where $k(s,t)$ is a given kernel and a and b are finite parameters. We suppose that $k(s,t)$ is a smooth kernel or at worst has known singularity such as kernels having logarithmic singularities; and kernels of Cauchy-type.

In general we cannot guarantee the existence of any solution $\lambda \neq o$ to (6.1.1); for example, a continuous Volterra kernel (for which $k(s,t) \equiv o$ if $t > s$) has no continuous eigenfunctions for $\lambda \neq o$. Another example (see [7]) is the equation

$$\int_o^{2\pi} k(s,t)\,\phi(t)\,dt = \lambda\phi(s) \quad ; \quad (o \leq s \leq 2\pi)$$

$$k(s,t) = \mathrm{Sin}(s)\,\mathrm{Cos}(t) \quad ; \quad o \leq s,\ t \leq 2\pi$$

has no non-zero eigenvalues. Any non-null function $\phi(s)$ for which

$$\int_0^{2\pi} \text{Cos}(t)\phi(t)\, dt = 0$$

is an eigenfunction corresponding to the eigenvalue zero.

The numerical method which we shall describe for solving (6.1.1) will yield an approximate eigenvalue $\tilde{\lambda}$ to λ nearest to some arbitrary number μ and their corresponding approximate eigenfunctions. Since there may be a countably infinite number of solutions to the equation (6.1.1). Hence we cannot claim to solve the problem (6.1.1) completely. We can usually obtain an approximate value to the largest few eigenvalues in modulus, and their corresponding eigenfunctions since the eigenfunctions corresponding to the largest eigenvalues in modulus are usually smoother than those corresponding to small eigenvalues and it is usually easier to approximate the smoother function. But we may find difficulty to approximate the eigenfunctions whose corresponding eigenvalues are close to one another or if an eigenvalue is a multiple eigenvalue, but if the kernel of the problem under consideration is Hermitian, this difficulty does not arise. If λ is an eigenvalue associated with the kernel in (6.1.1) then (Smithies [58], p.103) the adjoint kernel $k^*(s,t)$ possesses an eigenvalue λ^*, we then have:

$$\int_a^b k(s,t)\, \phi(t)\, dt = \lambda\phi(s) \quad ; \quad a \leq s \leq b \quad ; \text{ and}$$

$$\int_a^b \overline{k(t,s)}\, \nu(t)\, dt = \lambda^* \nu(s) \qquad ; \qquad a \leq s \leq b$$

where $\nu(s)$ is an eigenfunction of $k^*(s,t)$, if we set $U(s) = \overline{\nu(s)}$ then

$$\int_a^b k(t,s)\, U(t)\, dt = \lambda U(s) \qquad ; \qquad a \leq s \leq b$$

so that λ is an eigenvalue of the transposed kernel $k^T(s,t) = k(t,s)$ and $U(s)$ is the corresponding eigenfunction. Sometimes we say that $U(s)$ is a left eignfunction of $k(s,t)$ corresponding to λ, whilst the eigenfunction $\phi(s)$ is then known as a right eigenfunction.

The accuracy obtainable in approximating a simple eigenvalue λ by a numerical method is governed in part by the condition number

$$\delta(\lambda) = \frac{\left| \int_a^b \phi(s)\, U(s)\, ds \right|}{||\phi(s)||_2 ||U(s)||_2}$$

which is invariant under scaling of $\phi(s)$, $U(s)$. The eigenvalue λ is badly conditioned if the condition number $\delta(\lambda)$ is very small, for a simple eigenvalue of a Hermitian kernel the condition number is unity, (see [6]).

The available numerical methods fall into two classes; those based on integration formulae and those which are expansion methods, in particular the Rayleigh-Ritz, Collocation and Galerkin schemes, where the first class of methods are generally simpler to implement than the second class methods. For a discussion of the numerical methods and underlying theory (see [7]).

We consider in this chapter the numerical computation of a real simple eigenvalue to the integral equation (6.1.1). In order to treat multiple as well as simple eigenvalues of (6.1.1) it could be necessary to employ projections of the approximate eigenfunctions associated with a particular eigenvalue of (6.1.1) onto the space spanned by the true eigenfunctions of (6.1.1) associated with that same eigenvalue. [see 54].

This is done since a single sequence of approximate eigenfunctions associated with a multiple eigenvalue of (6.1.1) may approach along particular subspaces different eigenfunctions associated with that same eigenvalue.

However, the totality of approximate eigenfunctions associated with a multiple eigenvalue of (6.1.1) provides an approximate basis for the space of eigenfunctions associated with that eigenvalue, with an error which decreases as the order of approximation increases.

In the next section we employ the Fast Galerkin scheme described in chapter (2) to (6.1.1) by assuming the eigenfunction in the mapped interval $[-1, 1]$ is smooth, with a rapidly convergent Chebyshev expansion.

The aim in this chapter is to produce an accurate approximation to the eigenvalue λ of (6.1.1) by solving the (N+1) term Galerkin equations as cheaply as possible.

Where the solution of the $(N+1)\times(N+1)$ Galerkin equations will yield an accurate approximation to the N-term expansion if (and essentially only if) this expansion is rapidly convergent. Some examples are given in section (6.4) to show how well this aim has been met; other examples which occur in practice have singular kernel given by Green's function kernels.

6.2 THE BASIC ALGORITHM

Since the range of integration of (6.1.1) is assumed finite, we can employ a change of variables to rewrite equation (6.1.1) in the form:

$$\int_{-1}^{1} k^{(1)}(x,y)\, f(y)\, dy = \lambda f(x) \quad ; \quad -1 \leq x \leq 1 \tag{6.2.1}$$

where

$$k^{(1)}(x,y) = \frac{b-a}{2}\, k\left(\frac{b-a}{2}\, x + \frac{b+a}{2}\, ,\ \frac{b-a}{2} y + \frac{b+a}{2}\right) \quad ,$$

$$f(x) = \phi\left(\frac{b-a}{2}\, x + \frac{b+a}{2}\right)$$

We approximate the eigenfunction $f(x) \in L_2[-1,1]$ by the truncated Chebyshev expansion:

$$f(x) \simeq f_N(x) = \sum_{i=0}^{N} a_i\, T_i(x) \quad ; \quad -1 \leq x \leq 1 \tag{6.2.2}$$

By applying the weighted Galerkin method on the equation (6.2.1) as described in chapter (2) we end up with the matrix eigenvalue problem

$$(B - \lambda D)\, \underline{a} = \underline{0} \tag{6.2.3}$$

which is a non-standard eigenvalue problem, where the elements of the eigenvector $\underline{a}$ of (6.2.3) are the Chebyshev coefficients of the expansion (6.2.2) which define the approximate eigenfunction of (6.2.1) and the matrices B, D given by:

$$D_{i,j} = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}}\, T_j(x)\, dx = \begin{cases} \pi & ; \quad i = j = 0 \\ \frac{\pi}{2} & ; \quad i = j > 0 \\ 0 & ; \quad i \neq j \end{cases} \qquad (6.2.4)$$

$$B_{i,j} = \int_{-1}^{1} \frac{T_i(x)}{\sqrt{1-x^2}} \int_{-1}^{1} k^{(1)}(x,y)\, T_j(y)\, dy\, dx \qquad (6.2.5)$$

The technique described in chapter (2) retains a total cost of order $O(N^2 Log(N))$ operations for evaluating the integrals in equations (6.2.4-5) and $O(N^2)$ operations for solving the system (6.2.3) where N+1 is the number of expansion functions used.

Now the solution of the matrix eigenvalue problem (6.2.3) which we set up, gives the approximate eigenvalue to the problem (6.2.1) which as well are the eigenvalue of the integral equation (6.1.1) and the corresponding eigenvector

$$\underline{a} = (a_0, a_1, \ldots\ldots, a_N)^t$$

of the matrix eigenvalue problem (6.2.3) are the Chebyshev coefficients from which values of the eigenfunction of the integral equation (6.2.1) may be computed (see Theorem (4) [54]).

In the case $k^{(1)}(x,y) = \overline{k^{(1)}(y,x)}$ the classical Galerkin method reduces to the Rayleigh-Ritz method, preserving symmetry in D (which is already symmetric) and B giving one-sided bounds for the positive and for the negative eigenvalues of the kernel, respectively (see [7]). But this case does not arise in our method because of the weight function $(1-x^2)^{-\frac{1}{2}}$ which we introduced by the weighted Galerkin scheme. For this

we cannot guarantee to preserve symmetry in the matrix B, even if the kernel k(s,t) is symmetric.

Hence our task now is to construct a suitable technique to find the eigenvalues of the matrix eigenvalue problem (6.2.3).

6.3 INVERSE ITERATION TECHNIQUE

This scheme can be used to find the eigenvalue nearest to some arbitrary number μ, since in general it is not possible to choose the vector $\underline{a} = (a_0,\ldots,a_N)^t$ and λ to make the residual $\eta(x)$ of the Galerkin scheme on (6.2.1)

$$\int_{-1}^{1} k^{(1)}(x,y)\, f_N(y)\, dy - \lambda^* f_N(x) = \eta(x) \qquad (6.3.1)$$

vanish. We can write (6.2.3) as

$$(B - \mu D)^{-1} D\, \underline{a} = (\lambda - \mu)^{-1}\, \underline{a} \qquad (6.3.2)$$

thus the eigenvectors of $(B - \mu D)^{-1}D$ are the same as those of (6.2.3) but the eigenvalues are $(\lambda_i - \mu)^{-1}$. Hence the eigenvalue of the largest modulus of $(B - \mu D)^{-1}D$ will be the eigenvalue of (6.2.3) nearest to the number μ.

Thus performing the power method on $(B - \mu D)^{-1}D$ rather than B of (6.2.3) the scheme is:

Step (1) - starting with arbitrary $\underline{a}^{(0)}$

Step (2) - compute $\underline{b}^{(k+1)} = (B - \mu D)^{-1} D\, \underline{a}^{(k)}$

Step (3) - set $\underline{a}^{(k+1)} = \underline{b}^{(k+1)}/\alpha_{k+1}$

where α_{k+1} is the element of $\underline{b}^{(k+1)}$ with largest modulus then

$$\alpha_{k+1} \to \frac{1}{\lambda-\mu} \quad \text{as} \quad k \to \infty$$

$$\lambda = \mu + \frac{1}{\alpha_{k+1}}$$

where λ is the eigenvalue of (6.2.3) nearest to μ. It is unnecessary to calculate $(B - \mu D)^{-1}D$ explicitly. We can obtain $\underline{b}^{(k+1)}$ by solving the system

$$(B - \mu D)\, \underline{b}^{(k+1)} = D\, \underline{a}^{(k)}$$

(Gauss Elimination)

where at each iteration only the right hand side is different.

6.4 NUMERICAL EXAMPLES

We illustrate this method on several problems which are described below, where the calculations made by different values of N, and the results are presented in tables (1 - 4). The calculations were carried out on the 1906S Computer, using 12-significant digits at the University of Liverpool.

Problem (1)

$$k(s,t) = \exp((st)) \quad ; \quad 0 \leq s,t \leq 1$$

This problem is discussed by Cryer [12] who finds that $\lambda \approx 1.3530$. Linz [37] finds that $\lambda = 1.35299$, while Baker [7] who uses Trapezium rule with $h = \frac{1}{80}$ to obtain $\lambda \approx 1.353058$, but he got $\lambda = 1.35303$ using N-point Gauss-Legendre rule for $N \geq 4$. ($\mu = 1.3$; 8 - iteration).

For this problem we expect to get very rapid convergence because of the smoothness of the kernel. These results are displayed in table (1) where the corrected value is $\lambda = 1.3530302$ as stated in ([7], p.191). ($\mu = 1.3$; 8-iteration).

Problem (2)

$$k(s,t) = \tfrac{1}{2}s(2-t) \quad ; \quad 0 \leq s \leq t \leq 1$$

$$= \tfrac{1}{2}t(2-s) \quad ; \quad 0 \leq t \leq s \leq 1$$

This problem is discussed by a number of authors:

(I) Cryer [12] who finds $\lambda = 0.24285$. As reported in his references, Mikhlin [43] found $0.2287 \leq \lambda \leq 0.2431$, using Rayleigh-Ritz method, and the trace of the kernel.

(II) Linz [37] found $\lambda = 0.24300$.

(III) Baker [7] found $\lambda = \frac{1}{4}$ where he used three-point Trapezoidal rule on $[0,1]$, in the quadrature method. The true eigenvalue of this problem are the roots of the equation $\mu + \tan(\mu) = 0$; $\lambda = 1/\mu^2$, with μ the smallest root of the transcendental equation gives $\lambda = 0.2496$, although the kernel has discontinuous derivative across the line $s = t$, these are handled exactly by the method we use (see chapter (4), section 4.2) and hence should (and do) cause no problems. The results of this method are shown in table (1). ($\mu = 0.2$; 10-iteration).

Problem (3)

The continuous kernel

$$k(s,t) = (st)^{\frac{1}{4}} \qquad ; \qquad 0 \leq s, t \leq 1$$

is a degenerate kernel with one non-zero eigenvalue [7]

$$\lambda = \int_0^1 t^{\frac{1}{2}}\, dt = \frac{2}{3}$$

This kernel is non-smooth so we expect a slow convergence to the exact eigenvalue, as shown in table (2). ($\mu = 0.66$; 15-iteration).

Problem (4)

$$k(s,t) = |s-t| \qquad ; \qquad 0 \leq s, t \leq 1$$

This problem is discussed by Linz [37] who finds the largest eigenvalue $\lambda \approx 0.34725$, where he gave the exact eigenvalue $\lambda_0 = 0.34741$; our results are displayed in table (1). ($\mu = 0.3$; 10-iteration).

Problem (5)

$$k(s,t) = (s^2 + t^2)^{\frac{1}{2}} \quad ; \quad 0 \leq s, t \leq 1$$

This problem is discussed by Cryer [12] who finds $\lambda = 0.81084$. The kernel is non-smooth kernel, so we expect to get a slow convergence to obtain an accurate eigenvalue as shown in table (3) which shows the same accuracy he got with N = 11.

While Baker et al [4] who uses Simpson's rule with $h = \frac{1}{16}$ to obtain $\lambda \approx 0.81085$ ($\mu = 0.8$; 15-iteration).

Problem (6)

$$k(s,t) = \frac{s}{t}(1-t) \quad ; \quad 0 \leq s \leq t \leq 1$$

$$= (1-s) \quad ; \quad 0 \leq t \leq s \leq 1$$

This problem is slightly artificial. This kernel is not Hermitian and it has a discontinuity at t = s = 0, hence we expect a very slow convergence to the eigenvalue. Baker [7] proves the eigenvalues of the kernel are real. He used a quadrature method with step $h = \frac{1}{32}$ to get $\lambda \approx 0.272117$. Our results are shown in table (1); they show poor convergence, and suggest $\lambda = 0.3$. ($\mu = 0.27$; 15-iteration).

Problem (7)

$$k(s,t) = \text{Sin}\,(s+t) \quad ; \quad 0 \leq s, t \leq 1$$

This problem is discussed by Mysovskih [46] who used the method of mechancial quadrature, using for the quadrature formula Gauss' formula with two points to obtain the approximate eigenvalues $\tilde{\lambda}_1 = 0.7983585$; $\tilde{\lambda}_2 = -0.093287$. This kernel is degenerate kernel, and its eigenvalues $\lambda_j (j=1,2)$ are

$\lambda_1 = 0.7993732$, $\lambda_2 = -0.0912966$.

Our approximation to those eigenvalues are shown in table (4).

($\mu_1 = 0.78$; $\mu_2 = 0.01$; 15-iteration)

TABLE (1)

The computed eigenvalues of problems (1,2,4,6)

N	Problem (1)	Problem (2)	Problem (4)	Problem (6)
3	1.3330380510	0.24211532258	0.34781999675	0.31285252058
4	1.3527120680	0.24297264105	0.34781999406	0.32711026189
5	1.3530127111	0.24296459387	0.34740999126	0.27987281276
6	1.3530300050	0.24296250896	0.34740999125	0.29777802695
7	1.3530301538	0.24296267877	0.34740827468	0.30670330511
8	1.3530301647	0.24296268560	0.34740827467	0.30898580230
9	1.3530301647	0.24296268511	0.34740826904	0.31082180071
10	1.3530301648	0.24296268510	0.34740826905	0.31473889405
11	" "	-	0.34740826904	0.31739736087
12	" "	-	0.34740826903	0.32089778407
13	" "	-	" "	0.32379012941
14	" "	-	" "	0.32714006460
15	" "	-	" "	0.33006827026
16	" "	-	" "	0.33286422577
17	" "	-	" "	0.36206116477
18	" "	-	" "	0.33931326653

TABLE (2)

The Computed Eigenvalue for problem (3)

N	$\tilde{\lambda}$	N	$\tilde{\lambda}$	N	$\tilde{\lambda}$	N	$\tilde{\lambda}$
3	0.7394346	14	0.6668789	25	0.667251	36	0.6666833
4	0.67177003	15	0.6669301	26	0.6667071	37	0.6666854
5	0.6758110	16	0.6668154	27	0.6667134	38	0.6666811
6	0.6685818	17	0.6668480	28	0.6666997	39	0.6666812
7	0.6695270	18	0.6667753	29	0.6667046	40	0.6666792
8	0.6675869	19	0.6667972	30	0.6666941	41	0.6666807
9	0.6679339	20	0.6667486	31	0.6666980	42	0.6666776
10	0.6671819	21	0.6667639	32	0.6666897	43	0.6666789
11	0.6673431	22	0.6667301	33	0.6666928	44	0.6666763
12	0.6669858	23	0.6667412	34	0.6666862	45	0.6666774
13	0.6670723	24	0.6667168	35	0.6666887	46	0.6666752

TABLE (3)

The Computed eigenvalue for problem (5)

N	$\tilde{\lambda}$	N	$\tilde{\lambda}$	N	$\tilde{\lambda}$
3	0.78381715799	13	0.81084427795	31	0.81084441632
4	0.81155911925	14	0.81084438166	32	0.81084441649
5	0.81056534723	15	0.81084436520	33	0.81084441643
6	0.81084329808	16	0.81084440115	34	0.81084441654
7	0.81083142165	17	0.81084439477		
8	0.81084345244	18	0.81084440914		
9	0.81084248187	19	0.81084440636		
10	0.81084415128	20	0.81084441270		
11	0.81084396500	21	0.81084441138		
12	0.81084432751	30	0.81084441640		

TABLE (4)

The Computed eigenvalue for problem (7)

N	$\tilde{\lambda}_1$	$\tilde{\lambda}_2$
3	0.64935544168	-0.088439675656
4	0.79950241286	-0.092460449157
5	0.79934397729	-0.091295826942
6	0.79937196226	-0.091297916601
7	0.79937215751	-0.091298712181
8	0.79937212989	-0.091298712037
9	0.79937212974	-0.091298711477
10	0.79937212976	-0.091298711475
11	0.79937212976	-0.091298711474
12	0.79937212978	-0.091298711469
13	0.79937212976	-0.091298711468

COMMENTS ON THE RESULTS

The following points can be extracted from the tables (1-4).

(I) Numerical results for problems (1,7) shows that for smooth kernels we get a very rapid convergence to the exact eigenvalue, while for non-smooth kernels we expect, and obtain, a slow convergence as in problems (3, 5).

(II) For kernels which have a discontinuous derivative across the line $t = s$ (problem 2). The Fast Galerkin method (see Chapter (4), section 4.2 and the references stated) treats Greens-function type operators as the sum of "Volterra" and "inverse - Volterra" operators, and handles this kind of problems efficiently as shown in table (1).

(III) The results of problem (6) are very bad. The reason for this is the difficulty of the kernel, as we know the kernel has a discontinuity at $s=t=0$.

CHAPTER 7

GENERAL CONCLUSION

In this final chapter, we try to summarise the overall effectiveness of the Fast Galerkin method for solving integral and integro-differential equations (singular or non-singular), described in the previous chapters. With regard to the method of chapter (3), it is well known that there is no such simple method for inverting the Laplace transform which has been successful for all types of the transform $\psi(p)$. Our aim was to develop a reliable method which we might hope to be successful for square integrable defining functions. But from the results shown in chapter (3) and also from further experience of the method it seems that this aim has not been met. The problem with our method stems from the mapping used and also the extreme difficulty which was experienced in estimating the parameters C_f,R; recall the significance of these parameters which are constants such that

$$|a_i| \leq C_f \hat{i}^{-R} \quad ; \quad \begin{matrix} \hat{i} = i \; ; \; i > 0 \\ = 1 \; ; \; i = 0 \end{matrix} \quad ; \quad i = 0(1)N$$

where a_i ; $i = 0(1)N$ are the Chebyshev coefficients of the exact solution. Then we may regularize the solution by imposing the above condition as a constraint on the computed solution vector $\underline{a}$ by solving the problem

$$\text{minimize } ||B\underline{a} - \underline{g}||$$

$$\text{subject to } |a_i| \leq C_f \hat{i}^{-R}$$

This problem may be formulated in any norm.

We found from our numerical results in chapter (3) that the ill-posed solver was highly sensitive to the parameters C_f,R. Although none of the methods ((1), (2) and (3)), described in section (3.5) of chapter (3) were particularly successful for estimating these parameters, additional numerical

experiments (not reported in chapter (3)) have led us to conclude that method (3) is slightly better.

Clearly more work is needed in order that:

(i) A reliable method for estimating the parameters C_f, R is found.

(ii) A reliable mapping from the semi-infinite to the finite subspace is found.

(iii) It may also be worthwhile to consider an expansion of the exact inverse in terms of Laguerre polynomials, that is $\phi(r) = \sum_{j=0}^{N} a_j L_j(r)$; and using the property

$$\int_0^\infty e^{-pr} L_j(r)\, dr = \frac{-(j+1)}{p} (p-1)^j \quad ; \qquad \text{Re } p > 0$$

where the transform $\psi(p)$ is known on some finite interval $[a, b]$

Second, for the numerical solution of the Cauchy principal value problem described in chapter (4), it is known that in physical problems, the ends ± 1 are points of geometric singularity. Usually the investigation of the behaviour of the unknown functions in the neighbourhood of these singular points is one of the main objectives in solving the problem generally, as we saw at the start of chapter (4). The methods described there, force the approximate solution to be zero at the end points (± 1) of the interval, and as we have remarked previously, this leads to only slow convergence to the exact solution at the interior points of the interval. The results obtained showed how easily the Fast Galerkin method avoids this difficulty, and how the accuracy achieved is independent of the strength of these singularities. We conclude from the results obtained, and from our experience with the method, that the Fast Galerkin framework handles even strongly singular problems and the achievable accuracy is quite clear from the results which reflect the stability of the scheme. The Volterra and inverse-Volterra results are perhaps amusing rather than of immediate practical significance, but kernels of Fredholm-type are of common occurrence, and the technique used seems to be well suited to these.

The method, as we mentioned in chapter (4), has not been compared with other methods and the following points are worthy of further investigation.

(1) Comparison with other methods.

(2) Extension of the method for solving a system of singular integral equations of the form

$$\sum_{i=1}^{n} \left[\phi_i(s) + \int_{-1}^{1} \frac{\phi_i(t)}{t-s}\, dt + \int_{-1}^{1} k_{ij}(s,t)\ \phi_i(t)\, dt \right] = g_j(s) \quad ;$$

$$j = 1(1)n$$

where $k_{ij}(s,t)$ and $g_j(s)$ $(i,j = 1(1)n)$ are known functions.

The Fast Galerkin scheme of chapter (5) for solving linear integro-differential euqations, is a straightforward scheme for solving such problems. Again the results obtained show how easily the scheme handles even strongly singular problems; the achievable accuracy reflects the stability of the method.

The possible extension of the method is also considered by Linz [38] to solve the system

$$P(x)\ f''(x) + Q(x)\ f'(x) + R(x)\ f(x) + \lambda \int_{-1}^{1} k(x,y)\ H(y)\, dy = g(x) \quad ; \quad -1 \leq x \leq 1$$

$$H(y) = U_1(y)\ f(y) + U_2(y)\ f'(y) + U_3(y)\ f''(y)$$

where U_1, U_2 and U_3 are known functions. Also the functions $P(x)$, $Q(x)$, $R(x)$ and $g(x)$ are as defined in chapter (5).

The corresponding system of linear equations is (with the boundary condition (5.3.1))

$$\left[(P + \lambda B_3) + ((Q + \lambda B_2) + (R + \lambda B_1)A')\, A''\right]\ \underline{a}'' = \underline{v}$$

where

$\underline{v} = \underline{g} - \eta\underline{z} - \mu\underline{x}$, and $\underline{z}$, $\underline{x}$ are the first columns of

$\left[(Q + \lambda B_2) + (R + \lambda B_1)A'\right]$ and $(R + \lambda B_1)$

respectively. B_1, B_2 and B_3 are the corresponding matrices of the functions $k(x,y)\ U_1(y)$, $k(x,y)\ U_2(y)$ and $k(x,y)\ U_3(y)$ respectively. Obviously having B (defined in (5.2.4a-b)) and Chebyshev expansions of $U_1(y)$, $U_2(y)$ and $U_3(y)$ we can easily evaluate B_1, B_2 and B_3 using the technique of Delves, Abd-Elal and Hendry [16]. The system can be reduced to a system of first order integro-differential equations by setting $P(x) = U_3(x) = 0$.

Finally the Fast Galerkin scheme of chapter (6) handles exactly two types of kernels: those which are everywhere smooth, and those which have discontinuous derivatives across the line t=s of the kernel k(s,t). The method computes approximations to a simple (real) eigenvalue of the kernel (nearest to a number (μ))and the corresponding eigenfunction. The achieved accuracy reflects the stability of the method. The method fails to compute an approximation to the eigenvalue of problem (6) of section (6.4); this is because of the discontinuity of the kernel at s=t=0.

Generally we conclude that from the results shown in the previous chapters, apart from that of the Laplace Transform discussed in chapter (3), the Fast Galerkin framework handles even strongly singular problems; the achievable accuracy together with that reliable error estimate discussed in [14] reflects the stability of the scheme. An additional advantage of the formalism, not shown here but discussed in [16] is that of handling problems whose kernel contains a singular factor together with an additive or multiplicative smooth term.

An obvious extension of the scheme is that the use of a single expansion over the whole problem region is unlikely to be optimal; an obvious extension, which would retain the rapid convergence of the basic method, is to partition the interval under consideration and then use different expansion sets on different sub-regions.

APPENDIX A

Proof of Lemma (3.3.2)

$$\int_0^1 \frac{xW(x)}{mx+r}\,dx = \frac{1}{m}\int_0^1 \left(1 - \frac{r}{mx+r}\right) W(x)\,dx$$

$$= \frac{1}{m}\int_0^1 W(x)\,dx - \frac{r}{m}\int_0^1 \frac{W(x)}{mx+r}\,dx$$

From Lemma (3.3.1) we have:

$$\int_0^1 \frac{xW(x)}{mx+r}\,dx = \frac{\pi}{2m}\begin{cases} 1 & r = 0 \\ 1 - \frac{r}{\sqrt{mr+r^2}} & r > 0 \end{cases}$$

Proof of Lemma (3.3.3)

$$\int_0^1 y^x\, T^*_{j-1}(y)\,dy$$

integrate by parts we have

$$\int_0^1 y^x T^*_{j-1}(y)\,dy = \frac{1}{4}\left[y^x \left(\frac{T^*_j(y)}{j} - \frac{T^*_{j-2}(y)}{j-2} \right) \right]_0^1$$

$$- \frac{x}{4}\int_0^1 y^{x-1}\left(\frac{T^*_j(y)}{j} - \frac{T^*_{j-2}(y)}{j-2} \right) dy$$

$$= \frac{1}{4}\left[\frac{2}{j(2-j)} - \frac{1}{j}\int_0^1 x\, y^{x-1} T^*_j(y)\,dy + \frac{1}{j-2}\int_0^1 x\, y^{x-1} T^*_{j-2}(y)\,dy \right]$$

APPENDIX B

First Row	+1.273239544735153&+0	-5.273930875790495&-1	+1.590041941661328&-1	-3.1690817413216 68&-1
	+2.594696293646550&-1	-2.906426259901472&-1	+2.652002087268910&-1	-2.774002264990935&-1
	+2.626310484449970&-1	-2.687076830762715&-1	+2.588900738694726&-1	-2.623133643284891&-1
	+2.552287118757939&-1	-2.572987493657782&-1	+2.518962384023193&-1	-2.532002431076750&-1
	+2.489139478488925&-1	-2.497518039083209&-1	+2.462481726268819&-1	-2.467869948014292&-1
	+2.438557982817158&-1	-2.441950018233149&-1	+2.416966759122746&-1	-2.418984795322247&-1
	+2.397364522689649&-1	-2.398414029400242&-1	+2.379465497146313&-1	-2.379818558462629&-1
	+2.363033836340987&-1	-2.362881455617363&-1	+2.347874796872544&-1	-2.347349796291648&-1
	+2.333826876181881&-1	-2.333025851575873&-1	+2.320755329756063&-1	-2.319749155641544&-1

First Col. The first element is $\frac{4}{\pi}$ and zero the rest

Last Row	+0.000000000000000&+0	+2.891526273461010&-27	-1.156610497518806&-26	+2.602373574921646&-26
	-4.626441800228254&-26	+7.228815090383986&-26	-1.040949333860819&-25	+1.416847641439713&-25
	-1.850576416352752&-25	+2.342135640806077&-25	-2.891525294634778&-25	+3.498745355303444&-25
	-4.163795797906775&-25	+4.886676595170257&-25	-5.667387717450896&-25	+6.505929132738019&-25
	-7.402300806654138&-25	+8.356502702455866&-25	-9.368534781034909&-25	+1.043839700091910&-24
	-1.156608931827352&-24	+1.275161168680163&-24	-1.399496405824654&-24	+1.529614638139223&-24
	-1.665515860306494&-24	+1.807200066763454&-24	-1.954667251711597&-24	+2.107917409117078&-24
	-2.266950532710866&-24	+2.431766615988907&-24	-2.602365652212290&-24	+2.778747634407418&-24
	-2.960912553661911&-24	+3.148860407646181&-24	-3.342591183570823&-24	+3.542104875229609&-24
Last Col.	-2.319749155641544&-1	+2.097109327239597&-1	-1.533095594966233&-1	+8.980136897062433&-2
	-4.277868321076724&-2	+1.697526888356895&-2	-5.755977512320342&-3	+1.707737861217588&-3
	-4.527578519629373&-4	+1.092651355757327&-4	-2.439552173075687&-5	+5.111301938624200&-6
	-1.017545037444429&-6	+1.945659644396796&-7	-3.606420072575193&-8	+6.530198295738144&-9
	-1.162339033754016&-9	+2.043767712937169&-10	-3.563245285336532&-11	+6.176847600296240&-12
	-1.066696110732713&-12	+1.837591113585373&-13	-3.160678675322967&-14	+5.431127289205548&-15
	-9.326878747420549&-16	+1.601122515092468&-16	-2.747997969266734&-17	+4.715744957322328&-18
	-8.091882411877184&-19	+1.388443403884579&-19	-2.382289556114446&-20	+4.087461288586949&-21
	-7.013074698893867&-22	+1.203263493572475&-22	-2.064483991450531&-23	+3.542104875229609&-24

REFERENCES

[1] BABOLIAN, E. and DELVES, L.M. An Augmented Galerkin Method for First Kind Fredholm Equations, J. Inst. Maths. Applics. 24, 157-174 (1979)

[2] BABOLIAN, E. and DELVES, L.M. A Fast Galerkin Scheme for Linear Integro-Differential Equations, IMA Journal of Numerical Analysis, 1, 193-213 (1981)

[3] BAIN, M. and DELVES, L.M. The Convergence Rates of Expansions in Jacobi Polynomials, Numerical Mathematik, 27, 219-225 (1977)

[4] BAKER, C.T.H., FOX, L., MAYERS, D.F. and WRIGHT, K. Numerical Solution of Fredholm Integral Equations of the First Kind, Computer, J. 7, 141-147 (1964)

[5] BAKER, C.T.H. Expansion Methods, pp.80-94 of L.M. Delves and J. Walsh (Eds.) Numerical Solution of Integral Equations, Clarendon Press, Oxford (1974)

[6] BAKER, C.T.H. Numerical Solution of Eigenvalue Problem, pp.126-135 of L.M. Delves and J. Walsh (Eds.), Numerical Solution of Integral Equations, Clarendon Press, Oxford (1974)

[7] BAKER, C.T.H. The Numerical Treatment of Integral Equations, Oxford University Press, Oxford (1977)

[8] BELLMAN, R., KALABA, R. and LOCKETT, J. Numerical Inversion of Laplace Transform, New York; Elsevier (1966)

[9] CLENSHAW, C.W. The Numerical Solution of Linear Differential Equations in Chebyshev series, Proc. Camb. Phill. Soc. 53, 134-149 (1957)

[10] CLENSHAW, C.W. and CURTIS, A.R. A Method for Numerical Integration on an Automatic Computer, Numer. Math., Vol.2, 197-205 (1960)

[11] CLENSHAW, C.W. and NORTON, H.J. The solution of non-linear ordinary differential equations in Chebyshev series, Computer J., 6, 88-92 (1963)

[12] CRYER, C.W. On the calculation of the largest Eigenvalue of an Intergral Equation, Numerische Mathematik, 10, 165-176 (1967)

[13] DELVES, L.M. and WALSH, J. Numerical Solution of Integral Equations, Clarendon Press, Oxford (1974)

[14] DELVES, L.M. A Fast Method for the Solution of Integral Equations, J. Inst. Maths. Applics. 20, 173-182 (1977a)

[15] DELVES, L.M. On the solution of the set of Linear Equations arising from Galerkin Methods, J. Inst. Maths. Applics. 20, 163-171 (1977b)

[16] DELVES, L.M., ABD-ELAL, L.F. and HENDRY, J.A. A Fast Galerkin Algorithm for Singular Integral Equations, J. Inst. Maths. Applics. 23, 139-166 (1979)

[17] DELVES, L.M., ABD-ELAL, L.F. and HENDRY, J.A. A Set of Modules for the Solution of Integral Equations, Computer J., 24, 184-190 (1981)

[18] EL-GENDI, S.E. Chebyshev Solutions of Differential, Integro-Differential Equations, Computer J., 12, 282-287 (1969)

[19] ELLIOTT, D. The Numerical Solution of Integral Equations using Chebyshev Polynomials, J. Austral. Math. Soc. 1, 344-356 (1959/60)

[20] ELLIOTT, D. A Chebyshev Series Method for Numerical Solution of Fredholm Integral Equations, Computer J.,6, 102-111 (1963)

[21] ELLIOTT, D. The Approximate Solution of Singular Integral Equations, pp.83-107 of M.A. Golberg (Ed.), Solution Methods for Integral Equations, Plenum Press (1978)

[22] ERDOGAN, F., GUPTA, G.D. On the Numerical Solution of Singular Integral Equations, Quarterly of Applied Mathematics, 30, 525-534 (1972)

[23] ERDOGAN, F., GUPTA, G.D. and COOK, T.S. Numerical Solution of Singular Integral Equations, pp.368-425 of G.C. Sih (Ed.) Mechanics of Fracture, Vol.1, Noordhoff, Leyden (1973)

[24] FOX, L. and PARKER, I.B. Chebyshev Polynomials in Numerical Analysis, Oxford University Press, Oxford (1968)

[25] FREEMAN, T.L. and DELVES, L.M. On the Convergence Rates of Variational Methods III: Unsymmetric Systems, J. Inst. Maths. Applics. 14, 311-323 (1974)

[26] GRUNDY, R.E. Laplace Transform Inversion using Two Points Rational Approximants, J. Inst. Maths. Applics. 20, 299-306 (1977)

[27] HADAMARD, J., LL.D., Lectures on Cauchy's Problems in Linear Partial Differential Equations, Yale University Press (1923)

[28] HANSON, R. A Numerical Method for Solving Fredholm Integral Equations of the First Kind using Singular Values, SIAM, J. Numer. Anal. 8, 616-622 (1971)

[29] HASHMI, S.M. The Numerical Solution of the Principal Value Integral Equations, M.Sc. Dissertation, SCM Dept., The University of Liverpool (1980)

[30] HASHMI, S.M. and DELVES, L.M. The Fast Galerkin Method for the Solution of Cauchy Singular Integral Equations, pp.445-458 of C.T.H.BAKER and F. MILLER (Eds.) Treatment of integral equations by numerical methods, Academic Press (1982/83).

[31] HENNESSY, T.R. A Comparative Approach to Laplace Transform Inversion, J. Comput. Applied Mathematics, Vol.4, No.2, 89-91 (1978)

[32] IOAKIMIDIS, N.I. and THEOCARIS, P.S. A Comparison Between the Direct and the Classical Numerical Methods for the Solution of Cauchy-type Singular Integral Equations, SIAM J. Numer. Anal. Vol.17, No.1, 115-118 (1980)

[33] KANTOROVICH, I.V. and KRYLOV, V.I. Approximate Methods of Higher Analysis, Interscience, New York, and Noordhoof, Groningen, The Netherlands (1958)

[34] KRENK, S. On the Use of Interpolation Polynomial for the Solution of Singular Integral Equations, Quart. Appl. Maths. 32, 479-484 (1975)

[35] KRENK, S. A Quadrature Formulas for Singular Integral Equations of the First and the Second Kind, Quart. Appl. Maths. 33, 225-232 (1975)

[36] LEWIS, B.A. On the Numerical Solution of Fredholm Integral Equations of the First Kind, J. Inst. Maths. Applics. 16, 207-220 (1975)

[37] LINZ, P. On the Numerical Computation of Eigenvalues and Eigenvectors of Symmetric Integral Equations, Mathematics of Computation, Vol.24, No.112, 905-910 (1970)

[38] LINZ, P. A Method for the Approximate Solution of Linear Integro-Differential Equations, SIAM J. Numer. Anal. Vol.11, No.1, 137-144 (1974)

[39] LONGMAN, I.M. Numerical Laplace Transform Inversion of a Function arising in Viscoelasticity, Journal of Computational Physics, 10, 224-231 (1972)

[40] LONGMAN, I.M. On the Generation of Rational Function Approximations for Laplace Transform Inversion with an Application to Viscoelasticity, SIAM J. Appl. Math. Vol.24, No.4, 429-439 (1973)

[41] McWHIRTER, J.G. and PIKE, E.R. On the Numerical Inversion of the Laplace Transform and similar Fredholm Integral Equations of the First Kind, J. Phys. A: Math. Gen. 11, No.9 1729-1745 (1978)

[42] METROPOLIS, N. Algorithm in Unnormalized Arithmetic I. Recurrence Relation, Numerische Mathematik 7, 104-112 (1965)

[43] MIKHLIN, S.G. Integral Equations, Pergamon (1957)

[44] MIKHLIN, S.G. and SMOLITSKY, Approximate Methods for Solution of Differential and Integral Equations, (trans. R.E. Kalaba), American Elsevier (1967)

[45] MILLER, G.F. Fredholm Equations of the First Kind, pp.175-185 of L.M. Delves and J. Walsh (Eds.) Numerical Solution of Integral Equations, Clarendon Press, Oxford (1974)

[46] MYSOVSKIH, I.P. On Error Bounds for Approximate Methods of Estimation of Eigenvalues of Hermitian Kernels (Transl:), AMS Translations (ser. 2) 1964, 35, pp.237-250 (1959)

[47] OLIVER, J. Relative Error Propagation in the Recursive Solution of Linear Recurrence Relations, Numerische Mathematik 9, 323-340 (1967)

[48] OLIVER, J. The Numerical Solution of Linear Recurrence Relations, Numerische Mathematik, 11, 349-360 (1968)

[49] OLVER, F.W.J. and SOOKNE, D.J. Note on Backward Recurrence Algorithms, Mathematics of Computation, Vol.26, 941-947 (1972)

[50] PHILLIPS, D.L. A Technique for the Numerical Solution of Certain Integral Equations of the First Kind, J. Ass. Comput. Mach. 9, 84-97 (1963)

[51] PIESSES, R. and BRANDERS, M. Numerical Inversion of the Laplace Transform using Generalized Laguerre Polynomials, Poc. IEE 118 1517-1522 (1971)

[52] PIESSES, R. A New Numerical Method for the Inversion of the Laplace Transform, J. Inst. Maths. Applics. 10, 185-192 (1972)

[53] SANKAR, T.S., HOA, S.V. and FABRIKANT, V.I. Approximate Solution of Singular Integro-Differential Equations in Elastic Contact Problems, International Journal for Numerical Methods in Engineering, Vol.18, 503-519 (1982)

[54] SCHLESINGER, S. Approximating Eigenvalues and Eigenfunctions of Symmetric Kernels, SIAM J. Numer. Anal. Vol.5, 1-14 (1957)

[55] SCHONFELDER, J.L. and THOMASON, J.T. Arbitrary Precision Arithmetic in Algol 68, Software-Practice and Experience, Vol.9, 173-182 (1979)

[56] SCRATON, R.E. The Solution of Integral Equations in Chebyshev Series, Math. Comp. 23, 837-845 (1969)

[57] SIMON, R.M., STROOT, M.T. and WEISS, G.H. Numerical Inversion of Laplace Transforms with Application to Percentage Labelled Mitoses Experiments, Computers and Biomedical Research, 5, 596-607 (1972)

[58] SMITHIES, F. Integral Equations, Cambridge University Press (1970)

[59] THEOCARIE, P.S. and IOAKIMIDIS, N.I. Numerical Integration Methods for the Solution of Singular Integral Equations, Quarterly of Applied Mathematics 35, 173-183 (1977)

[60] TIKONOV, A.N. Solution of Non-Linear Integral Equations of the First Kind, Soviet. Maths. 5, 835-838 (1964)

[61] TRICOMI, F.G. Integral Equations, Interscience, New York (1957)

[62] TURCHIN, V.F., KOZLOV, V.P. and MALKEVICH, M.S. The Use of Mathematical Statistics Methods in the Solution of Incorrectly Posed Problems, Soviet Phys. Wsp. 13, 681-702 (1971)

[63] WOLFE, M.A. The Numerical Solution of Non-Singular Integral and Integro-Differential Equations by Iteration with Chebyshev Series, Computer J. 12, 193-196 (1969)

[64] ZABREYKO, P.P. et al, Integral Equations, a reference text (trans. R.S. Anderson et al), Noordhoff International (1975)

[65] ZAKIAN, V. and LITTLEWOOD, R.K. Numerical Inversion of Laplace Transform by Weights Least-Squares Approximation, Computer J. 16, 66-68 (1973)