Numerical solution of Y" = F(X,Y) with particular reference to the radical schrödinger eqution

J. L. Mohamed

NUMERICAL SOLUTION OF

$Y'' = F(X,Y)$

WITH PARTICULAR REFERENCE TO

THE RADIAL SCHRÖDINGER EQUATION

J.L. MOHAMED, B.Sc., M.A.,
DEPARTMENT OF MATHEMATICS,
UNIVERSITY OF DURHAM.

THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY, FEBRUARY 1979.

ABSTRACT

Many theoretical treatments of quantum-mechanical scattering processes require the numerical solution of a set of second order ordinary differential equations of special form (with first derivative absent). The methods used to solve such sets of equations are generally based on step-by-step methods for solving a single second order differential equation over a fixed mesh. For example Chandra (1973) has published a computer program which uses de Vogelaere's method to solve the differential equations arising in a close-coupling formulation of quantum mechanical scattering problems. Chandra's program makes no attempt to monitor the local truncation error and leaves the choice of steplength strategy entirely to the user.

Our aim is to improve on existing implementations of de Vogelaere's method for a single second order equation by incorporating a method of truncation error estimation and an automatic mesh-selection facility. Estimates of the truncation error in de Vogelaere's method are established together with an upper bound for the local truncation error; the interval of absolute stability is found to be [-2,0] and it is shown that the global truncation error is of order h^4 where h is the steplength.

In addition the characteristics of a method due to Raptis and Allison are investigated. A numerical comparison of computer programs which incorporate the methods of de Vogelaere, Numerov, Raptis and Allison and Adams-Bashforth Adams-Moulton, with an automatic error control is performed to determine which program gives the most reliable and efficient solution of the single channel radial Schrödinger equation.

A modification of Chandra's program is provided which performs the numerical integration of a set of coupled second order homogeneous

differential equations using de Vogelaere's method with an automatic error control.

ACKNOWLEDGEMENTS

I am extremely grateful to my supervisor Dr. J.P.Coleman for his helpful advice and guidance throughout the course of this work. My thanks also go to Mrs. J.McGough for the secretarial assistance which she has provided.

This work was financed by a Research Grant from the Science Research Council.

CONTENTS

Chapter 6 contd.

INTRODUCTION

Recent years have seen an upward trend in the number of comparative studies of methods and their associated automatic implementations for solving systems of first order ordinary differential equations of the type

$$y_i' = f(x, y_1, y_2, ---, y_n) \quad , \quad i = 1, 2, ---, n. \qquad (1)$$

Equations of order higher than one, e.g.

$$y^{(r)} = f(x, y, y', y'', ---, y^{(r-1)}) \qquad (2)$$

can be readily reduced to a system of first order equations by use of the following set of simple substitutions,

$$\begin{aligned} y &= y_1 \\ y' &= y_1' = y_2 \\ y'' &= y_2' = y_3 \\ &\vdots \\ y^{(r-1)} &= y_{r-1}' = y_r \\ y_r' &= f(x, y_1, y_2, ---, y_r) \end{aligned} \qquad (3)$$

and the methods for solving equation (1) are then immediately applicable to solving (3).

Special differential equations of the form

$$y'' = f(x, y) \qquad (4)$$

and systems of such equations arise in a variety of physical contexts. In atomic and nuclear scattering problems we are interested in solving sets of coupled integrodifferential equations of the general form

$$\left\{ \frac{d^2}{dx^2} + k_i^2 - \frac{\ell_i(\ell_i+1)}{x^2} \right\} y_i = \sum_{j=1}^{N} V_{ij}(x) y_j(x) + \sum_{j=1}^{N} \int_0^{\infty} K_{ij}(x', x) y_j(x') dx', \quad i = 1, 2, ---, N. \qquad (5)$$

In the close coupling approximation equation (5) reduces to a system of coupled second order differential equations in the case of no exchange, that is when $K_{ij} = 0$ for $i, j = 1, 2, ---, N$ and our study relates to the numerical solution of such a system, in particular to the 'single channel'

equation when i, j=1. Even in the presence of exchange the integro-differential equations can be replaced by a larger set of coupled second order ordinary differential equations.

Despite the frequent occurrence of such systems of equations there appears to be a total lack of comparative studies of automatic methods for their solution. Indeed, the use of automatic methods for the solution of such equations seems to be an area which has been entirely neglected; programs which are currently available for solving (5) make no attempt at controlling the local truncation error which arises from the method used in the integration, and changes in steplength are made entirely at the discretion of the user.

In Chapter 1 we give a discussion of the radial Schrödinger equation. Some comparisons at fixed steplength have appeared using the methods of de Vogelaere (1955), Numerov and Runge Kutta in the main, and these are discussed in this Chapter. An argument in favour of the need for automatic techniques is also put forward.

A number of direct methods exist for solving (4); the use of such methods is more intuitively appealing than those methods in which (4) is reduced to a system of first order equations thereby introducing first derivatives into the equations and perhaps giving rise to a less efficient method of solution. In atomic collision processes, the step by step integration of (5) consumes a large proportion of the total computing time and it seems appropriate to make use of methods designed specifically to cope with second order differential equations with first derivative missing. A study by Ash (1969) of the asymptotic errors produced by the use of linear multistep methods as applied to solving the equivalent first order system and direct methods for solving (4) leads him to recommend the use of direct methods for such problems. We note here that if a linear multistep method designed to solve $y' = f(x, y)$ has truncation error proportional to h^{p+1} where

h is the steplength then the global error is proportional to h^p; by contrast a linear multistep method for the solution of $y'' = f(x, y)$ with truncation error proportional to h^{p+2} has global error proportional to h^p. Both methods are said to have order p.

A major aim of this work is to improve on existing implementations of de Vogelaere's method which is a fourth order step by step method for solving (4) and in Chapter 2 we give a detailed study of the method. De Vogelaere's method has a truncation error which is proportional to h^5 and we shall see that the global error is proportional to h^4. We also develop truncation error estimates which lead to an efficient automatic error control in computations based on de Vogelaere's method. The implementation of de Vogelaere's method with automatic error control is discussed in Chapter 3 and we give a description of the test program along with the test runs performed.

We have also studied Numerov's method together with the method of Raptis and Allison (1977) for solving the single channel radial Schrödinger equation, and these methods together with their automatic implementations are described in Chapters 4 and 5 respectively.

Chapter 6 describes the N.A.G. routine D02AHF which uses a variable-step variable-order Adams method to solve a system of first order differential equations and in Chapter 7 we present a numerical comparison of the methods of Chapters 2-6 for solving the single channel radial Schrödinger equation.

In Chapter 8 we provide a modification of Chandra's (1973) program; the modified version is designed to solve a system of coupled homogeneous second order differential equations. Chandra's program uses de Vogelaere's method to solve the differential equations over a fixed mesh; by inserting into Chandra's program our own routine for de Vogelaere's method which has an inbuilt automatic control on the local truncation error per unit step we hope to improve on the efficiency

(with respect to the number of function evaluations performed in the numerical integration stage) of the calculation.

CHAPTER 1

The radial Schrodinger equation

§ 1.1 The form of the equation

Scattering experiments provide valuable information concerning properties of quantum mechanical systems and such experiments study the effect of directing a monochromatic or monoenergetic beam of particles at a target. In practice the incident beam is collimated by a series of slits and the products of the collision which result from the interaction of the incident beam with the atoms of the target are measured by some form of detector.

The time independent Schrödinger equation for the total system of incident particles of type 1 colliding with particles of type 2 is

$$(H - E)\,\Psi(\underline{x}_1, \underline{x}_2) = 0 \qquad (1.1)$$

for structureless particles where E is the energy of the system and H is the Hamiltonian given by

$$H = \frac{-\hbar^2}{2m_1}\nabla_1^2 - \frac{\hbar^2}{2m_2}\nabla_2^2 + V_{12}(\underline{x}_1 - \underline{x}_2) \qquad (1.2)$$

where m_1 and m_2 are the masses. $\hbar = \frac{h}{2\pi}$ where h is Planck's constant and V_{12} is the real potential energy of interaction which is a function of the relative position of the interacting bodies. Ψ is the wave function which describes the motion of the scattered particles and predictions about the position of the particles can be made by calculating $|\Psi|^2$; Ψ itself cannot be measured directly in the scattering experiment.

If we define the relative co-ordinate $\underline{x}$ by

$$\underline{x} = \underline{x}_1 - \underline{x}_2$$

and the centre of mass co-ordinate $\underline{X}$ by

$$\underline{X} = \frac{m_1 \underline{x}_1 + m_2 \underline{x}_2}{(m_1 + m_2)}$$

then we can write equation (1.1) using (1.2) as

$$\left[-\frac{\hbar^2}{2M} \nabla_X^2 - \frac{\hbar^2}{2m} \nabla_x^2 + V_{12}(x) - E \right] \Psi(\underline{x}_1, \underline{x}_2) = 0 \qquad (1.3)$$

where we have assumed that the interaction potential is spherically symmetric. $M = m_1 + m_2$ is the total mass and $m = \frac{m_1 m_2}{m_1+m_2}$ is the reduced mass of the system. If now we use separation of variables to write $\Psi(\underline{x}_1, \underline{x}_2)$ as

$$\Psi(\underline{x}_1, \underline{x}_2) = \phi(\underline{X}) \psi(\underline{x})$$

we obtain the pair of equations

$$-\frac{\hbar^2}{2M} \nabla_X^2 \phi(X) = E_c \phi(X) \qquad (1.4)$$

$$\left\{ -\frac{\hbar^2}{2m} \nabla_x^2 + V_{12}(x) \right\} \psi(\underline{x}) = E_r \psi(\underline{x}) \qquad (1.5)$$

where $E = E_c + E_r$. (1.4) and (1.5) represent the Schrödinger equations for the centre of mass motion and the relative motion respectively and it is the solution of (1.5) in which we are interested.

Equation (1.5) must be solved subject to two boundary conditions the first of which specifies the solutions to be regular at the origin; the second specifies an asymptotic condition which represents the solution as an incoming plane wave and an outgoing scattered radial wave. The boundary conditions are thus given by

$$\psi(\underline{0}) = 0 \qquad (1.6)$$

$$\psi(\underline{x}) \underset{x\to\infty}{\sim} e^{i\underline{k}.\underline{x}} + \frac{e^{ikx}}{x} f(\theta, \phi) \qquad (1.7)$$

where f is the scattering amplitude which is a function of θ and ϕ, the polar angles of $\underline{x}$. The method of partial waves can be used to convert the partial differential equation given by (1.5) into a set of ordinary differential equations. Since we have assumed that

the interaction potential is spherically symmetric the wave equation (1.5) can be separated in spherical co-ordinates (x,θ,ϕ). In spherical co-ordinates the Laplacian operator ∇^2 may be written as

$$\nabla^2 = \frac{1}{x^2}\frac{\partial}{\partial x}\left(x^2\frac{\partial}{\partial x}\right) - \frac{1}{\hbar^2 x^2} L^2(\theta,\phi)$$

where

$$L^2(\theta,\phi) = -\hbar^2\left[\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\phi^2}\right].$$

The operator L^2 represents the square of the orbital angular momentum and the spherical harmonics $Y_{\ell m}(\theta,\phi)$ are defined to be the eigenfunctions of L^2 and L_z which is the z component of $\underline{L}$ given by

$$L_z = -i\hbar\frac{\partial}{\partial\phi}.$$

We have

$$L^2 Y_{\ell m}(\theta,\phi) = \ell(\ell+1)\hbar^2 Y_{\ell m}(\theta,\phi)$$

and

$$L_z Y_{\ell m}(\theta,\phi) = m\hbar Y_{\ell m}(\theta,\phi)$$

where ℓ, m are quantum numbers with $\ell \geq |m|$. The function

$$\psi_{\ell m}(\underline{x}) \equiv \frac{y(x)}{x} Y_{\ell m}(\theta,\phi)$$

is then a solution of (1.5) provided that y(x) is a solution of the following radial equation

$$\frac{d^2}{dx^2}y(x) = \left[\frac{\ell(\ell+1)}{x^2} - k^2 + V(x)\right] y(x) \qquad (1.8)$$

where $k^2 = \frac{2m}{h^2} E_r$ is the energy of the projectile in rydbergs and $V(x) = \frac{2m}{h^2} V_{12}(x)$.

We introduce now the notion of channels. Thus far we have considered processes in which two structureless particles undergo an elastic collision. We now consider the process in which an electron is fired at a hydrogen atom. The outcome of such a process may be any one of the following:

$$e^- + H \longrightarrow \begin{cases} (i) & e^- + H \\ (ii) & e^- + H^* \\ (iii) & e^- + e^- + p^+ \end{cases}$$

where (i) represents elastic scattering in which the projectile and the target remain in their original states, (ii) represents inelastic scattering in which the target remains in an excited state after collision where H^* denotes an excited state of the hydrogen atom, and (iii) represents the ionisation of the hydrogen atom. Each of the possible outcomes is referred to as a channel and a process of the type above is referred to as a many channel collision. In processes such as the low-energy scattering of electrons off protons the only possible outcome is elastic scattering,

$$e^- + p^+ \longrightarrow e^- + p^+$$

and such processes are referred to as single channel processes.

Equation (1.8) is an example of a single channel equation and it must be solved subject to two boundary conditions the first of which requires that the solution be regular at the origin, that is

$$y(0) = 0 . \qquad (1.9)$$

If the potential V(x) is negligible for values of x greater than some r_o (in the so called asymptotic region) then (1.8) effectively reduces to

$$\frac{d^2y}{dx^2} = [\frac{\ell(\ell+1)}{x^2} - k^2]\ y(x) \qquad (1.10)$$

for which the solutions are $kxj_\ell(kx)$ and $kxy_\ell(kx)$ where $j_\ell(kx)$ and $y_\ell(kx)$ are the spherical Bessel functions of the first and second kind. For large values of $x > r_o$, y must be a linear combination of the two independent solutions of (1.10). Thus

$$y(x) \underset{x\to\infty}{\sim} Akxj_\ell(kx) - Bkxy_\ell(kx)$$

where A and B are dependent on k and we have

$$kxj_\ell(kx) \underset{x\to\infty}{\sim} \sin(kx-\tfrac{1}{2}\ell\pi)$$

and

$$kxy_\ell(kx) \underset{x\to\infty}{\sim} -\cos(kx-\tfrac{1}{2}\ell\pi).$$

Therefore

$$y(x) \underset{x\to\infty}{\sim} A\sin(kx-\tfrac{1}{2}\ell\pi) + B\cos(kx-\tfrac{1}{2}\ell\pi)$$

or

$$y(x) \underset{x\to\infty}{\sim} C\sin(kx-\tfrac{1}{2}\ell\pi+\delta_\ell) \qquad (1.11)$$

where $\tan\delta_\ell = \frac{B}{A}$. The second boundary condition is given by (1.11). The angle δ_ℓ is the (real elastic) scattering phase shift of the ℓ^{th} partial wave introduced by the potential V(x), and the calculation of this quantity leads to the determination for example of the scattering amplitude and scattering cross section. We write

$$R_\ell = \tan\delta_\ell$$

where we think of R_ℓ as a one-by-one matrix; in the study of inelastic scattering by a target system there will be many channels describing the excitation of internal states of the system, in which case the so called reactance matrix R will be a higher dimensional square matrix. The constant C in (1.11) is a normalisation constant which specifies the asymptotic amplitude and which together with δ_ℓ completely determines the solution y(x). Thus the phase shift is evaluated by numerically integrating the differential equation for the radial function and examining its behaviour in the asymptotic region. Since equation (1.8) is homogeneous in y(x), a solution which is multiplied by an arbitrary factor will also be a solution of (1.8); thus the constant C is arbitrary and we can fix it by setting $\frac{dy}{dx}\Big|_{x_0}$ = constant, where x_0 is the starting point of the numerical integration.

In a many channel problem where there are N possible final states resulting from a given initial state of the system, we are required to solve a set of coupled second order differential equations of the form

$$\frac{d^2}{dx^2} y_\mu(x) = \sum_{\nu=1}^{N} \left[\frac{l_\mu(l_\mu+1)}{x^2} \delta_{\mu\nu} - k_\mu^2 \delta_{\mu\nu} + V_{\mu\nu}(x) \right] y_\nu(x), \qquad \mu = 1, 2, \cdots, N \qquad (1.12)$$

where $\delta_{\mu\nu}$ is the kronecker delta and the solution $y_\mu(x)$ is the radial wave function of the projectile in the μth scattering channel. For an incident beam of a given energy it is possible that this energy will be sufficient to excite some of the states allowed in the eigenfunction expansion but insufficient to excite higher states and depending on whether k_μ^2 is positive or negative we refer to the μth channel as being open or closed respectively. For physically meaningful collision processes k_μ^2 must be positive; for solutions which arise from negative values of k_μ^2 the general solution of the appropriate Schrödinger equation in the asymptotic region will involve a linear combination of increasing and decreasing exponentials. However as the radial distance increases the exponentially increasing solution dominates the contribution to the solution from the decreasing exponential and such a solution would be physically meaningless. Thus any physically significant solution of the radial equation for $k_\mu^2 < 0$ must have the form of a decreasing exponential in the asymptotic region. Suppose that there are n_a open channels corresponding to $k_\mu^2 > 0$, $\mu = 1, \cdots, n_a$ and n_b closed channels corresponding to $k_\mu^2 < 0$, $\mu = n_a + 1, \cdots, n_a + n_b$ where $n_a + n_b = N$. Then N solutions must be found to satisfy the appropriate boundary conditions; in particular each solution must be matched to the correct asymptotic form, according as k_μ^2 is positive or negative.

In this work we are interested in solving (1.10) for all channels open, that is, subject to the boundary conditions

$$y_{\mu\nu}(0) = 0 \tag{1.13}$$

$$y_{\mu\nu}(x) \underset{x\to\infty}{\sim} \delta_{\mu\nu} \sin\left(k_\mu x - \tfrac{1}{2}\ell_\mu \pi\right) + R_{\mu\nu} \cos\left(k_\mu x - \tfrac{1}{2}\ell_\mu \pi\right) \tag{1.14}$$

where we have added another subscript to the solution; for an incident wave in channel ν the wave function for a wave scattered in channel μ has the asymptotic form given by (1.14). R is the (symmetric) reactance matrix from which valuable information (e.g. scattering amplitudes and cross sections) regarding the nature of the scattering process can be extracted, and the calculation of R is the object of most calculations for atomic collision processes. From R we can readily evaluate the (unitary) scattering and transition matrices, the so called S and T matrices, using the following formulae

$$S = (I - iR)^{-1}(I + iR)$$

$$T = (I - iR)^{-1} R \quad .$$

The S and T matrix asymptotic forms are given respectively by

$$y_{\mu\nu}(x) \underset{x\to\infty}{\sim} \delta_{\mu\nu} e^{-i(k_\mu x - \frac{1}{2}\ell_\mu\pi)} - S_{\mu\nu} e^{i(k_\mu x - \frac{1}{2}\ell_\mu\pi)}$$

and

$$y_{\mu\nu}(x) \underset{x\to\infty}{\sim} \delta_{\mu\nu} \sin\left(k_\mu x - \tfrac{1}{2}\ell_\mu\pi\right) + T_{\mu\nu} e^{i(k_\mu x - \frac{1}{2}\ell_\mu\pi)} \quad .$$

In our numerical calculations we work with real rather than complex numbers so that we shall use the asymptotic boundary condition given by equation (1.14).

§ 1.2 Numerical solution of the single channel equation

The calculation of scattering phase shifts by integrating a single channel Schrödinger equation generally involves three stages:

(i) unless the differential equation is self-starting suitable

values must be generated,

(ii) the differential equation is solved in a step by step fashion as far as some r_o, which may perhaps be determined by the program, and finally

(iii) a phase shift is extracted.

Each stage of the calculation may be effected by a wide variety of different methods. For example, starting values may be provided by Taylor expansion or by the initial use of a self-starting numerical technique such as a Runge-Kutta method, and phase shifts may be extracted by fitting the numerical solution to asymptotic forms of various degrees of complexity.

We consider the (real) potential function $V(x)$ to have the following expansion in the vicinity of the origin

$$V(x) = \frac{V_1}{x} + V_2 + V_3 x + \text{---} .$$

The potential function is such that

$$x^2 V(x) \xrightarrow[x \to 0]{} 0$$

and at small values of x the solution can be expanded in the following power series

$$y(x) = \sum_{r=o}^{\infty} a_r x^{r+s} .$$

Thus if we substitute the above into (1.8) we arrive at the following set of equations

$$\sum_{r=o}^{\infty} \{(r+s)(r+s-1)a_r x^{r+s} - \ell(\ell+1)\, a_r x^{r+s} + k^2 a_r x^{r+s+2} - V_1 a_r x^{r+s+1} - V_2 a_r x^{r+s+2} - V_3 a_r x^{r+s+3} - \text{---}\} = 0 .$$

If we equate coefficients of the powers of x^{r+s} and set r to be zero we obtain the indicial equation

$$[\, s(s-1) - \ell(\ell+1)]\, a_o = 0.$$

For $a_o \neq 0$ we have

$$s = -\ell \text{ or } \ell + 1$$

and we reject the choice of $s = -\ell$ since we want physically significant solutions and we require the solution to be regular at the origin. Thus the regular solution of the radial Schrödinger equation given by (1.8) may be expressed as

$$y(x) = x^{\ell+1}[a_o + a_1 x + a_2 x^2 + \text{---}]$$

for sufficiently small x, and the coefficients are given by the equations

$$\begin{aligned} 2(\ell+1)a_1 &= V_1 a_o \\ 2(2\ell+3)a_2 &= V_1 a_1 + (V_2 - k^2)a_o \\ i(2\ell+i+1)a_i &= V_1 a_{i-1} + (V_2 - k^2)a_{i-2} + \sum_{j=3}^{i} V_j a_{i-j}, \quad i > 2 \end{aligned} \tag{1.15}$$

From the series for $y(x)$ an expansion for $y'(x)$ is readily obtained; thus if x_o, the starting point of the integration, is provided it is possible to calculate $y(x_o)$ and $y'(x_o)$ sufficiently accurately in order to start the integration stage of the calculation.

The second stage of the calculation may now be entered. This is the stage of the calculation with which we are mainly concerned, in which some numerical method is employed for the step by step integration of the radial Schrödinger equation from the starting point into the asymptotic region where the effect of the potential becomes negligible. We shall carry out in later chapters a detailed comparison of a number of methods for solving (1.8) all of which incorporate an automatic steplength selection thus giving an error control based on an estimate at each step of the integration of the truncation error per unit step.

The final stage of the calculation involves evaluating the phase shift. If $y(x)$ is represented sufficiently accurately by the asymp-

totic form (1.11) when x exceeds some distance R then, for any x_1 and x_2 greater than R, we have

$$\tan \delta_\ell = \frac{y(x_2)\left\{kx_1 j_\ell(kx_1)\right\} - y(x_1)\left\{kx_2 j_\ell(kx_2)\right\}}{y(x_1)\left\{kx_2 y_\ell(kx_2)\right\} - y(x_2)\left\{kx_1 y_\ell(kx_1)\right\}} . \quad (1.16)$$

There are a variety of techniques some more elaborate than others for calculating the phase shift, but as we are concerned primarily with the second stage of the calculation, that is, the numerical integration of the radial Schrödinger equation, we choose to make use of the straight forward method given by (1.16) above for calculating the phase shift. The phase shift thus obtained is determined to within integer multiples of π. Consideration should be given to how a specified accuracy of the phase shift can be achieved in this part of the calculation and we shall compare the relative efficiencies of the various numerical methods which we test in calculating the phase shift of a given problem to a required accuracy.

§ 1.3 Numerical solution of a system of coupled differential equations

In solving (1.12)-(1.14) we are concerned with the provision of suitable starting values, the integration of the system of differential equations and the calculation of the reactance matrix R.

Since we are solving N second order differential equations and each solution has two integration constants it is necessary to satisfy 2N boundary conditions before the second stage of the calculation may be entered, namely the numerical integration of the system of differential equations. N boundary conditions may be satisfied by requiring that equation (1.13) holds, that is, by requiring that the solutions be regular at the origin, and the remaining N boundary conditions may be taken to be the values of the first derivatives at the origin. Alternatively if the integration is started at x_o, in

the vicinity of the origin, then use can be made of the known starting series for the solution and its first derivative to provide the necessary 2N boundary conditions. However, the specification of these 2N conditions will give rise to one solution only and since the general solution of (1.12) involves N arbitrary constants it is necessary to specify N different choices of the initial boundary conditions. It is also important to note that these starting conditions which satisfy (1.13) will not in general lead to solutions which satisfy the asymptotic boundary condition given by (1.14). Thus we need to generate N linearly independent solutions of (1.12). The N different sets of starting conditions can be represented by a (2N x N) matrix $\propto$, the columns of which are linearly independent. The general assumption is that if the columns of $\propto$ are linearly independent then the respective asymptotic forms will also be linearly independent. However, due to the finite word length of computers it may be the case that this linear independence is not maintained during the course of the integration. We shall discuss this point further in Chapter 8. The ith column of $\propto$ contains 2N elements of which (2N-2) elements are zeros and the (2i-1)th and (2i)th elements correspond to the values of the ith component of the solution and its derivative respectively at the origin (or alternatively at x_o).

The system of equations can then be integrated step by step N different times for each vector of starting conditions, out into the asymptotic region where the solution is then matched to the correct asymptotic form, and the R matrix is calculated.

§ 1.4 Some recent comparative studies at fixed steplength

Some comparisons of algorithms for the solution of coupled second order differential equations of the form (1.12) have appeared

in which numerical methods of integration are used over a fixed mesh. Some use of stepchanging facilities has been made with regard to problems where the integration mesh is divided into n predetermined regions where the ith region uses a steplength of $2^{(i-1)}h$; thus changes in steplength occur only when stepping from one region to another at which point the steplength is doubled. No regard is paid to the estimation of the local truncation error of the method and to its subsequent control.

It is important to establish the criteria which are to be considered before any comparison between the methods is made and it is equally important to appreciate that any conclusions drawn as to the relative performance of the various algorithms must necessarily reflect the performance of the computer code as a whole and not just the method of solution. Different implementations of the same method might lead us to reformulate our views on the performance of that particular method.

In comparing different algorithms for the solution of (1.12) one is generally interested in the reliability and efficiency of such algorithms. The reliability of an algorithm is reflected by how well the numerical solution approximates the true solution; the efficiency of an algorithm is measured by such things as the number of function evaluations required to achieve specified accuracies in the solution, the storage required by the computer in solving the system of differential equations and the overhead which is a measure of the computer time required to solve the problem independent of the time required for performing the necessary function evaluations.

One of the earliest comparisons of numerical methods for the solution of the radial Schrödinger equation, undertaken by Froese (1963), considered four different methods given below.

(a) 4th order Runge-Kutta method

$$y'_{n+1} = y'_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$y_{n+1} = y_n + hy_n' + \frac{h^2}{6}(k_1 + k_2 + k_3)$$

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + \frac{h}{2},\ y_n + \frac{h}{2} y_n')$$

$$k_3 = f(x_n + \frac{h}{2},\ y_n + \frac{h}{2} y_n' + \frac{h^2}{4}k_1)$$

$$k_4 = f(x_n + h,\ y_n + hy_n' + \frac{h^2}{2} k_2)$$

(b) Nystrom method

$$y_{n+1}' = y_n' + \frac{h}{6}(k_1 + 4k_2 + k_3)$$

$$y_{n+1} = y_n + hy_n' + \frac{h^2}{6}(k_1 + 2k_2)$$

$$k_1 = f(x_n,\ y_n)$$

$$k_2 = f(x_n + \frac{h}{2},\ y_n + \frac{h}{2} y_n' + \frac{h^2}{8} k_1)$$

$$k_3 = f(x_n + h,\ y_n + hy_n' + \frac{h^2}{2}k_2)$$

(c) Kutta - $\int^2$ method

$$y_{n+1} = 2y_n - y_{n-1} + \frac{h^2}{12}(k_1 + 10k_2 + k_3)$$

$$k_1 = f(x_{n-1},\ y_{n-1})$$

$$k_2 = f(x_n,\ y_n)$$

$$k_3 = f(x_n + h,\ 2y_n - y_{n-1} + h^2k_2)$$

(d) Numerov's method

$$y_{n+1} = (2y_n - y_{n-1} + \frac{h^2}{12}(10y_n'' + y_{n-1}''))/(1 - \frac{h^2}{12}f_{n+1})$$

For a detailed discussion of Numerov's method see Chapter 4.

Froese studied the relative accuracy and efficiency of these methods in solving the single channel radial Schrödinger equation for atomic hydrogen using a fixed steplength throughout and concluded that the relative accuracies of the methods are in agreement to within

a factor of ten and that the Numerov method is 'best' in the sense that it requires only 1 function evaluation per step compared to 4, 3 and 2 function evaluations respectively for methods (a), (b) and (c). Methods (b), (c) and (d) all take account of the special form of the differential equation and we would expect these methods to lead to a more efficient solution of the differential equation. Reference to method (b) is made in a discussion by Scraton (1964) concerning numerical methods of solution for the differential equation $y'' = f(x, y)$.

Blatt (1967) asserted that the Numerov method is clearly superior to the fourth order Runge Kutta method because of the higher order local truncation error in the Numerov method which is proportional to h^6 where h is the steplength; the local truncation error in the fourth order Runge Kutta method is proportional to h^5. Sloan (1968) subsequently pointed out that Blatt's conclusion is unjustified and showed that the cumulative errors in both methods are proportional to h^4, using an approach suggested earlier by Kopal (1955). A comparison of the actual errors obtained using the methods of Numerov and Nystrom in solving

$$y''(x) = -y(x)$$

with the initial conditions

$$y(0) = 0,\ y'(0) = 1$$

for a fixed steplength was made; the errors are easily found by comparing the numerical solution with the analytic solution $y(x) = \sin x$. The comparison showed that the Numerov method is only slightly superior if the same steplength is used in both methods, but that it is clearly superior when the comparison between Numerov with steplength h and Nystrom with steplength 2h is made. This is a more reasonable comparison since the Nystrom method requires function values at the half-

way points.

Lester (1968) and Lester and Bernstein (1968) have solved a system of coupled second order differential equations of the form (1.12) using the method of de Vogelaere (1955), which is a fourth order step by step method (with local truncation error proportional to h^5) for solving $y'' = f(x, y)$. For the system of equations

$$\frac{d^2}{dx^2} y_i(x) = f_i(x, y_1(x), y_2(x), ---, y_N(x)), \quad i = 1, 2, ---, N$$

the algorithm performs the integration from x_n to x_{n+1} by cyclic use of the equations

$$y_{i,\frac{1}{2}} = y_{i,o} + \frac{h}{2} y'_{i,o} + \frac{h^2}{24} (4f_{i,o} - f_{i,-\frac{1}{2}}),$$

$$y_{i,1} = y_{i,o} + hy'_{i,o} + \frac{h^2}{6}(f_{i,o} + 2f_{i,\frac{1}{2}}),$$

$$y'_{i,1} = y'_{i,o} + \frac{h}{6} (f_{i,o} + 4f_{i,\frac{1}{2}} + f_{i,1}),$$

in the notation of Lester (1971) where

$$y_{i,s} \equiv y_i(x + sh),$$

$$y'_{i,s} \equiv \frac{dy_i}{dx}(x + sh),$$

$$f_{i,s} \equiv f_i(x + sh, y_{1,s}, y_{2,s}, ---, y_{N,s})$$

and $h = x_{n+1} - x_n$. The neglected terms are of order h^4, h^5 and h^5 respectively. Lester chooses to neglect a comparison with the well used Numerov method on the basis that the Numerov method requires separate procedures for both starting the integration and changing the steplength. The integration using de Vogelaere's method however presents no such problems and this method is compared for a fixed steplength with the fourth order Runge-Kutta-Gill (Hildebrand, 1956) procedure, referred to henceforth as the RKG procedure, for the problem of two coupled harmonic oscillators for which the analytical solutions are available. The method of de Vogelaere is found to

be faster than the RKG process by a factor of approximately two and it is noted that the de Vogelaere algorithm performs two function evaluations per step compared to four in the RKG process. It is not clear however whether any stepchanging was in fact performed during the execution of de Vogelaere's method and this comment is equally applicable to a comparative study by Allison (1970) of de Vogelaere's method with the matrix and iterative Numerov methods for the calculation of cross sections for the rotational excitation of molecular hydrogen by heavy particle impact.

Allison's study concludes that the relative speeds of the methods to calculate the square of the modulus of the S matrix for sets of 4, 9 and 16 coupled equations are in the order:

matrix Numerov $<$ de Vogelaere $<$ iterative Numerov,

and that the accuracy of the methods for this calculation is best for the matrix Numerov method followed by the de Vogelaere and iterative Numerov methods. The range of integration for the comparison of the methods consists of two regions, the second of which uses a steplength which is twice the value of that used in the first region, corresponding to the usual practice of using a predetermined mesh.

Raptis and Allison (1977) have examined a method which is essentially the Numerov method in the nonclassical region, that is for $k^2 - V < 0$, but which solves the radial Schrödinger equation more efficiently in the classical region, that is for $k^2 - V > 0$ by exploiting the a priori knowledge of the asymptotic form of the solution. This method is compared with the Numerov method in solving the single channel radial Schrödinger equation for a Lennard-Jones potential for a range of values of k and ℓ and phase shifts accurate to three decimal places are computed. The number of integration steps required over the range of integration using the Raptis and

Allison method is reported to be often less than half the corresponding number required using the method of Numerov. Raptis and Allison state that the truncation error of their method can be used to control the interval size but it appears to be the case that no such control was used in their study.* They also report a rapid increase over the Numerov method in the optimal interval size allowed from truncation error considerations. For a detailed discussion of the Raptis and Allison method and its automatic implementation see Chapter 5.

A recent study by O'Shea (1978) examines numerical methods for the solution of a system arising in the close-coupling formulation of the Schrödinger equation with exchange terms. The methods included in the comparison are the Numerov and de Vogelaere methods and some fourth order Runge Kutta type methods; the Runge Kutta methods consist of methods (a) and (b) studied by Froese and the classical fourth order method, as applied to a system of first order differential equations, both with and without a Richardson-type truncation estimate correction. The relative efficiency of the methods is measured by the computer time and storage requirements of the methods and on this basis the conclusion of the study is that de Vogelaere's algorithm is the most efficient. With reference to the problem of an electron colliding with an oxygen ion, O'Shea shows that the computer time taken to solve this problem using de Vogelaere's method with a steplength of 2h is less than half that taken using Numerov's method with a steplength h, with the times for the various Runge Kutta methods lying between those for de Vogelaere and Numerov. It is also shown that de Vogelaere has the minimum storage requirements, the number of arrays required being precisely half that required in the Numerov method. O'Shea states that the de Vogelaere method is very much superior to all of the other methods tested in

* The published (1978) version of their paper indicates that the steplength is automatically doubled if the truncation error estimate falls below some value.

the efficient use of computer time; we feel however that too much emphasis is placed on the ability of de Vogelaere's method to use a larger steplength of 2h compared with a steplength h in Numerov in achieving the same accuracy in calculating the phase shift. The point at which the contribution of the exchange terms can be considered insignificant is the same for all of the algorithms and it is important to note that a single step of 2h performed by de Vogelaere's method requires 2 function evaluations, the same number being required for two steps of length h in Numerov's method. Since the number of function evaluations directly affects the computer time required for the solution of the system of equations this suggests that the superiority of de Vogelaere over Numerov is a consequence of the relative computing efficiency of the two methods. It is also important to note that the matrix Numerov method (see Smith, Henry and Burke, 1966; Allison, 1970) considered by O'Shea warrants a matrix inversion at each step of the calculation; this is a penalty of the implicit nature of the method. By contrast the method of de Vogelaere is an explicit method and it may be the case that some inefficiency arises in the implementation of Numerov's method thus leading perhaps to considerably higher overheads than those encountered in the particular implementation of de Vogelaere's method.

A more detailed discussion of O'Shea's work may be found in his Ph.D. thesis (1971) where the superiority of de Vogelaere's method in the above respects is again emphasized. O'Shea states that to his knowledge 'no worthwhile investigation of the stability of de Vogelaere's method has yet been undertaken' and he suggests a particular quantity (which is readily calculated from previously calculated values of the solution and its first and second derivatives) be used 'as an ad hoc criterion for stability'. He also states,

following de Vogelaere (1955), that the same quantity may be used to give an indication of the accuracy of the calculations, but beyond recognition of this fact no further consideration is given to the matter.

Some other comparisons of methods have appeared such as the use of perturbation and Numerov methods for the solution of systems of coupled second order differential equations (see e.g. Riehl, Diestler and Wagner, 1974) in addition to some comparisons of differential and integral equation techniques for the numerical solution of the radial Schrödinger equation. In particular Stern (1977) compares the phase shifts computed for the static electron-hydrogen potential using the standard fourth order Runge Kutta method and Numerov's method with the corresponding phase shifts obtained from quadrature solutions of the (Fredholm) integral formulation of the radial Schrödinger equation; the quadrature formulae which are considered are the composite trapezoidal and Simpson rules and an n-point Gauss-Legendre formula. Stern concludes that of the three quadrature methods the composite trapezoidal rule is the most reliable particularly when computing phase shifts at very low energies and that all the methods yield phase shifts of similar accuracy provided that a sufficient number of pivotal points are used in the quadrature methods and the steplength in the Runge Kutta and Numerov methods is 'small enough'. He reports that at low energies using the methods of Runge Kutta and Numerov discrepancies arise in the computed phase shifts for different values of the steplength h (Stern considers h = (0.1, 0.05, 0.01)). He unfortunately refers to these discrepancies as 'instabilities' but we suspect that this failure of the phase shifts to agree for a particular energy and angular momentum value at different values of h is a consequence of the method used for the phase shift calculation. Stern comments that the steplength in the methods

of Runge Kutta and Numerov can be adjusted automatically until a specified accuracy is achieved but no consideration is actually given to such automatic methods on the grounds that the necessary computer time is not easy to predict. However if a particular accuracy is requested then we should reasonably expect to use more computer time particularly at the more stringent error requirements.

In summary, we have looked at a number of comparative studies involving methods applicable to solving systems of equations of the form (1.12). Although these studies are by no means comprehensive they do give some guidance as to the relative performance of various methods for a fixed steplength. Our interest lies primarily in direct numerical integration techniques; it is probably fair to say that the most widely used method for solving (1.12) is that of Numerov. The method of de Vogelaere has also been used fairly extensively (e.g. Verlet, 1967; Lester, 1968, 1971; Chandra, 1973; Basavaiah and Broom, 1975; Launay, 1976); however despite the frequent use of de Vogelaere's method we are unaware of any previous error analysis, or any study of the stability of the method.

§1.5 The need for automatic integration techniques

To our knowledge no consideration has been given to the use of fully automatic methods for solving the radial Schrödinger equation. Such methods provide an estimate of the local truncation error per step or per unit step at each step of the integration and according as this estimate is less or greater than some specified tolerance an increase or decrease in the steplength is performed.

There appears to be a general belief that in scattering problems it is valid to multiply the steplength successively by some integral factor as x increases, as for example, Chandra (1973) invites the

user of his program to do. Allison (1970) argues that in many problems the function V(x) is a rapidly varying function for small x, gradually becoming smoother and finally tending to zero for large x and that to integrate such a problem efficiently requires a steady increase in steplength. This suggests that satisfactory error control could be achieved by calculating some measure of the truncation error, and doubling the steplength whenever this quantity is less than some specified tolerance. However for scattering problems this approach is not entirely reliable. Ultimately both the calculated solution and its attendant error are oscillatory functions, and therefore it is possible to find a small local error at some point and a substantially larger error a few steps later. We contend that whatever numerical method is used automatic steplength control requires not only a procedure for increasing the steplength when this is desirable, but also a facility for steplength decrease and the ability to detect when this is necessary.

Indeed the manner in which the steplength changes during the course of the integration may provide valuable information to the user concerning the nature of the solution. When solving an initial value problem for an ordinary differential equation we are normally interested in the actual or global error, the difference between the numerical solution and the exact solution of the problem. While estimates of the global error can sometimes be obtained, for example, by two parallel integrations and some form of global extrapolation (Shampine and Watts, 1976) it is not possible in general to simultaneously estimate and control the global error. We therefore adopt the more modest goal of controlling the local error; the extent to which our control of the local error per unit step controls the global error then depends on the stability of the differential equation.

We choose to follow Hull et al (1972) and control the estimated error per unit step instead of the estimated error per step; when a differential equation is integrated over a fixed interval, a decrease in h results in an increased number of steps, and the error criterion adopted should take some account of this.

With regard to integration over a fixed range, the simplest strategy which can be used for changing the steplength is that of halving and doubling, since if h is restricted to undergo changes by factors of two, we ensure that the endpoint of the range of integration is reached without any need for interpolation. This strategy is frequently used in variable step-variable order Adams codes, and in particular in the N.A.G. routine D02AHF which solves a system of first order ordinary differential equations.

The use of a varying steplength is important in reducing the computational work with respect to the number of steps and hence the number of function evaluations which in turn leads to increased efficiency in solving the differential equation. A variety of methods for changing the steplength using linear multistep methods are discussed by Krogh (1973).

CHAPTER 2

De Vogelaere's method

§2.1 Derivation of Method

The method of de Vogelaere (1955) is a fourth order step by step process for solving

$$\frac{d^2y}{dx^2} = f(x, y) \tag{2.1}$$

subject to the initial conditions

$$y(x_o) = y_o,\ z(x_o) = z_o$$

where y_o and z_o are specified numbers and

$$z(x) = \frac{dy}{dx}.$$

The method is equally applicable to solving systems of such equations.

The general step of de Vogelaere's method, leading from x_{2n} to $x_{2n+2} = x_{2n} + 2h$, for a fixed steplength h, may be described as follows:

Given y_{2n}, z_{2n}, f_{2n} and f_{2n-1},

$$\text{(i)} \quad y_{2n+1} = y_{2n} + hz_{2n} + \frac{h^2}{6}(4f_{2n} - f_{2n-1}) \tag{2.2}$$

$$\text{(ii)} \quad f_{2n+1} = f(x_{2n+1}, y_{2n+1}) \tag{2.3}$$

$$\text{(iii)} \quad y_{2n+2} = y_{2n} + 2hz_{2n} + \frac{h^2}{3}(4f_{2n+1} + 2f_{2n}) \tag{2.4}$$

$$\text{(iv)} \quad f_{2n+2} = f(x_{2n+2}, y_{2n+2}) \tag{2.5}$$

$$\text{(v)} \quad z_{2n+2} = z_{2n} + \frac{h}{3}(f_{2n} + 4f_{2n+1} + f_{2n+2}) \tag{2.6}$$

where y_n and z_n are approximations to the exact solution $y(x_n)$ and its first derivative $y'(x_n)$ at the mesh point x_n. The local truncation errors in y_{2n+2} and z_{2n+2} are of order h^5 and that in y_{2n+1} is of order h^4. De Vogelaere's method makes use of an intermediate point at the mid-point of the interval $[x_{2n}, x_{2n+2}]$; the solution is predicted at this

point to third order in equation (2.2) and the predicted value is then used in the corrector equation (2.4) to obtain the solution correct to fourth order at the end of the interval.

We follow de Vogelaere and give a derivation of the method in terms of finite difference expansions. For a derivation based on Taylor expansions the interested reader is referred to Coleman and Mohamed (1978). We set up the machinery by introducing the notation

$$^{(n)}f = \int_o^x {}^{(n-1)}f\,dx, \qquad {}^{(o)}f = f, \quad n = 1, 2$$

and we see that

$$\int_o^x \int_o^x f(x)dx^2 = \int_o^x \left\{ {}^{(1)}f(x) - {}^{(1)}f(0) \right\} dx$$
$$= {}^{(2)}f(x) - {}^{(2)}f(0) - {}^{(1)}f(0)x.$$

(2.1) may be written as

$$y = {}^{(2)}f$$

and we also have

$$\frac{dy}{dx} = {}^{(1)}f = z .$$

We can express f(x), using Newton's forward difference formula, as

$$f(x) = f_o + \frac{x}{h}\Delta f_o + \frac{x(x-h)}{2!h^2}\Delta^2 f_o + \frac{x(x-h)(x-2h)}{3!\,h^3}\Delta^3 f_o + \text{---}$$

and substituting x = uh leads to the formula

$$f(uh) = f_o + u\Delta f_o + \frac{u(u-1)}{2!}\Delta^2 f_o + \frac{u(u-1)(u-2)}{3!}\Delta^3 f_o + \text{---} \quad .$$

Thus

$$^{(2)}f = h^2 \int_o^u \int_o^u f(uh)du^2$$
$$= h^2 \left\{ f_o \frac{u^2}{2} + \Delta f_o \frac{u^3}{6} + \Delta^2 f_o \frac{(u^4-2u^3)}{24} + \Delta^3 f_o \frac{(3u^5-15u^4+20u^3)}{360} + \text{---} \right\}.$$

This may be written in the form

$$\int_o^x \int_o^x f(x)dx^2 = h^2 \sum_{p=o}^{\infty} (-1)^p P_{2,p}(-u)\Delta^p f_o \qquad (2.7)$$

where x = uh, $P_{2,o}(u) = \frac{u^2}{2}$; $P_{2,1}(u) = \frac{u^3}{6}$; $P_{2,2}(u) = \frac{(2u^3+u^4)}{24}$;

$$P_{2,3}(u) = \frac{(20u^3 + 15u^4 + 3u^5)}{360}; \text{ ---}$$

Similarly, Newton's backward difference formula, expressed as

$$f(x) = f_o + \frac{x}{h}\,\Delta f_{-1} + \frac{x(x+h)}{2!h^2}\,\Delta^2 f_{-2} + \frac{x(x+h)(x+2h)}{3!h^3}\,\Delta^3 f_{-3} + \text{---}$$

leads to

$$\int_o^x \int_o^x f(x)dx^2 = h^2 \sum_{p=o}^{\infty} Q_{2,p}(u)\,\Delta^p f_{-p} \qquad (2.8)$$

where $x = uh$, $Q_{2,p}(u) = P_{2,p}(u)$ for $p \geq 0$. By including terms up to fourth order in the forward difference form for $^{(2)}f$, and then differentiating with respect to u and substituting the appropriate value for u we obtain

$$^{(1)}f_2 - {}^{(1)}f_o = h(2f_1 + \frac{1}{3}\Delta^2 f_o - \frac{1}{90}\Delta^4 f_{-1} + \text{---}) \cdot \qquad (2.9)$$

It is now possible to derive de Vogelaere's method by integrating Newton's difference formulae and considering various truncations of the expansions (2.7), (2.8).

When $x = h$, corresponding to $u = 1$, and Δf_{-1} is neglected in (2.8) we have

$$^{(2)}f_1 - {}^{(2)}f_o - {}^{(1)}f_o h = h^2\,(\tfrac{1}{2}f_o)$$

or

$$\tilde{y}_1 = y_o + hz_o + \frac{h^2}{2} f_o \qquad (2.10)$$

where $\tilde{y}_1$ signifies an approximation to $y(x_1)$ of second order.

Similarly when $x = h$ and $\Delta^2 f_o$ is neglected in (2.7) we have

$$^{(2)}f_1 - {}^{(2)}f_o - {}^{(1)}f_o h = h^2[\tfrac{1}{2}f_o + \tfrac{1}{6}\Delta f_o]$$

or

$$\hat{\tilde{y}}_1 = y_o + hz_o + \frac{h^2}{6}(2f_o + f_1) \qquad (2.11)$$

where $\hat{\tilde{y}}_1$ represents an approximation to $y(x_1)$ of third order.

Another third order approximation to $y(x_1)$ may be obtained by neglecting $\Delta^2 f_{-2}$ when $x = h$ in (2.8). Then

$$^{(2)}f_1 - {}^{(2)}f_o - {}^{(1)}f_o h = h^2\,[\tfrac{1}{2}f_o + \tfrac{1}{6}\Delta f_{-1}]$$

or

$$y_1 = y_o + hz_o + \frac{h^2}{6}(-f_{-1} + 4f_o).$$

The above formula corresponds to (2.2) for n = 0.

By neglecting $\Delta^3 f_o$ when x = 2h in (2.7) we have

$$(2)f_2 - {}^{(2)}f_o - {}^{(1)}f_o 2h = h^2 [2f_o + \frac{4}{3}\Delta f_o + 0.\Delta^2 f_o]$$

or

$$y_2 = y_o + 2hz_o + \frac{h^2}{3}(2f_o + 4f_1)$$

which is a fourth order approximation to $y(x_2)$; the formula for y_2 corresponds to (2.4) for n = 0.

Finally by neglecting $\Delta^4 f_{-1}$ in (2.9) we arrive at

$$z_2 = z_o + \frac{h}{3}(f_o + 4f_1 + f_2)$$

which is the well known Simpson rule; the formula corresponds to (2.6) for n = 0.

The starting value of the independent variable is taken to be zero and we regard each step to be comprised of two intervals of length h. We assume that f_{2n-1} is known to third order and that y_{2n}, z_{2n}, f_{2n} are known to fourth order. The general step leading from x_{2n} to $x_{2n+2} = x_{2n}+2h$ is then given by (2.2) - (2.6), in which y_{2n+1}, f_{2n+1} are calculated to third order and y_{2n+2}, z_{2n+2}, f_{2n+2} are calculated to fourth order. These values then serve as initial values for the next step and cyclic use of (2.2) - (2.6) enables the solution to be obtained over the range of integration.

It only remains to start the method. The values of y_o and z_o alone are insufficient to start the calculation and de Vogelaere has suggested two separate methods for providing the additional starting value; f_o is computed from (2.1).

Method (a) : A second order approximation $\hat{y}_1$ is calculated using (2.10) and the corresponding function value $\hat{f}_1 = f(x_1, \hat{y}_1)$ is calculated

(also to second order) by use of equation (2.1). Equation (2.11) may then be used with $\hat{f}_1$ replacing f_1 to give a third order approximation $\hat{\hat{y}}_1$ to $y(x_1)$ which can now be used in (2.3) - (2.6) with n=0.

Method (b) : A second order approximation for $y(x_{-1})$ can be calculated using the backwards form of (2.9):

$$y_{-1} = y_o - hz_o + \frac{h^2}{2} f_o \qquad (2.12)$$

and the corresponding function value $f_{-1} = f(x_o-h, y_{-1})$ is calculated to the same order using (2.1). It is now possible to use (2.2) - (2.6) with n = 0 to provide the solution over the first step since (2.2) only requires f_{-1} to be accurate to first order.

To our knowledge method (b) has been used exclusively until Coleman and Mohamed (1979) in all applications of de Vogelaere's method to solve the radial Schrödinger equation; perhaps a reason for the continual neglect of method (a) as a means of starting is that once y_{-1} has been calculated in method (b) it is possible to enter the general step immediately. In terms of asymptotic error estimates the methods are equivalent, both giving approximations for $y(x_1)$ with errors of order h^4. We shall see later however that in practice the accuracies of the two estimates may be very different.

The de Vogelaere algorithm has some similarity with Runge-Kutta methods but it involves only 2 function evaluations per step whereas a Runge-Kutta method of the same order requires 3 (see e.g. Scraton, 1964). Unlike Runge-Kutta methods, the de Vogelaere algorithm is not self-starting but this difficulty is easily overcome by use of either method (a) or (b).

§ 2.2 Estimation of the local truncation error for a fixed steplength

The leading terms in the local truncation errors in y_1, y_2, z_2 correspond to the first neglected terms in the finite difference ex-

pansions (2.7) and (2.8) from which the formulae for y_1, y_2, z_2 were derived. Thus we see that the leading term in the truncation error in the step from x_o to x_2 is

$$\frac{2h^5}{45} f_o''' \text{ or } \frac{2h^5}{45} y_o^v \tag{2.13}$$

where we have used $\Delta^3 f_o \triangleq h^3 f_o'''$. Since in general we do not have the fifth derivative of the solution readily available during the calculation we must find some means of estimating (2.13).

It is possible by setting x = h in (2.7) and neglecting the $\Delta^3 f_o$ term to generate a further approximation for $y(x_1)$ which is of fourth order. We shall denote this fourth order approximation to $y(x_1)$ by y_1^* where

$$y_1^* = y_o + hz_o + h^2 [\tfrac{1}{2} f_o + \tfrac{1}{6} \Delta f_o - \tfrac{1}{24} \Delta^2 f_o]$$

that is

$$y_1^* = y_o + hz_o + \frac{h^2}{24} (7f_o + 6f_1 - f_2) \tag{2.14a}$$

$$= y_2 - hz_2 + \frac{h^2}{24} (7f_2 + 6f_1 - f_o) \tag{2.14b}$$

where (2.14b) is obtained from (2.14a) by first substituting for y_o using (2.2) with n = 0 and then substituting for z_o using (2.6) with n = 0.

If method (a) of starting is used the difference $(y_1^* - \hat{\hat{y}}_1)$ is given by

$$y_1^* - \hat{\hat{y}}_1 = \frac{h^2}{24} (-f_o + 2f_1 - f_2)$$

where we have used equations (2.11) and (2.14a). We make use of the following Taylor expansions for f_1, f_2 about the point x_o:

$$f_1 = f_o + hf_o' + \frac{h^2}{2} f_o'' + \frac{h^3}{6} f_o''' + \text{---}$$

$$f_2 = f_o + 2hf_o' + 2h^2 f_o'' + \frac{4}{3} h^3 f_o''' + \text{---}$$

to obtain

$$y_1^* - \hat{y}_1 = -\frac{h^4}{24} f_o'' - \frac{h^5}{24} f_o''' + O(h^6) . \qquad (2.15)$$

In the next step y_3 is calculated using equation (2.2) with n = 1 and the difference $(y_3^* - y_3)$ is given by

$$y_3^* - y_3 = \frac{h^2}{24}(4f_1 - 9f_2 + 6f_3 - f_4)$$

where y_3^* has been obtained from the analogue of equation (2.14a). We make use of the following Taylor expansions for f_1, f_3, f_4 about x_o:

$$f_1 = f_2 - hf_2' + \frac{h^2}{2}f_2'' - \frac{h^3}{6}f_2''' + \text{---}$$

$$f_3 = f_2 + hf_2' + \frac{h^2}{2}f_2'' + \frac{h^3}{6}f_2''' + \text{---}$$

$$f_4 = f_2 + 2hf_2' + 2h^2 f_2'' + \frac{4}{3}h^3 f_2''' + \text{---}$$

to obtain

$$y_3^* - y_3 = \frac{h^4}{8} f_2'' - \frac{h^5}{24} f_2''' + O(h^6). \qquad (2.16)$$

Further use of Taylor expansions for f_2'', f_2''' about x_o permits an estimate of (2.13) by consideration of a suitable combination of the differences $(y_1^* - \hat{y}_1)$ and $(y_3^* - y_3)$. We find

$$(y_3^* - y_3) + 3(y_1^* - \hat{y}_1) \simeq \frac{h^5}{12} f_o'''$$

which is $\frac{45}{24}$ times the allowed error. Thus the leading term in the truncation error is given by

$$\frac{2}{45} h^5 f_o''' \simeq \frac{8}{15}[(y_3^* - y_3) + 3(y_1^* - \hat{y}_1)] . \qquad (2.17)$$

If instead, method (b) of starting is used, we consider the difference $(y_1^* - y_1)$ in the first step, where y_1 is given by equation (2.2) with n=0. We have

$$y_1^* - y_1 = \frac{h^4}{8} f_o'' - \frac{h^5}{24} f_o''' + O(h^6) \qquad (2.18)$$

which follows directly from equation (2.16). We can now obtain the following linear combination of (2.16) and (2.18):

$$(y_3^* - y_3) - (y_1^* - y_1) \simeq \frac{h^5}{4} f_o'''$$

which is $\frac{45}{8}$+ times the allowed error. Thus the leading term in the truncation error is given by

$$\frac{2}{45} h^5 f_o''' \simeq \frac{8}{45} [(y_3^* - y_3) - (y_1^* - y_1)]. \qquad (2.19)$$

An estimation of the truncation error for a general step from x_{2n} to x_{2n+2} is given by

$$\frac{2}{45} h^5 f_{2n}''' \simeq \frac{8}{45} [(y_{2n+3}^* - y_{2n+3}) - (y_{2n+1}^* - y_{2n+1})] \qquad (2.20)$$

regardless of which method of starting is used.

§ 2.3 A bound for the local truncation error

The error analysis described here is based on three functionals which are related to the truncation errors in the formulae (2.2), (2.4), (2.6). For an arbitrary function y(x), having p+1 continuous derivatives, we define the functional

$$\mathcal{L}_1[y(x),h] = y(x+h) - y(x) - hy'(x) - \frac{h^2}{6}[4y''(x) - y''(x-h)]. \qquad (2.21)$$

Taylor's theorem can be expressed in the form

$$y(x+jh) = y(x)+jhy'(x)+\cdots+\frac{(jh)^p}{p!}y^{(p)}(x)+\frac{h^{p+1}}{p!}\int_o^j (j-s)^p y^{(p+1)}(x+sh)ds$$

and we make use of the following Taylor expansions for use in (2.21):

$$y(x+h) = y(x) + hy'(x) + \frac{h^2}{2}y''(x) + \frac{h^3}{6}y'''(x) + \frac{h^4}{6}\int_o^1 (1-s)^3 y^{1v}(x-sh)ds$$

$$y''(x-h) = y''(x) - hy'''(x) + h^2 \int_o^{-1} (-1-s)y^{1v}(x+sh)ds .$$

Thus it can be shown that

+This factor is incorrectly quoted as $\frac{8}{45}$ in de Vogelaere (1955, p.123); consequently the factor appearing in his error estimate is also incorrect and should be read as $\frac{8}{45}$, as in (2.19).

$$\mathcal{L}_1[y(x),\ h] = \frac{h^4}{6}\left[\int_0^1 (1-s)^3 y^{1v}(x+sh)ds + \int_{-1}^{0} (1+s)y^{1v}(x+sh)ds\right]$$

$$= \frac{h^4}{6}\int_{-1}^{1} G_1(s)y^{1v}(x+sh)ds \qquad (2.22)$$

with

$$G_1(s) = \begin{cases} (1+s) & , \quad -1 \leq s \leq 0 \\ (1-s)^3 & , \quad 0 \leq s \leq 1 \end{cases}.$$

For $-1 \leq s \leq 1$, $G_1(s) \geqslant 0$. Since $G_1(s)$ is of constant sign on the interval of integration equation (2.21) may be written using the Mean Value Theorem as

$$\mathcal{L}_1[y(x),\ h] = h^4 c_1 y^{1v}(x+\theta_1 h),\ -1 < \theta_1 < 1 \qquad (2.23)$$

with

$$c_1 = \frac{1}{6}\int_{-1}^{1} G_1(s)\,ds = \frac{1}{8}.$$

We now define the functional

$$\mathcal{L}_2[y(x),\ h] = y(x+2h) - y(x) - 2hy'(x) - \frac{h^2}{3}[4y''(x+h)+2y''(x)] \qquad (2.24)$$

and we make use of the following expansions:

$$y(x+2h) = y(x)+2hy'(x)+2h^2y''(x)+\frac{4}{3}h^3y'''(x)+\frac{2}{3}h^4y^{1v}(x) + \frac{h^5}{24}\int_0^2 (2-s)^4 y^v(x+sh)ds$$

$$y''(x+h) = y''(x) + hy'''(x) + \frac{h^2}{2}y^{1v}(x) + \frac{h^3}{2}\int_0^1 (1-s)^2 y^v(x+sh)ds.$$

Then

$$\mathcal{L}_2[y(x),\ h] = \frac{h^5}{24}\left[\int_0^2 (2-s)^4 y^v(x+sh)ds - 16\int_0^1 (1-s)^2 y^v(x+sh)ds\right]$$

$$= \frac{h^5}{24}\int_0^2 G_2(s)y^v(x+sh)ds \qquad (2.25)$$

where

$$G_2(s) = \begin{cases} (2-s)^4 - 16(1-s)^2 & , \quad 0 \leq s \leq 1 \\ (2-s)^4 & , \quad 1 \leq s \leq 2. \end{cases}$$

For $0 \leq s \leq 1$, $G_2(s) = s^2(s^2 - 8s + 8) \geq 0$ and for $1 \leq s \leq 2$, $G_2(s) = (2-s)^4 \geq 0$.

Thus $G_2(s)$ is of constant sign on the interval of integration and we have

$$\mathcal{L}_2[y(x), h] = h^5 c_2 y^{v}(x + \theta_2 h), \quad 0 < \theta_2 < 2 \tag{2.26}$$

with

$$c_2 = \frac{1}{24}\int_0^2 G_2(s)ds = \frac{2}{45}.$$

The third functional required is

$$\mathcal{L}_3[y(x), h] = y(x+2h) - y(x) - \frac{h}{3}\left[y''(x) + 4y''(x+h) + y''(x+2h)\right] \tag{2.27}$$

and the standard expression for the truncation error in Simpson's rule gives

$$\mathcal{L}_3[y(x), h] = h^5 c_3 y^{v1}(x + \theta_3 h), \quad 0 < \theta_3 < 2 \tag{2.28}$$

where $c_3 = -\frac{1}{90}$.

Let $y(x)$ be the exact solution of the initial -value problem (2.1). To investigate the local truncation error in the step from x_{2n} to x_{2n+2} we suppose that the starting values at x_{2n} are exact, that is

$$y_{2n} = y(x_{2n}), \quad z_{2n} = y'(x_{2n})$$

$$f_{2n} = y''(x_{2n}), \quad f_{2n-1} = y''(x_{2n-1}).$$

In view of the assumed starting values,

$$\mathcal{L}_1[y(x_{2n}), h] = y(x_{2n+1}) - y(x_{2n}) - hy'(x_{2n}) - \frac{h^2}{6}[4y''(x_{2n}) - y''(x_{2n-1})]$$

$$= y(x_{2n+1}) - y_{2n} - hz_{2n} - \frac{h^2}{6}[4f_{2n} - f_{2n-1}]$$

$$= y(x_{2n+1}) - y_{2n+1}. \tag{2.29}$$

Thus the truncation error at x_{2n+1} is

$$y(x_{2n+1}) - y_{2n+1} = \frac{h^4}{8} y^{1v}(x_{2n} + \theta_1 h), \quad -1 < \theta_1 < 1. \tag{2.30}$$

If we assume a bound on the fourth derivative of y for all x in the appropriate interval [a, b], namely

$$\left|y^{iv}(x)\right| \leq M_4 \quad , \qquad x\in[a, b]$$

we obtain

$$\left| y(x_{2n+1}) - y_{2n+1} \right| \leq \frac{h^4}{8} M_4 \ . \tag{2.31}$$

Similarly,

$$\mathcal{L}_2[y(x_{2n}), h] = y(x_{2n+2})-y(x_{2n})-2hy'(x_{2n})-\frac{h^2}{3}[4y''(x_{2n+1}) +2y''(x_{2n})]$$

$$=y(x_{2n+2})-y_{2n}-2hz_{2n}- \frac{h^2}{3}[4y''(x_{2n+1})+2f_{2n}] \ . \tag{2.32}$$

The truncation error at x_{2n+2} is given by

$$y(x_{2n+2})-y_{2n+2} = y(x_{2n+2}) - y_{2n} - 2hz_{2n} - \frac{h^2}{3}(4f_{2n+1} + 2f_{2n})$$

$$= \mathcal{L}_2[y(x_{2n}),h] + \frac{4}{3} h^2[y''(x_{2n+1}) - f_{2n+1}]. \tag{2.33}$$

To obtain a bound on this error we assume a Lipschitz condition

$$\left|f(x, y) - f(x,\eta)\right| \leq K\left|y-\eta\right| \tag{2.34}$$

for all x in [a, b] and all finite y and η. Then if

$$\left|y^{v}(x)\right| \leq M_5, \quad x\in[a, b]$$

we obtain the bound

$$\left|y(x_{2n+2}) - y_{2n+2}\right| \leq \frac{2}{45} h^5 M_5 + \frac{Kh^6}{6} M_4. \tag{2.35}$$

Finally,

$$\mathcal{L}_3[y(x_{2n}),h] = y'(x_{2n+2})-y'(x_{2n}) - \frac{h}{3}[y''(x_{2n}) + 4y''(x_{2n+1}) + y''(x_{2n+2})]$$

$$= y'(x_{2n+2}) - z_{2n} - \frac{h}{3}[f_{2n}+4y''(x_{2n+1})+y''(x_{2n+2})] \tag{2.36}$$

We thus have

$$y'(x_{2n+2})-z_{2n+2} = y'(x_{2n+2})-z_{2n}- \frac{h}{3}[f_{2n} + 4f_{2n+1}+f_{2n+2}]$$

$$= \mathcal{L}_3[y(x_{2n}),h] + \frac{4}{3} h[y''(x_{2n+1})-f_{2n+1}]+\frac{h}{3}[y''(x_{2n+2})-f_{2n+2}] \tag{2.37}$$

Now if we assume a bound on the sixth derivative of y for all x in

[a, b]:

$$\left| y^{vl}(x) \right| \leq M_6$$

we obtain the bound

$$\left| y'(x_{2n+2}) - z_{2n+2} \right| \leq \frac{h^5}{90} M_6 + \frac{Kh^5}{18}(3+Kh^2)M_4 + \frac{2Kh^6}{135} M_5 \qquad (2.38)$$

§ 2.4 Stability analysis

If y(x) is the exact solution of the initial -value problem (2.1), the global truncation errors in the function and derivative values at the end of the nth de Vogelaere step are

$$e_n^{(1)} = y(x_{2n-1}) - y_{2n-1} \qquad (2.39)$$

$$e_n^{(2)} = y(x_{2n}) - y_{2n} \qquad (2.40)$$

$$e_n^{(3)} = h[y'(x_{2n}) - z_{2n}] \qquad (2.41)$$

where the factor of h in the third definition is introduced to simplify the form of later equations.

Equation (2.29) may be rearranged as

$$y(x_{2n+1}) = y(x_{2n}) + hy'(x_{2n}) + \frac{h^2}{6}[4y''(x_{2n}) - y''(x_{2n-1})] + \mathcal{L}_1[y(x_{2n}), h] \qquad (2.42)$$

and equation (2.2) is given by

$$y_{2n+1} = y_{2n} + hz_{2n} + \frac{h^2}{6}[4f_{2n} - f_{2n-1}].$$

Therefore subtracting we have

$$e_{n+1}^{(1)} = e_n^{(2)} + e_n^{(3)} + \frac{2h^2}{3}[y''(x_{2n}) - f_{2n}] - \frac{h^2}{6}[y''(x_{2n-1}) - f_{2n-1}] + \mathcal{L}_1[y(x_{2n}), h]. \qquad (2.43)$$

Equation (2.32) may be rearranged as

$$y(x_{2n+2}) = y(x_{2n}) + 2hy'(x_{2n}) + \frac{h^2}{3}[4y''(x_{2n+1}) + 2y''(x_{2n})] + \mathcal{L}_2[y(x_{2n}), h] \qquad (2.44)$$

and equation (2.4) is given by

$$y_{2n+2} = y_{2n} + 2hz_{2n} + \frac{h^2}{3}[4f_{2n+1} + 2f_{2n}].$$

Thus

$$e^{(2)}_{n+1} = e^{(2)}_n + 2e^{(3)}_n + \frac{4h^2}{3}[y''(x_{2n+1}) - f_{2n+1}] + \frac{2h^2}{3}[y''(x_{2n}) - f_{2n}] + \mathcal{L}_2[y(x_{2n}), h]. \tag{2.45}$$

Equation (2.36) may be rearranged as

$$y'(x_{2n+2}) = y'(x_{2n}) + \frac{h}{3}[y''(x_{2n}) + 4y''(x_{2n+1}) + y''(x_{2n+2})] + \mathcal{L}_3[y(x_{2n}), h] \tag{2.46}$$

and equation (2.6) is given by

$$z_{2n+2} = z_{2n} + \frac{h}{3}[f_{2n} + 4f_{2n+1} + f_{2n+2}].$$

Hence

$$e^{(3)}_{n+1} = e^{(3)}_n + \frac{h^2}{3}[y''(x_{2n}) - f_{2n}] + \frac{4}{3}h^2[y''(x_{2n+1}) - f_{2n+1}] + \frac{h^2}{3}[y''(x_{2n+2}) - f_{2n+2}] + h\mathcal{L}_3[y(x_{2n}), h]. \tag{2.47}$$

In accordance with normal practice (e.g. Lambert, 1973) the stability of the method is discussed with reference to the equation

$$y'' = \lambda^2 y.$$

In this case the equations for the cumulative errors simplify to

$$A\,\underline{e}_{n+1} = B\,\underline{e}_n + \underline{\phi}_{n+1} \tag{2.48}$$

where $\underline{e}_n = [e^{(1)}_n, e^{(2)}_n, e^{(3)}_n]^T$, $\underline{\phi}_{n+1} = [\mathcal{L}_1, \mathcal{L}_2, h\mathcal{L}_3]^T$ and the matrices A and B have the form

$$A = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{4}{3}\bar{h} & 1 & 0 \\ -\frac{4}{3}\bar{h} & -\frac{\bar{h}}{3} & 1 \end{bmatrix}, \quad B = \begin{bmatrix} -\frac{\bar{h}}{6} & 1 + \frac{2}{3}\bar{h} & 1 \\ 0 & 1 + \frac{2}{3}\bar{h} & 2 \\ 0 & \frac{\bar{h}}{3} & 1 \end{bmatrix}$$

with

$$\bar{h} = \lambda^2 h^2.$$

Since A is non-singular equation (2.48) may be written as

$$\underline{e}_{n+1} = C\underline{e}_n + A^{-1}\underline{\phi}_{n+1}$$

where

$$C = A^{-1}B = \begin{bmatrix} -\frac{\bar{h}}{6} & 1+\frac{2}{3}\bar{h} & 1 \\ -\frac{2\bar{h}}{9}^2 & 1+2\bar{h}+\frac{8}{9}\bar{h}^2 & 2(1+\frac{2}{3}\bar{h}) \\ -\frac{2\bar{h}}{9}^2(1+\frac{\bar{h}}{3}) & \frac{\bar{h}}{3}(6+\frac{14}{3}\bar{h}+\frac{8}{9}\bar{h}^2) & 1+2\bar{h}+\frac{4}{9}\bar{h}^2 \end{bmatrix}$$

The characteristic polynomial of the matrix C is

$$\rho(r, \bar{h}) = 6r^3 - (12 + 23\bar{h} + 8\bar{h}^2)r^2 + (6-2\bar{h} - 4\bar{h}^2)r + \bar{h} .$$

The "Schur criterion" described in Lambert (1973, p. 78) can be used to show that $\rho(r,\bar{h})$ is a Schur polynomial, in other words that all its zeros lie inside the unit circle if and only if $\bar{h} \in (-2, 0)$. Thus the interval of absolute stability of de Vogelaere's method is [-2,0]. The moduli of the zeros of $\rho(r, \bar{h})$ are plotted in Figure 1 for a range of values of $\bar{h}$. The three zeros, though distinct, have the same modulus when $\bar{h} = -1.732$ (to three decimal places).

§ 2.5 The cumulative error

For the purpose of this section we redefine the quantities $e_n^{(i)}$, i = 1, 2, 3 in equations (2.39) - (2.41) to be

$$e_n^{(1)} = h[y(x_{2n-1}) - y_{2n-1}] \qquad (2.49)$$

$$e_n^{(2)} = y(x_{2n}) - y_{2n} \qquad (2.50)$$

$$e_n^{(3)} = y'(x_{2n}) - z_{2n} \qquad (2.51)$$

Notice that the leading terms in $e_n^{(1)}$, $e_n^{(2)}$, $e_n^{(3)}$ are proportional to h^5. A comparison of equations (2.2), (2.4), (2.6) with (2.42), (2.44), (2.46) respectively leads to the following set of equations:

$$e_{n+1}^{(1)} = he_n^{(2)} + h^2 e_n^{(3)} + \frac{2}{3}h^2[y''(x_{2n}) - f_{2n}] - \frac{h}{6}^3[y''(x_{2n-1}) - f_{2n-1}] + \ell_1[x_{2n}, h] \qquad (2.52)$$

$$e_{n+1}^{(2)} = e_n^{(2)} + 2he_n^{(3)} + \frac{4}{3}h^2[y''(x_{2n+1}) - f_{2n+1}] + \frac{2}{3}h^2[y''(x_{2n}) - f_{2n}] + \ell_2[y(x_{2n}), h] \qquad (2.53)$$

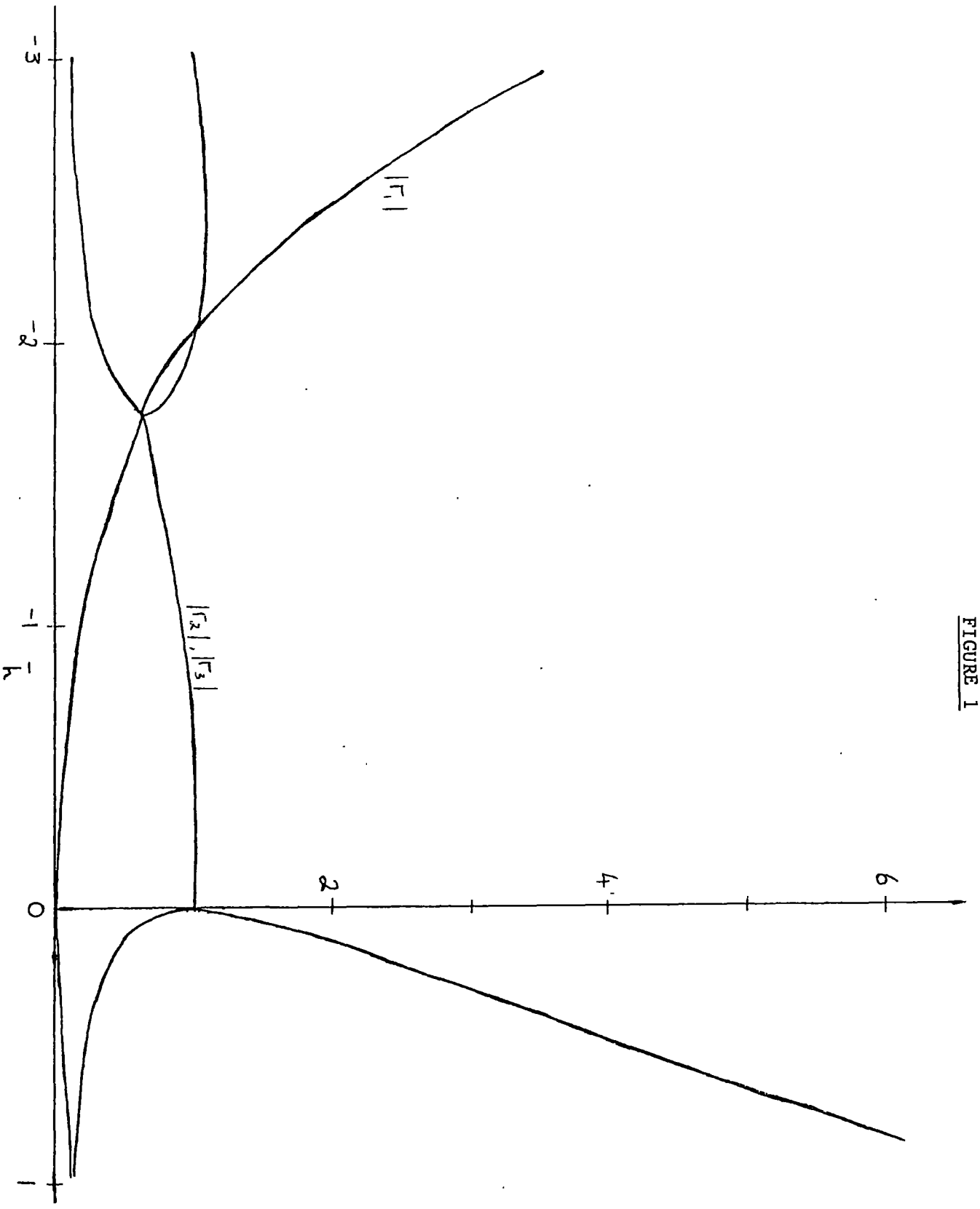

FIGURE 1

$$e_{n+1}^{(3)} = e_n^{(3)} + \frac{h}{3}[y''(x_{2n})-f_{2n}] + \frac{4h}{3}[y''(x_{2n+1})-f_{2n+1}]+\frac{h}{3}[y''(x_{2n+2})-f_{2n+2}]$$

$$+\mathcal{L}_3[y(x_{2n}),\ h] \qquad (2.54)$$

where

$$\mathcal{L}_1 = h^4 c_1\, y^{1v}(x+\theta_1 h) \quad , \quad -1<\theta_1<1$$

$$\mathcal{L}_2 = h^5 c_2\, y^{v}(x+\theta_2 h) \quad , \quad 0<\theta_2<2$$

$$\mathcal{L}_3 = h^5 c_3\, y^{vl}(x+\theta_3 h) \quad , \quad 0<\theta_3<2$$

If as in the previous section we assume a Lipschitz condition

$$\left| f(x, y) - f(x,\eta) \right| \leq K \left| y - \eta \right|$$

for all x in the appropriate interval [a, b] and all finite y and η and if

$$\left|y^{1v}(x)\right| \leq M_4, \quad \left|y^{v}(x)\right| \leq M_5, \left|y^{vl}(x)\right| \leq M_6, \ x \in [a, b]$$

equations (2.52) - (2.54) give the bounds

$$\left|e_{n+1}^{(1)}\right| \leq \frac{h^2}{6} K \left|e_n^{(1)}\right| + h(1+\frac{2}{3}h^2K)\left|e_n^{(2)}\right| + h^2\left|e_n^{(3)}\right| + h^5 c_1 M_4$$

$$\left|e_{n+1}^{(2)}\right| \leq \frac{4}{3} hK \left|e_{n+1}^{(1)}\right| + (1+\frac{2}{3}h^2K)\left|e_n^{(2)}\right| + 2h\left|e_n^{(3)}\right| + h^5 c_2 M_5$$

$$\left|e_{n+1}^{(3)}\right| \leq \frac{4}{3} K \left|e_{n+1}^{(1)}\right| + \frac{h}{3} K\left(\left|e_n^{(2)}\right| + \left|e_{n+1}^{(2)}\right|\right) + \left|e_n^{(3)}\right| + h^5 c_3 M_6 \ .$$

Then for $\left|e_n^{(1)}\right| \leq \alpha_n$, $\left|e_n^{(2)}\right| \leq \beta_n$, $\left|e_n^{(3)}\right| \leq \gamma_n$ the following set of equations is satisfied

$$\alpha_{n+1} = \frac{h^2}{6} K\alpha_n + h(1+\frac{2}{3}h^2K)\beta_n + h^2\gamma_n + \delta_1 \qquad (2.55)$$

$$\beta_{n+1} = \frac{4}{3}hK\alpha_{n+1} + (1+\frac{2}{3}h^2K)\beta_n + 2h\gamma_n + \delta_2 \qquad (2.56)$$

$$\gamma_{n+1} = \frac{4}{3}K\alpha_{n+1} + h\frac{K}{3}(\beta_n + \beta_{n+1}) + \gamma_n + \delta_3 \qquad (2.57)$$

where $\delta_1 = h^5 c_1 M_4$, $\delta_2 = h^5 c_2 M_5$, $\delta_3 = h^5 c_3 M_6$. Equations (2.55) - (2.57) may be written in matrix form as

$$A\,\underline{u}_{n+1} = B\underline{u}_n + \underline{\delta} \qquad (2.58)$$

where $\underline{u}_n = [\alpha_n, \beta_n, \gamma_n]^T$, $\underline{\xi} = [\xi_1, \xi_2, \xi_3]^T$ and the matrices A and B have the form

$$A = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{4}{3}hK & 1 & 0 \\ -\frac{4}{3}K & -\frac{h}{3}K & 1 \end{bmatrix} \qquad B = \begin{bmatrix} \frac{h^2}{6}K & h(1+\frac{2}{3}h^2K) & h^2 \\ 0 & (1+\frac{2}{3}h^2K) & 2h \\ 0 & \frac{h}{3}K & 1 \end{bmatrix}$$

A is clearly non-singular. Now $\underline{u}_{n+1}$ is obtained by multiplying equation (2.58) on the left by A^{-1}:

$$\underline{u}_{n+1} = \bar{C}\underline{u}_n + A^{-1}\underline{\xi} \tag{2.59}$$

where

$$\bar{C} = A^{-1}B = \begin{bmatrix} \frac{h^2}{6}K & h(1+\frac{2}{3}h^2K) & h^2 \\ \frac{2}{9}h^3K^2 & 1+2h^2K+\frac{8}{9}h^4K^2 & 2h+\frac{4}{3}h^3K \\ \frac{2}{9}h^2K^2(1+h^2\frac{K}{3}) & \frac{h}{3}K(6+\frac{14}{3}h^2K+\frac{8}{9}h^4K^2) & 1+2h^2K+\frac{4}{9}h^4K^2 \end{bmatrix}$$

and

$$A^{-1}\underline{\xi} = \begin{bmatrix} h^4c_1M_4 \\ \frac{4}{3}h^6Kc_1M_4 + h^5c_2M_5 \\ \frac{4}{3}K(1+h^2\frac{K}{3})h^5c_1M_4 + \frac{h^6}{3}Kc_2M_5 + h^5c_3M_6 \end{bmatrix}.$$

We look for a bound on the second component of $\underline{u}_n$ and a measure of this component can be obtained by considering the vector norm of $\underline{u}_n$, given by $\|\underline{u}_n\|$. We consider the subordinate matrix norm defined by

$$\|\bar{C}\| = \sup_{\|\underline{u}_n\| \neq 0} \frac{\|\bar{C}\underline{u}_n\|}{\|\underline{u}_n\|}$$

and we consider in particular the infinity norm where $\|\bar{C}\|_\infty$ denotes the maximum absolute row sum of $\bar{C}$. Then from (2.59) it follows that

$$\left\|\underline{u}_{n+1}\right\|_\infty \leq \left\|\bar{C}\right\|_\infty \left\|\underline{u}_n\right\|_\infty + \left\|A^{-1}\underline{\zeta}\right\|_\infty . \qquad (2.60)$$

The row sums of $\bar{C}$ are given by

$$h\left[1 + h\left(1 + \frac{K}{6}\right) + \frac{2}{3}h^2K\right] \qquad (2.61)$$

$$1 + 2h + 2h^2K + \frac{2}{9}h^3K(K+6) + \frac{8}{9}h^4K^2 \qquad (2.62)$$

$$1+2hK + 2h^2K\left(1+\frac{K}{9}\right) + \frac{14}{9}h^3K^2 + \frac{2}{27}h^4K^2(K+6) + \frac{8}{27}h^5K^3 \qquad (2.63)$$

and both (2.62) and (2.63) are bounded by $1 + 2ah$ where a is a constant which can be specified for $h \leq$ some h_o. Thus we have

$$\left\|\bar{C}\right\|_\infty \leq 1 + 2ah$$

and

$$\left\|\underline{u}_{n+1}\right\|_\infty \leq (1 + 2ah)\left\|\underline{u}_n\right\|_\infty + h^5M,$$

where M is a constant suitably defined for all $h \leq h_o$. It follows that

$$\left\|\underline{u}_n\right\|_\infty \leq (1 + 2ah)^n \left\|\underline{u}_o\right\|_\infty + \frac{[(1+2ah)^n - 1]}{2ah} \cdot h^5M$$

$$\leq e^{2anh}\left\|\underline{u}_o\right\|_\infty + \frac{[e^{2anh} - 1]}{2a} \cdot h^4M$$

$$= e^{a(x - x_o)}\left\|\underline{u}_o\right\|_\infty + \left\{\frac{e^{a(x-x_o)} - 1}{2a}\right\} h^4M$$

where we have used $2nh = x - x_o$. Thus for any $h_o > 0$, constants a and M exist such that, for all $h \leq h_o$,

$$\left|y(x_{2n}) - y_{2n}\right| \leq \mathcal{E}e^{a(x_{2n} - x_o)} + h^4\frac{M}{2a}\left\{e^{a(x_{2n} - x_o)} - 1\right\} \qquad (2.64)$$

where

$$\mathcal{E} = \max\left\{h\left|y(x_{-1}) - y_{-1}\right|, \left|y(xo) - y_o\right|, \left|z(x_o) - z_o\right|\right\}.$$

In particular if the initial conditions $y_o = y(x_o)$ and $z_o = z(x_o)$ are satisfied exactly then

$$\mathcal{E} = h\left|y(x_{-1}) - y_{-1}\right|$$

which is of order h^4 if equation (2.12) is used to compute y_{-1}.

We have shown that the global truncation error is of order h^4 where h is the steplength; the dependence on the steplength h can also be seen in an alternative approach described by Kopal (1955, p.219) and the explicit form of the global error for the initial-value problem

$$y'' = -y, \quad y(0) = 0, \quad y'(0) = 1$$

is calculated in Coleman and Mohamed (1978).

§ 2.6 De Vogelaere's method with variable stepsize

An appealing feature of de Vogelaere's method is the ease with which an arbitrary change of steplength can be introduced without the need for interpolation or additional function evaluations. If a steplength h_1 is used as far as x_{2n}, the quantity f_{2n-1} refers to the mesh point $x_{2n-1} = x_{2n}-h_1$. If we now change the steplength to $h_2=ch_1$, f_{2n-1} must be replaced in equation (2.2) by $\bar{f}_{2n-1}$, an approximation for f at $x_{2n}-h_2$. This can be achieved by defining

$$\bar{f}_{2n-1} = f_{2n} + c\,(f_{2n-1} - f_{2n}) \qquad (2.65)$$

which has a local truncation error of order h_2^2.

We have established that the leading term in the truncation error in the step from x_{2n} to x_{2n+2} is

$$\frac{2}{45}h^5 f_{2n}'''.$$

and we showed in § 2.2 that an estimate of this quantity is given by the expression

$$\frac{2}{45}h^5 f_{2n}''' \simeq \frac{8}{45}[(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})].$$

The truncation error per unit step, which is a more appropriate basis for decisions about the steplength, is approximated by

$$\frac{4}{45h}[(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})] \qquad (2.66)$$

When the steplength is changed the expression given by (2.66) has to be modified. There are three separate cases:

(i) Immediately following a step change

Let h_1 be the steplength used in the step from x_{2n-2} to x_{2n} and in the preceeding step, and let $h_2 = ch_1$ be the new steplength for the step from x_{2n} to x_{2n+2}. We shall consider the forms of the differences $(y^*_{2n-1} - y_{2n-1})$ and $(y^*_{2n+1} - y_{2n+1})$ where y^*_{2n+1} is a fourth order approximation to $y(x_{2n+1})$; the form of y^*_{2n+1} has been discussed in Section 2 and is given by

$$y^*_{2n+1} = y_{2n} + hz_{2n} + \frac{h^2}{24}(7f_{2n} + 6f_{2n+1} - f_{2n+2}) \qquad (2.67a)$$

$$= y_{2n+2} - hz_{2n+2} + \frac{h^2}{24}(7f_{2n+2} + 6f_{2n+1} - f_{2n}) \qquad (2.67b)$$

for the general step from x_{2n} to x_{2n+2} with constant steplength h. In the step from x_{2n-2} to x_{2n} we have y^*_{2n-1} given by

$$y^*_{2n-1} = y_{2n-2} + h_1 z_{2n-2} + \frac{h_1^2}{24}(7f_{2n-2} + 6f_{2n-1} - f_{2n}) \qquad (2.68)$$

where we have replaced n by n-1 in equation (2.67a).

y_{2n-1} is given by equation (2.2) with n replaced by n-1 and h replaced by h_1:

$$y_{2n-1} = y_{2n-2} + h_1 z_{2n-2} + \frac{h_1^2}{6}(4f_{2n-2} - f_{2n-3}) \qquad (2.69)$$

and we have

$$y^*_{2n-1} - y_{2n-1} = \frac{h_1^2}{24}(4f_{2n-3} - 9f_{2n-2} + 6f_{2n-1} - f_{2n})$$

$$= \frac{h_1^4}{8} f''_{2n-1} - \frac{h_1^5}{6} f'''_{2n-1} + O(h_1^6) \,. \qquad (2.70)$$

In the next step from x_{2n} to $x_{2n+2} = x_{2n} + 2h_2$, y^*_{2n+1} is given by

$$y^*_{2n+1} = y_{2n} + h_2 z_{2n} + \frac{h_2^2}{24}(7f_{2n} + 6f_{2n+1} - f_{2n+2}) \,. \qquad (2.71)$$

Equation (2.65) combines with (2.2) to give

$$y_{2n+1} = y_{2n} + h_2 z_{2n} + \frac{h_2^2}{6}[(3+c) f_{2n} - c f_{2n-1}] \qquad (2.72)$$

and for c = 1 equation (2.72) correctly reduces to (2.2).

We have

$$y^*_{2n+1} - y_{2n+1} = \frac{h_2^4}{24}(1 + \frac{2}{c}) f''_{2n+1} - \frac{h_2^5}{36}(2 + \frac{3}{c} + \frac{1}{c^2}) f'''_{2n+1} + O(h_2^6). \qquad (2.73)$$

An estimate of the truncation error following the stepchange may now be constructed using a suitable linear combination of (2.70) and (2.73) to eliminate the terms in h_1^4 and h_2^4. Thus we consider

$$(y^*_{2n+1} - y_{2n+1}) - \frac{1}{3} c^4 (1 + \frac{2}{c})(y^*_{2n-1} - y_{2n-1}) = \frac{h_2^5}{72}(-1 + \frac{7}{c} + \frac{12}{c^2}) f'''_{2n} + O(h_2^6)$$

and an estimate of the truncation error per unit step is given by

$$\frac{h_2^4}{45} f'''_{2n} = \frac{1}{2h_2} \cdot \frac{2}{45} h_2^5 f'''_{2n} \approx \frac{8c[(y^*_{2n+1} - y_{2n+1}) - \alpha(y^*_{2n-1} - y_{2n-1})]}{5h_1(12+7c-c^2)} \qquad (2.74)$$

where

$$\alpha = \frac{1}{3} c^3 (2 + c).$$

(ii) The second step after a stepchange

If the steplength $h_2 = ch_1$, introduced for the step from x_{2n} to x_{2n+2} is retained in the next step, the equation

$$y^*_{2n+3} - y_{2n+3} = \frac{h_2^4}{8} f''_{2n+3} - \frac{h_2^5}{6} f'''_{2n+3} + O(h_2^6) \qquad (2.75)$$

follows directly from (2.70). An estimate of the truncation error in this case is given by considering the following combination of (2.73) and (2.75):

$$\frac{1}{3}(1+\frac{2}{c})(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1}) = \frac{h_2^5}{36}(3 + \frac{5}{c} + \frac{1}{c^2}) f'''_{2n+2} + O(h_2^6)$$

and an estimate of the truncation error per unit step is given by

$$\frac{4c\,[\beta(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})]}{5h_1\,(1 + 5c + 3c^2)} \qquad (2.76)$$

where

$$\beta = \frac{1}{3}\left(1 + \frac{2}{c}\right).$$

(iii) Two stepchanges in succession

If $x_{2n} - x_{2n-2} = 2h_1$, $x_{2n+2} - x_{2n} = 2h_2$, and $x_{2n+4} - x_{2n+2} = 2h_3$, with $h_2 = ch_1$ and $h_3 = c_1 h_2$, we have by analogy with equation (2.73)

$$y^*_{2n+3} - y_{2n+3} = \frac{h_3^4}{24}\left(1+\frac{2}{c_1}\right) f''_{2n+3} - \frac{h_3^5}{36}\left(2 + \frac{3}{c_1} + \frac{1}{c_1^2}\right) f'''_{2n+3} + O(h_3^6) . \qquad (2.77)$$

The appropriate combination of (2.73) and (2.77) is

$$\frac{1}{3}\left(1 + \frac{2}{c}\right)(y^*_{2n+3} - y_{2n+3}) - \frac{1}{3}c_1^4\left(1 + \frac{2}{c_1}\right)(y^*_{2n+1} - y_{2n+1})$$

$$= \frac{h_3^5}{216}\left[\left(-1 + \frac{7}{c_1} + \frac{12}{c_1^2}\right) + \frac{1}{c}\left(-2 + \frac{12}{c_1} + \frac{20}{c_1^2}\right) + \frac{2}{c^2}\left(\frac{1}{c_1} + \frac{2}{c_1^2}\right)\right] + O(h_3^6)$$

and we have the following estimate of the truncation error per unit step

$$\frac{24c^2c_1^2\,[\beta(y^*_{2n+3} - y_{2n+3}) - \alpha_1(y^*_{2n+1} - y_{2n+1})]}{5h_3\,[c^2(12 + 7c_1 - c_1^2) + c(20 + 12c_1 - 2c_1^2) + 4 + 2c_1]} \qquad (2.78)$$

where

$$\beta = \frac{1}{3}\left(1 + \frac{2}{c}\right), \qquad \alpha_1 = \frac{1}{3}c_1^3(2 + c_1) .$$

We have presented estimates of the truncation error per unit step which allow for every conceivable sequence of stepsizes in computations based on de Vogelaere's method. Two steps of de Vogelaere's method are required before an estimate of the truncation error can be obtained since the error estimate involves a linear combination of the differences $(y^*_{2n+1} - y_{2n+1})$ for two consecutive steps. With these estimates readily available it is now possible

to consider an implementation of de Vogelaere's method which will permit an efficient automatic error control which automatically selects steps as large as possible while satisfying some error criterion specified by the user.

CHAPTER 3

RADISH - an implementation of de Vogelaere's method with automatic error control

Introduction

We investigate the performance of an automatic implementation in Fortran IV of de Vogelaere's method to solve

$$y''(x) = f(x, y(x)). \tag{3.1}$$

We are interested primarily in the application of de Vogelaere's method to solve the single channel radial Schrödinger equation where $f(x, y)$ is given by

$$f(x, y) = \left[\frac{\ell(\ell+1)}{x^2} - k^2 + V(x)\right] y(x)$$

and a description of the program RADISH to solve the above problem will be given. The test program listed in Appendix 1 solves the single channel radial Schrödinger equation for scattering of an electron by the static potential of atomic hydrogen. The test program thus solves

$$y''(x) = \left[\frac{\ell(\ell+1)}{x^2} - k^2 - 2\left(1 + \frac{1}{x}\right) e^{-2x}\right] y(x) \tag{3.2}$$

for a specified energy $E = k^2$ and angular momentum ℓ, and calculates the scattering phase shift.

As we are concerned mainly with the control of truncation errors in the numerical solution of the differential equation, we use only the most straight forward methods for providing suitable starting values for the integration stage and for the extraction of a phase shift; the program is written however so that the user may readily substitute his own initialisation and phase-calculation routines.

The subroutine DEVOG which is the main routine in RADISH solves (3.1) using de Vogelaere's method where f is supplied as a function

sub-program. The choice of steplength within DEVOG is made according to the strategy described in § 3.2.

§ 3.1 Programming de Vogelaere's algorithm

We have seen in § 2.1 that for a fixed steplength the general step of de Vogelaere's method leading from x_{2n} to $x_{2n+2} = x_{2n} + 2h$ is:

Given y_{2n}, z_{2n}, f_{2n} and f_{2n-1},

$$y_{2n+1} = y_{2n} + hz_{2n} + \frac{h^2}{6}(4f_{2n} - f_{2n-1}) \tag{3.3}$$

$$f_{2n+1} = f(x_{2n+1}, y_{2n+1}) \tag{3.4}$$

$$y_{2n+2} = y_{2n} + 2hz_{2n} + \frac{h^2}{3}(4f_{2n+1} + 2f_{2n}) \tag{3.5}$$

$$f_{2n+2} = f(x_{2n+2}, y_{2n+2}) \tag{3.6}$$

$$z_{2n+2} = z_{2n} + \frac{h}{3}(f_{2n} + 4f_{2n+1} + f_{2n+2}) \quad . \tag{3.7}$$

Following de Vogelaere (1955) we introduce the quantities

$$Z_{2n} = hz_{2n},\ F_{2n} = \frac{h^2}{3} f_{2n},\ F_{2n+1} = \frac{4}{3} h^2 f_{2n+1}$$

and equations (3.3) - (3.7) may now be written as

$$y_{2n+1} = y_{2n} + Z_{2n} + 2F_{2n} - \frac{1}{8} F_{2n-1} \tag{3.8}$$

$$F_{2n+1} = \frac{4}{3} h^2 f(x_{2n+1}, y_{2n+1}) \tag{3.9}$$

$$y_{2n+2} = y_{2n} + 2Z_{2n} + F_{2n+1} + 2F_{2n} \tag{3.10}$$

$$F_{2n+2} = \frac{1}{3} h^2 f(x_{2n+2}, y_{2n+2}) \tag{3.11}$$

$$Z_{2n+2} = Z_{2n} + F_{2n} + F_{2n+1} + F_{2n+2} \quad . \tag{3.12}$$

If at the nth step of the calculation we let

$$Y = y_{2n},\ Z = Z_{2n},\ \text{FEVEN} = F_{2n},\ \text{FODD} = F_{2n-1}$$

equations (3.8) - (3.12) can be formulated using the FORTRAN statements as:

```
Z       =  Z  +  FEVEN                          (3.13)
Y       =  Y  +  Z                              (3.14)
FODD    =  Y + FEVEN - 0.125 * FODD             (3.15)
FODD    =  4.0* H2* F(X+H, FODD) /3.0           (3.16)
Z       =  Z  +  FODD                           (3.17)
Y       =  Y  +  Z                              (3.18)
FEVEN   =  H2 * F (X + HH, Y)/3.0               (3.19)
Z       =  Z  +  FEVEN                          (3.20)
```

where X represents the mesh point x_{2n} and H, HH, H2 represent h, 2h, h^2 respectively. F is a function sub-program which represents f and which calculates the second derivative of the solution. Thus we emerge from (3.13) - (3.20) with

$$Y = y_{2n+2},\quad Z = Z_{2n+2},\quad FEVEN = F_{2n+2},\quad FODD = F_{2n+1}.$$

The general step may be implemented essentially by using (3.13) - (3.20) and the cost per step in terms of arithmetic operations is 9 additions and 6 multiplications; 2 function evaluations are required per step.

The steplength h at each step will be chosen so as to make the estimated error per unit step less than some prescribed tolerance; thus we estimate

$$TRERR = \frac{4}{45h}\left[(y^{*}_{2n+1} - y_{2n+1}) - (y^{*}_{2n-1} - y_{2n-1}) \right] \qquad (3.21)$$

from which a decision can be taken as to whether to keep the same steplength h for the next step or to increase or decrease h according as the estimate is respectively less than or greater than the prescribed tolerance. The precise strategy adopted is described in § 3.2.

If a steplength h_1 is used as far as x_{2n}, and the steplength is then changed to $h_2 = ch_1$, a sufficiently accurate estimate for f at $x_{2n} - h_2$ is given by

$$\bar{f}_{2n-1} = f_{2n} + c(f_{2n-1} - f_{2n}) \quad . \tag{3.22}$$

At the end of the final step which uses the steplength h_1 we have

$$Y = y_{2n}, \quad Z = h_1\bar{z}_{2n}, \quad FEVEN = \frac{1}{3} h_1^2 f_{2n}, \quad FODD = \frac{4}{3} h_1^2 f_{2n-1} .$$

Then if we define

$$F = \frac{4}{3} h_2^2 \bar{f}_{2n-1}$$

we can use the following implementation of (3.22):

```
F = 4.0*C2* FEVEN + C**3*(FODD - 4.0*FEVEN)
```

where C, C2 represent c, c^2. Thus if the steplength is changed from h_1 to h_2 it is necessary to update the values of Z, FEVEN and FODD for the next step. This is effected by the following FORTRAN statements:

```
Z      = C * Z                                   (3.23)
FEVEN  = C2 * FEVEN                              (3.24)
FODD   = C2 * FODD                               (3.25)
FODD   = 4.0 * (1.0 - C)* FEVEN + C*FODD  .      (3.26)
```

An estimate of the truncation error per unit step is now given by

$$TRERR = \frac{8C\,[(y^*_{2n+1} - y_{2n+1}) - \alpha(y^*_{2n-1} - y_{2n-1})]}{5h_1(12 + 7C - C^2)} \tag{3.27}$$

with

$$\alpha = \frac{1}{3} C^3 (2 + C) ,$$

The next step from x_{2n+2} to x_{2n+4} will either

(i) retain the steplength h_2, or

(ii) require a further change in steplength to $h_3 = C_1 h_2$ in which case further updating of the values Z, FEVEN, FODD will be required.

The corresponding estimates of TRERR are then given by

(i) $$\frac{4C[\beta(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})]}{5h_1(1 + 5C + 3C^2)} \qquad (3.28)$$

(ii) $$\frac{24\,C^2\,C_1^{\,2}\,[\beta(y^*_{2n+3} - y_{2n+3}) - \alpha_1(y^*_{2n+1} - y_{2n+1})]}{5h_3\,\gamma} \qquad (3.29)$$

where

$$\beta = \frac{1}{3}\left(1 + \frac{2}{C}\right), \quad \alpha_1 = \frac{1}{3}C_1^{\,3}(2 + C_1)$$

and

$$\gamma = C^2(12 + 7C_1 - C_1^{\,2}) + C(20 + 12C_1 - 2C_1^{\,2}) + 4 + 2C_1.$$

§ 3.2 Steplength strategy

The user is required to specify as input for the subroutine DEVOG, a parameter EPS which is the largest allowed local error per unit step. Since the magnitude of the solution may not be known in advance the program RADISH calculates at each step

$$\text{TOL} = \text{EPS} \times \text{Max}\left\{1, |y_{2n}|\right\}, \qquad (3.30)$$

so as to provide an absolute or relative error criterion according as the absolute value of the calculated solution is less or greater than unity. The calculated y_{2n} is deemed satisfactory if

$$|\text{TRERR}| \leq \text{TOL}. \qquad (3.31)$$

If this inequality is not satisfied the step is rejected and it is necessary to restart from y_{2n-2} with a decreased steplength. Since the local error per unit step varies approximately as h^4 we argue that a steplength αh would give a truncation error of approximately α^4 TRERR, and we can therefore estimate a suitable α. When the inequality (3.31) is almost satisfied a difficulty could arise, in that the value found for α is almost one, and a large number of very small adjustments of steplength may be required before a satisfactory value is found. To avoid this we arrange that when a decrease is

necessary a non-trivial decrease is carried out, by choosing

$$\alpha = \left(\frac{0.5\ \text{TOL}}{\text{TRERR}}\right)^{\frac{1}{4}}.$$

Successive decreases in steplength must be allowed where necessary and are monitored accordingly.

When the calculated y_{2n} has been accepted as sufficiently accurate we consider whether or not a steplength increase is justified. One approach would be to choose a new steplength $Ch>h$, such that

$$C^4 \left| \text{TRERR} \right| < \text{TOL},$$

and this would result in a change of steplength at almost every step. Experience shows that by allowing the steplength to increase at successive steps, particularly at the beginning of the integration, the progress of the solution is hampered, often due to the need to decrease immediately after an increase in steplength. Thus to avoid frequent increases by small amounts, and, much worse, increases followed by immediate decreases, we introduce a number of restrictions. The steplength is allowed to increase only by a specified factor C (typically C = 2) and this increase is carried out only if

$$C^4 \left| \text{TRERR} \right| < 0.5\ \text{TOL}, \qquad (3.32)$$

for three consecutive steps. The numbers in Table 1 given below come from a test run at a stage when the steplength is 0.035880.

TABLE 1

x_{2n}	y_{2n}	TRERR
0.378	0.167	-5.9×10^{-8}
0.450	0.241	-7.5×10^{-9}
0.521	0.331	3.7×10^{-8}
0.593	0.435	7.5×10^{-8}
0.665	0.556	1.0×10^{-7}
0.737	0.692	1.2×10^{-7}

In this calculation EPS = 10^{-6} and C = 2 so the inequality is satisfied if $|\text{TRERR}| < 3 \times 10^{-8}$, as happens on the second line of Table 1. Because we require this inequality to be satisfied at three consecutive steps the current steplength is not changed, a decision which is vindicated by later lines of the table when we recall that by doubling the steplength we would increase the local error per unit step by a factor of about sixteen.

The subroutine DEVOG also requires an initial steplength h which must be supplied by the user. Because of our automatic error control the value of h supplied is not particularly important; if it is unnecessarily small some time may be wasted since we have adopted a rather conservative approach to steplength increases.

Given the initial steplength h the subroutine DEVOG calculates the initial step of the method using method (a) of starting of § 2.1. We prefer to use method (a) of starting to method (b) and indeed advocate its use in all calculations for the solution of the radial Schrödinger equation. We shall devote the next section to a discussion of the merits of method (a) over method (b) and to its subsequent implementation in the subroutine DEVOG.

§ 3.3 The initial step

Methods (a) and (b) of starting described in § 2.1 both provide an approximation for $y(x_1)$ with an error which is proportional to h^4. We shall consider an asymptotic expansion of the error resulting in the computed solution at x_1 for both methods. The normal practice has been to use method (b) of starting in all applications of de Vogelaere's method to solve the radial Schrödinger equation; this method of starting calculates a fourth order estimate for $y(x_1)$ using

$$y_1 = y_o + hz_o + \frac{h^2}{6}(4f_o - f_{-1}) \qquad (3.33)$$

where f_{-1} is given by

$$f_{-1} = f(x_o - h, y_{-1})$$

with

$$y_{-1} = y_o - hz_o + \frac{h^2}{2} f_o \qquad (3.34)$$

and it is apparent that for sufficiently accurate starting values y_o, z_o, f_o any error in y_1 must be a combination of the error in the term $(\frac{h^2}{6} f_{-1})$ in (3.33) resulting from the error in y_{-1} and the local truncation error in (3.33).

The differential equation which we are studying has the form

$$y''(x) = g(x)y$$

and we write $g_k = g(x_k)$, $k = 0, 1, ---$. Then if we define the error in the approximation y_{-1} to $y(x_{-1})$ by

$$\delta y_{-1} \equiv y(x_o - h) - y_{-1} \qquad (3.35)$$

we have

$$\delta y_{-1} = \frac{h^3}{6} f_o' + ---$$

where we have used the Taylor expansion of $y(x_{-1})$ about x_o in (3.35). The difference between y_1 and the true solution $y(x_1)$ is given by

$$\delta y_1 \equiv y(x_1) - y_1$$

where

$$y_1 = y_o + hz_o + \frac{h^2}{6}(4 f_o - f_{-1}) .$$

We use

$$f(x_{-1}, y(x_{-1})) - f(x_{-1}, y_{-1}) = g_{-1} \delta y_{-1}$$

to express y_1 in the form

$$y_1 = y_o + hz_o + \frac{h^2}{2} f_o + \frac{h^3}{6} f_o' - \frac{h^4}{12} f_o'' + \frac{h^2}{6} g_{-1} \delta y_{-1} + O(h^5)$$

$$= y(x_1) - \frac{h^4}{8} f_o'' + \frac{h^2}{6} g_{-1} \delta y_{-1} + O(h^5).$$

Thus

$$\delta y_1 = \frac{h^4}{8} f_o'' - \frac{h^2}{6} g_{-1} \delta y_{-1} + O(h^5) \qquad (3.36)$$

where the second term in (3.36) is of order h^5.

In order to compare the form of the error in the computed solution at x_1 with that obtained using method (a) of starting we investigate the errors in $\tilde{y}_1$ and $\tilde{\tilde{y}}_1$ where

$$\tilde{y}_1 = y_o + hz_o + \frac{h^2}{2} f_o \qquad (3.37)$$

$$\tilde{\tilde{y}}_1 = y_o + hz_o + \frac{h^2}{6} (2f_o + \tilde{f}_1) \qquad (3.38)$$

with

$$\tilde{f}_1 = f(x_1, \tilde{y}_1).$$

The error in $\tilde{y}_1$ is given by

$$\delta\tilde{y}_1 \equiv y(x_1) - \tilde{y}_1 = \frac{h^3}{6} f_o' + \text{---}$$

and we have

$$f(x_1, y(x_1)) - f(x_1, \tilde{y}_1) = g_1 \delta\tilde{y}_1.$$

Thus we can express $\tilde{\tilde{y}}_1$ as

$$\tilde{\tilde{y}}_1 = y_o + hz_o + \frac{h^2}{2} f_o + \frac{h^3}{6} f_o' + \frac{h^4}{12} f_o'' - \frac{h^2}{6} g_1 \delta\tilde{y}_1 + O(h^5)$$

$$= y(x_1) + \frac{h^4}{24} f_o'' - \frac{h^2}{6} g_1 \delta\tilde{y}_1 + O(h^5)$$

and hence

$$\delta\tilde{\tilde{y}}_1 = y(x_1) - \tilde{\tilde{y}}_1$$

$$= \frac{-h^4}{24} f_o''' + \frac{h^2}{6} g_1 \delta\tilde{y}_1 + O(h^5) \qquad (3.39)$$

where the second term in (3.39) is of order h^5.

The leading terms in δy_1 and $\delta\tilde{\tilde{y}}_1$ are very similar in form, but, for the Schrödinger equation with $\ell \neq 0$, $|g(x)|$ is generally a rapidly decreasing function of x near the origin. Consequently g_{-1} may be substantially greater than g_1. The following numerical example demonstrates this effect. With $\ell = 1$, $E = 0.01$, $x_o = 0.04$, and the equation (3.2) used in the test program, Taylor expansion about

the origin gives

$$y_o = 1.568756 \times 10^{-3}, \quad z_o = 7.767540 \times 10^{-2}.$$

The precise method of choosing the initial values x_o, y_o, z_o, f_o will be explained in §3.5.1 which describes the main routine of RADISH. Then with h = 0.025 method (b) of starting gives

$$y_1 = 4.099156 \times 10^{-3},$$

by method (a) we get

$$\hat{\hat{y}}_1 = 4.093352 \times 10^{-3}$$

and the exact solution is

$$y(x_1) = 4.093015 \times 10^{-3}.$$

In this case $|\delta y_1|$ exceeds $|\delta\hat{\hat{y}}_1|$ by a factor of about 18, which is close to the factor by which g_{-1} ($\simeq 8758$) exceeds g_1 ($\simeq 445$).

Our conclusion is that when solving the Schrödinger equation method (a) is superior to method (b). The less accurate method is used for example by Chandra (1973) in a computer program to solve the differential equations arising in a close-coupling formulation of quantum-mechanical scattering problems. The argument in favour of method (a) can be applied to a more general function f(x,y) simply by replacing g_k by $\left(\frac{\partial f}{\partial y}\right)_{x = x_k}$.

Equation (3.5) shows that the error $\delta\hat{\hat{y}}_1$ contributes the term $(\frac{4}{3} h^2 g_1 \delta\hat{\hat{y}}_1)$ to the error in y_2. Although this term is of order h^6 it may, when g_1 is large, exceed the $O(h^5)$ truncation error in the formula for y_2, and therefore it is imperative that its magnitude be controlled. Having obtained $\hat{\hat{y}}_1$ from equation (3.38) and y_2 from equation (3.5) we may use the formula

$$y_1^* = y_2 - hz_2 + \frac{h^2}{24} (7f_2 + 6f_1 - f_o)$$

which gives a new estimate for $y(x_1)$. The error in this approximation

is of order h^5 and consequently

$$\delta\hat{\hat{y}}_1 \approx y_1^* - \hat{\hat{y}}_1 .$$

This allows us to monitor the contribution from $\delta\hat{\hat{y}}_1$ to the error in y_2, and to adjust the steplength accordingly. With the completion of the first step the cumulative (actual) and truncation errors equate and the truncation error at the end of this step is given by 2h times the truncation error per unit step. Thus a reasonable requirement of the method is that the absolute value of the actual error in y_2 be less than or equal to 2h TOL where

$$\text{TOL} = \text{EPS} \times \text{Max} \left\{ 1, |y_2| \right\} ,$$

and we ask that

$$\left| \frac{4}{3} h^2 g_1 \delta\hat{\hat{y}}_1 \right| < 2h \text{ TOL}$$

be satisfied before proceeding to the next step. We make this condition more restrictive by requiring

$$\left| h g_1 \delta\hat{\hat{y}}_1 \right| < \text{TOL}$$

that is

$$\left| h \delta f_1 \right| < \text{TOL} \qquad (3.40)$$

where

$$\delta f_1 \equiv f(x_1, y_1^*) - f(x_1, \hat{\hat{y}}_1).$$

Thus given the initial steplength h the subroutine DEVOG calculates $\tilde{y}_1$, $\hat{\hat{y}}_1$, y_2 and y_1^* and the steplength h is considered satisfactory at this stage if the inequality in (3.40) holds. If this inequality is not satisfied a more suitable steplength is found by using the fact that $\delta\hat{\hat{y}}_1$ varies approximately as h^4, and the calculation is repeated. When y_1 is known to the required accuracy we calculate y_3, y_4, y_5 and y_6 and then use (3.21) with n = 2 to estimate the truncation error; if this is small enough the calculation continues, and if not a smaller steplength is chosen and the calculation begins again. It is in fact possible to estimate the truncation error per

unit step a step earlier after the first two steps have been completed using equation (2.17) and thereafter using equation (3.21) for as long as the steplength remains constant; however we choose to estimate this quantity after the completion of the third step thereby using equation (3.21) alone to estimate TRERR, until a change in steplength is necessitated. In this way we dispense with the need for additional programming in initially estimating TRERR. Thus we do not interfere with the natural flow of the coding for a constant steplength and we feel that this approach is satisfactory despite the loss of two function evaluations in the event of the estimated TRERR calculated after the first three steps being too large.

We now give a listing of the FORTRAN statements required for the calculation of the first step from x_o to x_2 using method (a) of starting. We start with

$$XO = x_o, \quad YO = y_o, \quad ZO = hz_o, \quad FO = \frac{h^2}{3} f_o.$$

The method (a) of starting consists of the following statements:

```
Y    =   YO + Z + 1.5 * FEVEN                          (3.41)
FODD =   4.0* H2* F(XO + H, Y)/ 3.0                    (3.42)
Y    =   Y + (FODD - 4.0* FEVEN) /8.0                  (3.43)
FODD =   4.0* H2* F(XO + H, Y) /3.0                    (3.44)
Z1   =   Z + FEVEN                                     (3.45)
Y    =   YO + 2.0* Z1 + FODD                           (3.46)
FEVEN =  H2* F(XO + HH, Y) /3.0                        (3.47)
Z    =   Z1 + FODD + FEVEN                             (3.48)
```

where y_o, z_o, f_o can be obtained from a starting series so that at the beginning of the calculation YO, ZO, FO are readily available.

The cost in terms of arithmetic operations required for the first step for a given H is 11 additions and 12 multiplications; 3 function evaluations must be performed.

§ 3.4 Subroutine DEVOG

The subroutine DEVOG is applicable to any linear or non-linear equation of the form (3.1) and is based on de Vogelaere's method which is described in the preceeding Chapter. Our implementation of the method makes use of the results of § 2.6 which enable us to monitor and control the local error and thus relieve the user of the task of steplength selection. Subroutine DEVOG is written in double precision form to solve (3.1) where f is supplied as a function subprogram. For ease of identification, all statements which refer to the scattering problem, rather than to the numerical solution of the differential equation, are placed between lines of asterisks.

§ 3.4.1 Input

The parameters which must be supplied to DEVOG as input data are:

H : the initial steplength (see § 3.2)

XO, YO, ZO, FO : the starting point, and the values of the solution and its first and second derivatives at that point,

EPS, C : the tolerance parameter and factor by which the steplength is increased when an increase is permitted (see § 3.2).

XF : The calculation terminates somwhere between XF - H and XF + H unless earlier termination has occurred because of convergence of the phase shift to the required accuracy.

§ 3.4.2 The structure of DEVOG

Figure 2 overleaf shows a flow diagram for this subroutine, and the function of the variables used is described in comment cards in

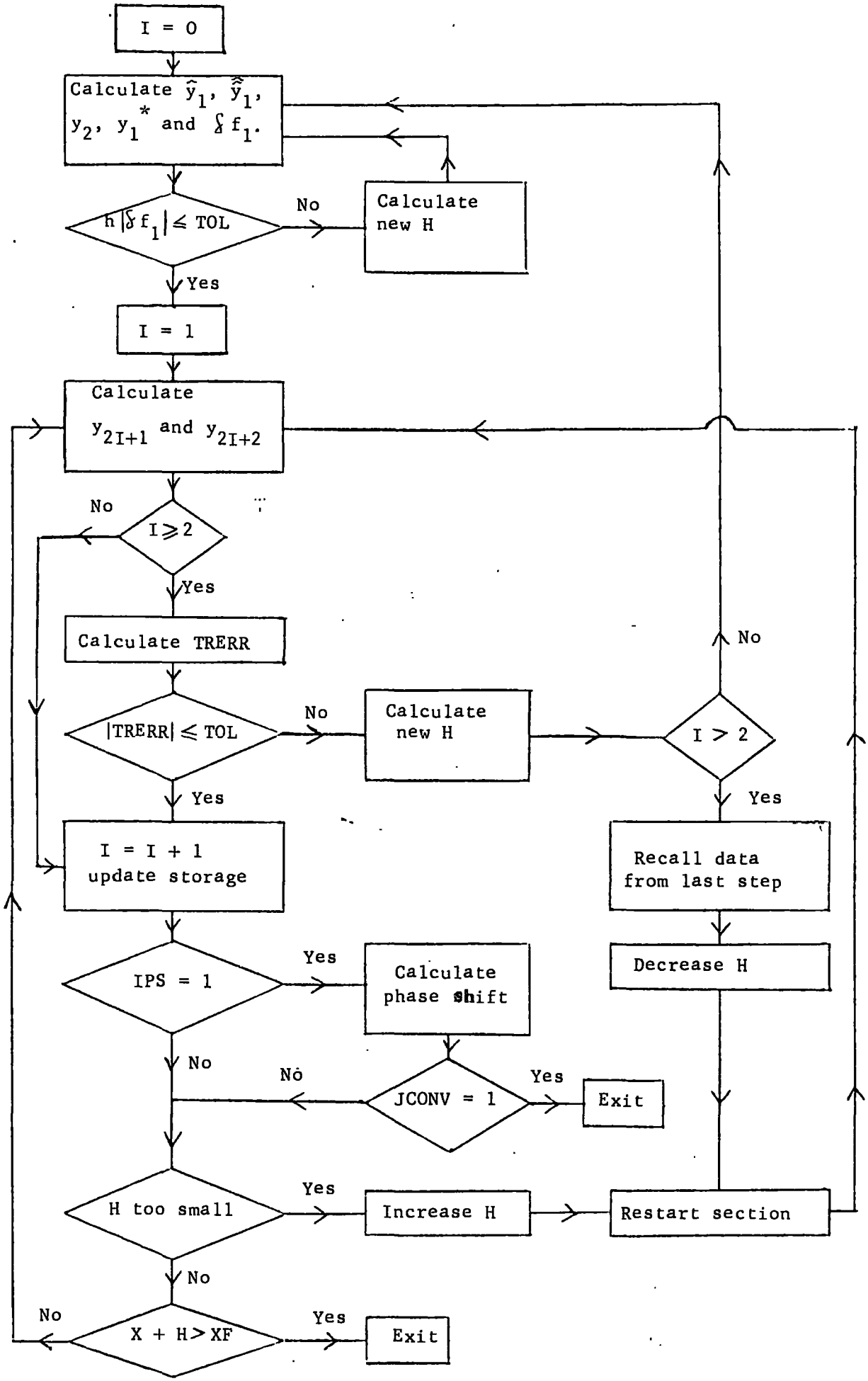
I = 0
Calculate $\hat{y}_1$, $\hat{\hat{y}}_1$, y_2, y_1^* and δf_1.
$h|\delta f_1| \leq TOL$
No
Calculate new H
Yes
I = 1
Calculate y_{2I+1} and y_{2I+2}
No
$I \geq 2$
Yes
Calculate TRERR
$|TRERR| \leq TOL$
No
Calculate new H
$I > 2$
No
Yes
I = I + 1 update storage
Recall data from last step
Decrease H
IPS = 1
Yes
Calculate phase shift
No
No
JCONV = 1
Yes
Exit
H too small
Yes
Increase H
Restart section
No
No
X + H > XF
Yes
Exit

FIGURE 2

the text of the program.

If the initial steplength used were very large it would be possible for the calculation to proceed to XF without providing an estimate of the truncation error. To avoid this difficulty DEVOG takes H to be either the value supplied by the user or (XF - XO)/5, whichever is the smaller. Lines (180 - 208)* of DEVOG calculate $\hat{y}_1$, $\hat{\hat{y}}_1$, y_2 and y_1^* as described in § 3.3 and, if necessary, reduce the steplength repeatedly until the estimated contribution to the error in y_2 from that in $\hat{\hat{y}}_1$ is sufficiently small.

Lines 218 - 294 implement the de Vogelaere algorithm to advance the calculation one step of length 2H, evaluate the truncation error estimate, and decide whether or not to alter the steplength. The natural flow of control corresponds to a situation where the steplength is neither increased nor decreased.

If at line 256 it is decided that the truncation error is too large, control is transferred to lines 313 - 339 for the steplength decrease, and thence to the restart section (lines 361 - 383), followed by a return to the main loop at line 218.

The decision to calculate a phase shift is taken within the function subprogram F. When the calculation has progressed sufficiently far that a phase shift calculation is worthwhile, the parameter IPS is assigned the value 1. When this condition is encountered at line 278 of DEVOG the subroutine PS is called to calculate the phase shift; the current value of x and the calculated phase shift are then printed.

---

* The line numbers quoted are those appearing on the right-hand side of the listing, corresponding to the sequential numbering of the card deck.

If the steplength is to be increased line 287 transfers control to the steplength increase section (lines 344 - 356) which then leads to the restart section and back to the beginning of the de Vogelaere loop.

Apart from that required to bring about changes of steplength, the only branching that occurs in DEVOG arises from the need to treat the truncation error differently in certain cases. At lines 240 and 270 this just involves by-passing some statements, and when a non-trivial branching occurs (at lines 241 and 370) we have imposed an IF...THEN...ELSE structure in the interests of readability.

Exit from the de Vogelaere loop occurs (see line 280) if the subroutine PS indicates, by setting JCONV = 1, that the phase shift has been calculated to the required accuracy. Otherwise termination occurs when it is noted (at line 293 or line 382) that the next step would take the calculation beyond XF. In either case the user is informed of the reason for termination, and information on the number of steps carried out and the number of increases and decreases of steplength is also printed.

§ 3.4.3 Modifications to solve other problems

If the lines between rows of asterisks are deleted, DEVOG will solve the equation

$$y'' = f(x, y)$$

and will terminate somewhere between XF - H and XF + H where H is the current steplength. Values of the solution at each successful step may be printed by a WRITE statement placed after line 256. If the calculation is to end precisely at XF it is necessary only to adjust the final step so that XF is a mesh point; alternatively, to avoid a very large reduction in steplength when XF is very close

to a mesh point, it is possible to look two steps ahead and take action when X + 2H exceeds XF.

The test program is based on the Schrödinger equation in the precise form of (3.2), for positive energy E. However DEVOG is equally applicable to bound state problems (E<0). Also any desired changes of variable may be introduced, as for example in bound state problems where it is convenient to take the logarithm of the radial distance as the integration variable, provided only that the resulting differential equation has the form (3.1).

§ 3.5 Description of program RADISH

Figure 3 below shows the relationship between the six routines of the program RADISH which solves the radial Schrödinger equation

$$y''(x) = \left[\frac{L(L+1)}{x^2} - E + V(x)\right] y(x) \qquad (3.49)$$

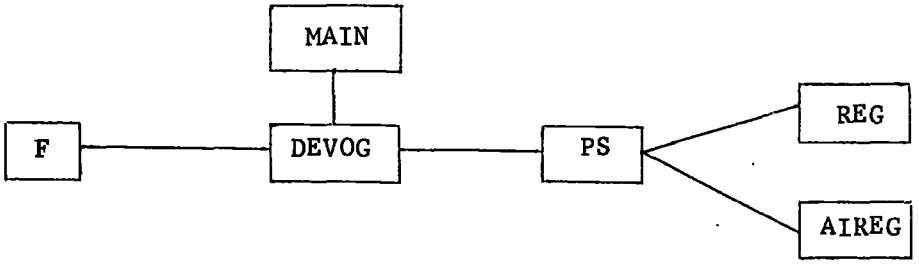

FIGURE 3

Double precision is used for all REAL variables.

§ 3.5.1 The main routine

The program applies to a potential which, in the vicinity of the origin, has the expansion

$$V(x) = \frac{V_1}{x} + V_2 + V_3 x + \text{---} \quad . \qquad (3.50)$$

The first four coefficients are read as data at line 28 and are stored

in the array VCOEFF. The regular solution of the Schrödinger equation (3.49) may be expressed as

$$y(x) = x^{\ell+1} [a_1 + a_2 x + a_3 x^2 + ---] \qquad (3.51)$$

for sufficiently small x, and the coefficients are given by the equations

$$2(\ell + 1)\, a_2 = V_1 a_1$$

$$2(2\ell + 3)\, a_3 = V_1 a_2 + (V_2 - E) a_1$$

$$6(\ell + 2)\, a_4 = V_1 a_3 + (V_2 - E) a_2 + V_3 a_1$$

$$4(2\ell + 5)\, a_5 = V_1 a_4 + (V_2 - E) a_3 + V_3 a_2 + V_4 a_1 .$$

Having read the coefficients V_1, V_2, V_3, V_4, the program then reads the parameters EPS, C and XF. For given values of these parameters an arbitrary number of calculations may be carried out, for different values of the energy E and angular momentum L. Execution terminates when a negative value of L is read.

The program calculates the coefficients a_2, ---, a_5 with $a_1 = 1$. Although the value of H supplied to the subroutine DEVOG is not particularly important, we have chosen to provide a fairly realistic value by using the power series to estimate, a priori, the truncation error in y_2. For $L \leq 4$ equation (3.51) shows that

$$y^{v}(0) = 5!\, a_{5-L}$$

and therefore we take

$$H = \left[\frac{45 \cdot EPS}{|y^{v}(0)|}\right]^{\frac{1}{4}} = \left[\frac{3 \cdot EPS}{8\,|a_{5-L}|}\right]^{\frac{1}{4}} .$$

For $L > 4$ some other approach must be used; it would be quite satisfactory to supply an arbitrary numerical value, e.g. 0.1. Alternatively if the form of the first, second and third derivatives of V(x) with respect to x are known, then by successively differentiating equation (3.49) we can find an exact value for $y_o^{v} = y^{v}(x_o)$ in terms of

x_o, y_o, z_o and f_o. Thus we have

$$y^v_o = \left\{\left[\frac{-24L(L+1)}{XO^5} + V'''(XO)\right] + \left[\frac{L(L+1)}{XO^2} + V(XO)-E\right]\left[\frac{-2L(L+1)}{XO^3}+V'(XO)\right]\right\} YO$$
$$+\left\{3\left[\frac{6L(L+1)}{XO^4} + V''(XO)\right]+\left[\frac{L(L+1)}{XO^2} + V(XO)-E\right]2\right\} ZO$$
$$+ 3\left[\frac{-2L(L+1)}{XO^3} + V'(XO)\right] FO .$$

Notice that for y = sin kx substitution of L = 0 = V in the above expression leads to

$$y_o^v = E^2 Z0$$

corresponding to

$$y_o^v = k^5 \cos kx_o = k^4 z_o .$$

From the series for y(x) an expansion for y'(x) is readily obtained. On the assumption that the magnitude of the first neglected term in each of these expansions gives a measure of the error in truncating that expansion, x_o is chosen so that y_o and hz_o are given sufficiently accurately by the first four terms. We require

$$\left.\begin{array}{l} |a_5| x_o^{L+5} \\ H(L+5)|a_5| x_o^{L+4} \end{array}\right\} \leqslant 0.1\,EPS .$$

Then y_o, z_o and f_o are computed from the truncated expansion and the subroutine DEVOG is called.

The common block EKLL1 is used to transfer parameters to the routines F and PS.

§ 3.5.2 The function F.

This subprogram, which is called by DEVOG, calculates

$$F(x, y) = \left[\frac{L(L+1)}{x^2} + V(x) - E\right] y .$$

In the listing of program RADISH in Appendix 1, V(x) is given by

$$V(x) = -2(1 + \frac{1}{x})\, e^{-2x} \qquad (3.52)$$

corresponding to the static potential between an electron and a hydrogen atom. Here it is also decided when a phase shift should be calculated; when

$$\frac{L(L+1)}{x^2} < E \quad \text{and} \quad \left|V(x)\right| < EPS \cdot \left|E - \frac{L(L+1)}{x^2}\right|$$

the value IPS = 1 is returned to DEVOG.

§ 3.5.3 The subroutine PS.

If the potential is such that

$$\lim_{x\to\infty} x^2 V(x) = 0$$

the general solution of equation (3.49) has the asymptotic form

$$y(x) \sim A\,[S(x) + \tan\delta\; C(x)] \qquad (3.53)$$

where δ is the scattering phase shift, A is a normalisation constant, and

$$S(x) = kx j_L(kx) \underset{x\to\infty}{\sim} \sin(kx - \tfrac{1}{2}L\pi) \qquad (3.54)$$

$$C(x) = -kx y_L(kx) \underset{x\to\infty}{\sim} \cos(kx - \tfrac{1}{2}L\pi)\,, \qquad (3.55)$$

If y(x) is represented sufficiently accurately by the asymptotic form (3.53) when x exceeds some distance R then, for any x_1 and x_2 greater than R, we have

$$\tan\delta = \frac{y(x_2)\,S(x_1) - y(x_1)\,S(x_2)}{y(x_1)\,C(x_2) - y(x_2)\,C(x_1)} \qquad (3.56)$$

The central part of the subroutine PS (lines 424 - 433) calculates the phase shift from this formula, prints the current phase shift estimate and indicates, by setting JCONV = 1, when successive estimates agree to the required accuracy. To guard against spurious convergence x_1 and x_2 in the formula (3.56) are required to differ by at least one atomic unit.

On the first entry to this subroutine the phase shift is arbitrarily set equal to zero and values of the functions S and C are stored for use on the next entry.

The COMMON block EKLL1 transfers parameters from the MAIN routine (see § 3.5.1) and PHASE, which appears only in PS, is used to ensure that information required on the next entry to this subroutine is not lost on returning to the calling program.

§ 3.5.4 The functions REG and AIREG

The functions S(x) and C(x) defined in equations (3.54) and (3.55) are calculated by using the recurrence relation

$$f_{L-1}(z) + f_{L+1}(z) = (2L+1)\frac{f_L(z)}{z} \qquad (3.57)$$

which is satisfied by the spherical Bessel functions $j_L(z)$ and $y_L(z)$. Forward recurrence for $y_L(z)$ is stable for all values of L and z, and this is the method used in AIREG. The accuracy of the calculation is comparable to that of the built-in functions DSIN and DCOS.

For $z > L$, REG also uses forward recurrence, but this process is unstable for $z \leq L$ and it is then necessary to recur backwards. The procedure adopted is that suggested by Corbató and Uretsky (1959), which involves using (3.57) with starting values $f_{v+1} = 0$ and f_v arbitrary, and then normalising by using the known values of $j_0(z)$ and $j_1(z)$. Corbató and Uretsky provide a prescription for choosing v such that, for specified L and z, $j_L(z)$ is calculated with a relative error no greater than some specified δ; we have chosen $\delta = 10^{-7}$. The dimension of the array PF restricts the size of L but REG prints a warning if there is any difficulty from this source; the difficulty can be overcome simply by changing the DIMENSION statement.

The recursive evaluation of j_L and y_L is most useful when we wish to evaluate spherical Bessel functions of different orders

simultaneously for a particular value of the argument. It may be that for the present application, where only a single value of L is relevant to a particular phase shift calculation, there are more efficient methods, but this is peripheral to our main concern.

§ 3.6 Test runs

Program RADISH was tested for a number of different potentials V(x) and phase shifts calculated for each problem for a specified range of values of E and L to enable comparison of the computed phase shifts with the published results. However as the aim of Chapter 7 is to present a comparison of the performance of various test programs each of which incorporates a different numerical method with automatic error control for the solution of (3.49) we shall leave a detailed discussion of the results of de Vogelaere's method to Chapter 7; Chapter 7 investigates the reliability of each of the numerical methods in controlling the global error during the numerical integration stage of the calculation and compares the relative efficiency of the methods in calculating the relevant phase shifts.

Errors in a phase shift calculation can arise from the starting values supplied to the differential equation solver, from the method of solving the differential equation, and from the asymptotic fitting procedure. Thus a comparison of phase shifts calculated by different methods gives information on the performance of the test program as a whole, but not specifically on the behaviour of the differential equation solver. To obtain a better understanding of the performance of DEVOG we solved the following differential equations over specified ranges for different values of the accuracy parameter EPS:

(i) $y'' = -k^2 y$; $y_o = \sin kx_o$, $z_o = k \cos kx_o$, $x_o = 0.01$

(ii) $y'' = \left(\frac{2}{x^2} - k^2\right)y$; $y_o = x_o j_1(kx_o)$, $z_o = \frac{d}{dx}[xj_1(kx)]\Big|_{x_o}$, $x_o = 0.01$.

(i) and (ii) were solved for k = 0.1, 0.2, 0.5, 1.0, 2.0, 5.0 representing a wide range of values of the energy E; for each value of k (i) and (ii) were solved with EPS = 10^{-n}, n = 3, 4, 6, 8. The exact solutions of (i) and (ii) are y(x) = sin kx and y(x) = $xj_1(kx)$ respectively. The parameter EPS is a bound on the local error (absolute or relative as appropriate) per unit step, and to provide a meaningful comparison we have found

$$\max_{[x_o,d]} \left| \frac{y_{exact} - y_{calc}}{\max\{1, |y_{exact}|\}} \right|$$

for d = 5, 10 and 20. The above quantity provides a scaled maximum error over various stages of the integration and this quantity, divided by $(d - x_o)$, is tabulated in Tables 2 and 3 in § 3.7 for each of the problems (i) and (ii) throughout the ranges of k and EPS. We also tabulate the initial value of the steplength H supplied by the main routine for use in DEVOG; this value of H however is not necessarily the initial steplength accepted by DEVOG as sufficiently small. In addition the tables display the number of function evaluations carried out on the interval $[x_o, 20]$. RADISH can be used directly to solve (i) and (ii) by reading in L = 0, 1 for problems (i), (ii) respectively and specifying V(x) = 0 in the function program F; we dispense with the starting series in the main routine and provide instead exact starting values for x_o, y_o, z_o and f_o since the exact solutions of (i) and (ii) are known. The results for problems (i) and (ii) given in the next Section were obtained using RADISH with the initial value of the steplength chosen to be $\left(\frac{45\ EPS}{|y_o^v|}\right)^{\frac{1}{4}}$ where $y_o^v = k^4 z_o$. In the case of problem (ii) the quantity $k^4 z_o$ serves as an estimate for y_o^v but is exact for problem (i).

Our main interest in DEVOG lies in its effectiveness in solving the radial Schrödinger equation of the form (3.49). We can draw up tables of the same form as Tables 2 and 3 where the "exact" result is obtained by writing our second order differential equation as a pair of first order equations, and using a high accuracy variable-step, variable-order subroutine, namely D02AHF, from the N.A.G. Library. Subroutine D02AHF is described fully in Chapter 6. Program RADISH which is used in Chapter 7 to solve a number of scattering problems using de Vogelaere's method must be slightly modified to provide in particular the same starting point and initial steplength for a particular problem to allow a meaningful comparison with the other numerical methods tested for the same problem; details of the modifications required in RADISH will be described in Chapter 7. A discussion of the results of program RADISH which solves (3.2) and calculates the phase shifts for electron hydrogen elastic scattering in the static approximation without any of the modifications required in Chapter 7 is given in Coleman and Mohamed (1979). The starting values x_o, y_o and z_o are determined in the main routine of RADISH.

§ 3.7 Test results

The effectiveness of the steplength strategy in controlling the global error for problems (i) and (ii) of § 3.6 is evident from Tables 2 and 3. If we denote by a the factor by which each value of the scaled maximum error exceeds the corresponding EPS then a_{max}, which represents the maximum such factor a, is 2.60 for problem (i) and 26.3 for problem (ii). As expected, as the energy increases for a particular EPS the initial steplength supplied to DEVOG is decreased and this is evident from the tables. Equally evident is the fact that as the accuracy parameter becomes more stringent for a particular value of the energy the initial steplength supplied to

DEVOG decreases. This decrease in initial steplength is reflected by the number of function evaluations carried out for the range of integration.

The interested reader is referred to Coleman and Mohamed (1979) for a tabulation of the scaled maximum errors in solving equation (3.2) for the same range of values of k and EPS used in problems (i) and (ii). In this case $a_{max} = 3.31$. The test run which accompanies the test program RADISH in Appendix 1, is for EPS = 10^{-6} and PSIG = 10^{-4} for the following values of E and L:

L = 0, 1, 2

E = 0.16, 0.25, 0.5, 0.8, 1.0, 4.0, 9.0, 16.0, 25.0 .

The low energies ($E \leq 0.8$) were chosen to allow comparison with results quoted by Burke and Smith (1962), and some results at higher energies may be compared with those of McDougall (1932), Chandrasekhar and Breen (1946) and of Moiseiwitsch and O'Brien (1970). In many cases we get exact agreement with the published values, and where there are small discrepancies our results are confirmed by calculations which we have carried out with more stringent accuracy requirements. Table 32 of Appendix 1 compares phase shifts computed using

(a) EPS = 10^{-6}, PSIG = 10^{-4}

(b) EPS = 10^{-8}, PSIG = 10^{-5}

(c) EPS = 10^{-4}, PSIG = 10^{-3}

with the published result for the range of values of E and L. The agreement between the phase shifts obtained using the accuracy parameters given by (a) and (b) indicate that we can have confidence in our results for the phase shift.

TABLE 2

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	5.16(-5)	5.16(-5)	7.43(-5)	3.998	8
	0.2	7.36(-4)	7.36(-4)	2.07(-3)	3.444	8
	0.5	6.96(-4)	1.29(-3)	2.23(-3)	1.095	20
	1.0	1.52(-3)	1.49(-3)	1.17(-3)	0.461	54
	2.0	1.51(-3)	1.25(-3)	1.15(-3)	0.194	126
	5.0	1.16(-3)	1.12(-3)	1.13(-3)	0.062	390
10^{-4}	0.1	5.16(-5)	5.16(-5)	7.43(-5)	3.998	8
	0.2	9.13(-5)	1.22(-4)	1.98(-4)	1.936	12
	0.5	1.57(-4)	1.45(-4)	1.48(-4)	0.616	40
	1.0	1.52(-4)	1.44(-4)	1.24(-4)	0.259	94
	2.0	1.43(-4)	1.24(-4)	1.14(-4)	0.109	220
	5.0	1.24(-4)	1.14(-4)	1.14(-4)	0.035	686
10^{-6}	0.1	8.52(-7)	5.84(-7)	1.25(-6)	1.456	16
	0.2	5.25(-7)	1.17(-6)	1.86(-6)	0.612	34
	0.5	1.99(-6)	1.46(-6)	1.42(-6)	0.195	120
	1.0	1.46(-6)	1.42(-6)	1.24(-6)	0.082	284
	2.0	1.42(-6)	1.24(-6)	1.15(-6)	0.034	686
	5.0	1.24(-6)	1.14(-6)	1.13(-6)	0.011	2164
10^{-8}	0.1	8.53(-9)	4.43(-9)	1.44(-8)	0.461	46
	0.2	4.37(-9)	1.44(-8)	1.77(-8)	0.194	112
	0.5	2.08(-8)	1.42(-8)	1.42(-8)	0.062	370
	1.0	1.43(-8)	2.13(-8)	1.93(-8)	0.026	818
	2.0	2.60(-8)	2.28(-8)	2.12(-8)	0.011	1896
	5.0	2.11(-8)	1.96(-8)	1.96(-8)	0.003	5978

TABLE 3

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	9.41(-4)	9.41(-4)	3.40(-4)	3.998	8
	0.2	7.91(-4)	5.18(-4)	2.54(-4)	3.998	26
	0.5	1.40(-3)	5.82(-4)	5.80(-4)	3.834	42
	1.0	1.04(-3)	7.39(-4)	6.38(-4)	1.612	80
	2.0	1.29(-3)	1.13(-3)	1.20(-3)	0.678	116
	5.0	7.52(-4)	7.75(-4)	8.04(-4)	0.216	298
10^{-4}	0.1	1.70(-4)	1.07(-4)	4.82(-5)	3.998	22
	0.2	1.99(-4)	9.50(-5)	4.34(-5)	3.998	34
	0.5	1.33(-4)	1.36(-4)	9.70(-5)	2.156	70
	1.0	1.50(-4)	8.28(-5)	4.84(-5)	0.906	140
	2.0	6.35(-5)	3.99(-5)	3.38(-5)	0.381	268
	5.0	3.42(-5)	3.36(-5)	3.39(-5)	0.121	632
10^{-6}	0.1	5.11(-6)	2.99(-6)	1.39(-6)	3.998	50
	0.2	5.19(-6)	2.56(-6)	1.25(-6)	2.143	74
	0.5	3.03(-6)	1.48(-6)	1.17(-6)	0.682	164
	1.0	1.33(-6)	1.02(-6)	9.58(-7)	0.287	306
	2.0	7.98(-7)	1.16(-6)	1.34(-6)	0.121	562
	5.0	1.38(-6)	1.52(-6)	1.59(-6)	0.038	1332
10^{-8}	0.1	3.30(-8)	1.85(-8)	1.27(-8)	1.612	92
	0.2	1.29(-7)	7.20(-8)	3.64(-8)	0.678	170
	0.5	2.63(-7)	1.33(-7)	6.64(-8)	0.216	414
	1.0	2.33(-7)	1.16(-7)	5.83(-8)	0.091	812
	2.0	1.01(-7)	5.04(-8)	2.73(-8)	0.038	1714
	5.0	4.27(-8)	2.41(-8)	1.76(-8)	0.012	4274

CHAPTER 4

Numerov's Method

§4.1 Derivation of method

Numerov's method for solving

$$y'' = f(x, y) \tag{4.1}$$

is well known and is expressed as:

Given y_{n-1}, y_n

$$y_{n+1} = 2y_n - y_{n-1} + \frac{h^2}{12}(f_{n+1} + 10f_n + f_{n-1}) \tag{4.2}$$

where y_n is an approximation to the solution at the mesh point x_n. We shall give a derivation of (4.2) in this Section. We can use Taylor's theorem to express the solution of (4.1) at $x \pm h$ as

$$y(x \pm h) = y(x) \pm hy'(x) + \int_x^{x\pm h} (x \pm h - t)\, y''(t)dt.$$

Thus we have

$$\Delta^2 y(x-h) = y(x+h) - 2y(x) + y(x-h) = \int_x^{x+h} (x+h-t)f(t)dt + \int_x^{x-h} (x-h-s)f(s)ds$$

and substitution of $+zh$, $-zh$ for $x + h - t$, $x - h - s$ respectively leads to the equation

$$\Delta^2 y(x-h) = h^2 \int_0^1 z\{f(x+h-zh) + f(x-h+zh)\}dz.$$

We introduce the operator E:

$$E^p f(x) = f(x+ph)$$

and we see that the backward difference operator can be expressed as

$$\nabla = I - E^{-1}.$$

Thus we have

$$\Delta^2 y(x-h) = h^2 \int_0^1 z\left\{(I-\nabla)^z + (I-\nabla)^{2-z}\right\} f(x+h)dz.$$

We use the operator expansions

$$(I - \nabla)^z = I - z\nabla + z\frac{(z-1)}{2}\nabla^2 - z\frac{(z-1)(z-2)}{6}\nabla^3 + z\frac{(z-1)(z-2)(z-3)}{24}\nabla^4 + \text{----}$$

$$(I-\nabla)^{2-z} = I+(z-2)\nabla+\frac{(z-2)(z-1)}{2}\nabla^2+\frac{(z-2)(z-1)z}{6}\nabla^3+\frac{(z-2)(z-1)z(z+1)}{24}\nabla^4+---$$

to express $\Delta^2 y(x-h)$ as

$$\Delta^2 y(x-h) = h^2\left[1 - \nabla + \frac{1}{12}\nabla^2 - \frac{1}{240}\nabla^4 + ---\right] f(x+h).$$

If we truncate the above expansion at the term involving the fourth difference of f and substitute x_n for x we have

$$y_{n+1} - 2y_n + y_{n-1} \simeq \frac{h^2}{12}(f_{n+1}+10f_n+f_{n-1}) - \frac{h^2}{240}\nabla^4 f_{n+1}.$$

Thus equation (4.2) provides a two step method to obtain the solution of (4.1) with the leading term of the local truncation error per step given by

$$\frac{-h^2}{240}\nabla^4 f_{n+1} \simeq \frac{-h^6}{240} y_{n+1}^{vl} . \qquad (4.3)$$

We shall use Numerov's method to solve the single channel radial Schrödinger equation in the form

$$y''(x) = \left[\frac{\ell(\ell+1)}{x} - E + V(x)\right] y(x) \qquad (4.4)$$

and we shall be required to provide two starting values, namely y_o and y_1. Details of the implementation will be described in § 4.4.

§ 4.2 Characteristics of Numerov's method

Numerov's method has been studied at a fixed steplength by a number of authors (see § 1.4) to solve the radial Schrödinger equation. The characteristics of the method such as the local truncation error, the region of absolute stability and the cumulative error are well known and we shall discuss them briefly in this Section.

§4.2.1 The local truncation error

We saw in the preceeding Section that the leading term in the local truncation error in the step from x_n to x_{n+1} is given by (4.3). A bound for the local truncation error may be obtained from a more explicit form for the local truncation error by considering the functional defined by

$$\mathcal{L}[y(x);h] = y(x+h)-2y(x)+y(x-h)-\frac{h^2}{12}\left\{y''(x+h)+10y''(x)+y''(x-h)\right\}$$

for an arbitrary function $y(x)$ having six continuous derivatives. After using the appropriate Taylor expansions we arrive at the following form for $\mathcal{L}[y(x);h]$:

$$\mathcal{L}[y(x);h] = \frac{h^6}{720}\int_{-1}^{1} G(s)y^{vl}(x+sh)ds \qquad (4.5)$$

with

$$G(s) = \begin{cases} -6(-1-s)^5+10(-1-s)^3, & -1 \leq s \leq 0 \\ 6(1-s)^5-10(1-s)^3, & 0 \leq s \leq 1. \end{cases}$$

$G(s)$ is negative over the range $[-1, 1]$; thus we may write (4.5) as

$$\mathcal{L}[y(x);h] = h^6\, cy^{vl}(x+\theta h), \quad -1<\theta<1 \qquad (4.6)$$

with

$$c = \frac{1}{720}\int_{-1}^{1} G(s)ds = -\frac{1}{240}\ .$$

If $y(x)$ is the exact solution of (4.1) and we assume that the starting values at x_{n-1} and x_n are exact, that is

$$y_{n-1} = y(x_{n-1}), \quad y_n = y(x_n)$$

$$f_{n-1} = y''(x_{n-1}), \quad f_n = y''(x_n)$$

then

$$\mathcal{L}[y(x);h] = y(x_{n+1})-2y(x_n)+y(x_{n-1})-\frac{h^2}{12}\left\{y''(x_{n+1})+10y''(x_n)+y''(x_{n-1})\right\}$$

$$= y(x_{n+1})-2y_n+y_{n-1}-\frac{h^2}{12}\left\{y''(x_{n+1})+10f_n+f_{n-1}\right\}. \qquad (4.7)$$

The truncation error at x_{n+1} is given by

$$y(x_{n+1}) - y_{n+1} = y(x_{n+1}) - 2y_n + y_{n-1} - \frac{h^2}{12}(f_{n+1} + 10f_n + f_{n-1})$$

$$= \mathcal{L}[y(x);h] + \frac{h^2}{12}[y''(x_{n+1}) - f_{n+1}]. \qquad (4.8)$$

To obtain a bound on this error we assume a Lipschitz condition given by equation (2.34):

$$|f(x, y) - f(x, \eta)| \leq K|y - \eta|$$

for all x in the appropriate interval [a, b] and all finite y and η. We also assume a bound on the sixth derivative of y for all x in [a, b]:

$$|y^{vi}(x)| \leq M_6, \quad x \in [a, b].$$

Thus we obtain the bound

$$|y(x_{n+1}) - y_{n+1}| \leq \frac{h^6}{240} M_6 + \frac{h^2 K}{12} |y(x_{n+1}) - y_{n+1}|$$

whereby

$$|y(x_{n+1}) - y_{n+1}| \leq \frac{\frac{h^6}{240} M_6}{\left|1 - \frac{h^2 K}{12}\right|}$$

Then for all $h \leq h_o < \sqrt{\frac{12}{K}}$ we have

$$|y(x_{n+1}) - y_{n+1}| \leq h^6 N \qquad (4.9)$$

where

$$N = \frac{M_6}{240\left(1 - \frac{h_o^2 K}{12}\right)}$$

§ 4.2.2 Absolute stability of the method

We shall establish the interval of absolute stability for Numerov's method using the boundary locus method which is discussed in Lambert (1973, p. 82).

The locus of the boundary of the region of absolute stability is given by

$$\bar{h}(\theta) = \frac{\rho(\exp(i\theta))}{\sigma(\exp(i\theta))} \tag{4.10}$$

where ρ, σ are the first and second characteristic polynomials of the stability polynomial

$$\pi(r, \bar{h}) = \rho(r) - \bar{h}\sigma(r) = 0.$$

ρ and σ are given by

$$\rho(r) = r^2 - 2r + 1, \quad \sigma(r) = \frac{1}{12}(r^2 + 10r + 1).$$

Thus

$$\bar{h}(\theta) = \frac{12(-18 + 16\cos\theta + 2\cos 2\theta)}{(102 + 40\cos\theta + 2\cos 2\theta)}$$

so that the boundary of the region is an interval of the real axis; the end points of the interval are given by the maximum and minimum values of the function $\bar{h}(\theta)$. Thus the interval of absolute stability is found to be $[-6, 0]$.

§4.2.3 The cumulative error

We can find a bound on the cumulative error using Henrici's approach (1962, p. 312). Numerov's method expressed in general multistep form is

$$\sum_{i=0}^{2} \alpha_i y_{n+i} = h^2 \sum_{i=0}^{2} \beta_i f_{n+i} \tag{4.11}$$

where

$$\alpha_0 = 1 = \alpha_2, \quad \alpha_1 = -2$$

$$\beta_0 = \frac{1}{12} = \beta_2, \quad \beta_1 = \frac{5}{6}.$$

The exact solution satisfies

$$y(x_n+2h)-2y(x_n+h)+y(x_n) = \frac{h^2}{12}\left\{y''(x_n+2h)+10y''(x_n+h)+y''(x_n)\right\}+\mathcal{L}[y(x_{n+1});h].$$

Then if we assume

$$\left|y(x_\mu) - y_\mu\right| \leq h\delta, \quad \mu = 0, 1$$

with

$$h^2 < K^{-1}\left|\alpha_2\beta_2^{-1}\right|$$

we obtain the following bound

$$\left|y(x_n)-y_n\right| = \left|e_n\right| \leq \Gamma^*\left[8(x_n-a^*)\delta + \frac{(x_n-a^*)^2}{2}Nh^4\right]\exp[(x_n-a^*)^2\,\Gamma^*KB] \tag{4.12}$$

for $a \leq x_n \leq b$, where K is the Lipschitz constant defined in (2.34), a^* and Γ^* are given by

$$a^* = a - \frac{h\gamma}{\Gamma}$$

$$\Gamma^* = \frac{\Gamma}{1-h^2K\left|\alpha_2^{-1}\beta_2\right|}$$

where Γ and γ are constants (see Henrici) and B is given by

$$B = \left|\beta_1\right| + \left|\beta_2\right| + \left|\beta_3\right| = 1.$$

Thus the cumulative error is of order h^4.

The dependence on the steplength h can also be seen in an alternative approach described by Kopal (1955, p. 219). We consider the application of Numerov's method to the solution of

$$y'' = -y,\ y(0) = 0,\ y'(0) = 1 \tag{4.13}$$

for which the exact solution is $y(x) = \sin x$. Substitution of (4.13) in (4.11) leads to

$$y_{n+2} = 2\left\{\frac{12 - 5h^2}{12 + h^2}\right\} y_{n+1} - y_n$$

and the solution of this difference equation is given by

$$y_n = Ar_1^n + Br_2^n$$

where A and B are arbitrary constants and r_1, r_2 are roots of the characteristic equation

$$r^2 - 2\left(\frac{12 - 5h^2}{12 + h^2}\right)r + 1 = 0.$$

Notice that the product of the roots is unity, thus we write r_1, r_2 as

$$r_{1,2} = \cos\phi \pm i \sin\phi$$

and hence

$$\phi = \cos^{-1}\left(\frac{12 - 5h^2}{12+h^2}\right) = h + \frac{h^5}{480} + \text{---} \quad .$$

If now we set

$$A = \frac{C_1 - iC_2}{2} \quad , \qquad B = \frac{C_1 + iC_2}{2}$$

it follows that

$$y_n = C_1 \cos n\phi + C_2 \sin n\phi$$

$$\hat{=} \; C_1 \cos nh + C_2 \sin nh - C_1 \frac{nh^5}{480} \sin nh + C_2 \frac{nh^5}{480} \cos nh \quad .$$

For exact starting values, that is $y(0) = 0$ and $y(h) = \sin h$, we obtain

$$C_1 = 0, \quad C_2 = \frac{\sin h}{\sin \phi} = 1 - \frac{h^4}{480} + \text{---} \quad .$$

Thus the cumulative error is given by

$$u_N = \sin x - y_n = (1 - C_2)\sin x - C_2 \frac{h^4}{480} x \cos x$$

$$= \frac{h^4}{480}(\sin x - x \cos x) + \text{---} \quad . \qquad (4.14)$$

The corresponding result obtained for de Vogelaere's method is given in Coleman and Mohamed (1978) as

$$u_D = \frac{h^4}{180}(9 x \cos x - 5 \sin x) + \text{---} \quad . \qquad (4.15)$$

In particular when $x = \frac{\pi}{2}$ the leading term in the estimates given by (4.14) and (4.15) is $\frac{h^4}{180}$ and $-\frac{h^4}{36}$ respectively which agrees closely with numerical results as can be seen from Tables 4 and 5 in § 4.3.

§ 4.3 Numerov's method with variable stepsize

A feature of linear multistep methods for equation (4.1) which have a local truncation error of order h^{p+2} is that the global error is of order h^p, (see Henrici, 1962) and we have shown that for Numerov's method the leading term of the local truncation error is proportional to h^6 whereas the global or cumulative error is proportional to h^4. This contrasts with the hybrid method of de Vogelaere where the local

truncation error is of order h^5 and the global error is of order h^4. We chose to control the local error per unit step in an attempt to control the global error in calculations based on de Vogelaere's method. In view of the difference of a factor of h^2 in the local truncation and global errors for Numerov's method we suspect that a control of the local error per unit step per unit step will give a better control of the global error than will a control of the local error per unit step. Of course the extent to which these error controls are effective in controlling the global error will depend on the stability of the differential equation.

An estimate of the local error per unit step is given by

$$-\frac{h^5}{240} y_{n+1}^{vl} \qquad (4.16)$$

and an estimate of the local error per unit step per unit step is given by

$$-\frac{h^4}{240} y_{n+1}^{vl} \,. \qquad (4.17)$$

We have used the methods of de Vogelaere and Numerov with a constant steplength over the range $\left[0, \frac{\pi}{2}\right]$ to solve the problem $y'' = -y$, $y(0) = 0$, $y'(0) = 1$, $x \in \left[0, \frac{\pi}{2}\right]$ which has the exact solution $y = \sin x$. At each step of the calculation in de Vogelaere the local truncation error per unit step is estimated and in the calculation based on Numerov's method estimates of the local truncation error per unit step and per unit step per unit step are provided. Table 4 shows the maximum error and the maximum truncation error per unit step multiplied by the length of the range of integration which is recorded over the range $\left[0, \frac{\pi}{2}\right]$ using de Vogelaere's method for a range of values of steplength h. Also tabulated is the appropriate value of the leading term of u_D in (4.15). The corresponding results for Numerov's method are recorded in Table 5 along with an

TABLE 4

h	$u_D \triangleq -\frac{h^4}{36}$	Max. error in $\left[0, \frac{\pi}{2}\right]$	Max. trunc. error per unit step x $\frac{\pi}{2}$
$\frac{\pi}{20}$	-1.7(-5)	-9.1(-6)	1.9(-5)
$\frac{\pi}{40}$	-1.1(-6)	-8.1(-7)	1.3(-6)
$\frac{\pi}{60}$	-2.1(-7)	-1.8(-7)	2.7(-7)
$\frac{\pi}{80}$	-6.6(-8)	-5.8(-8)	8.3(-8)
$\frac{\pi}{100}$	-2.7(-8)	-2.4(-8)	3.5(-8)
$\frac{\pi}{120}$	-1.3(-8)	-1.2(-8)	1.6(-8)
$\frac{\pi}{140}$	-7.0(-9)	-6.6(-9)	8.8(-9)
$\frac{\pi}{160}$	-4.1(-9)	-3.9(-9)	5.2(-9)
$\frac{\pi}{180}$	-2.6(-9)	-2.4(-9)	3.3(-9)
$\frac{\pi}{200}$	-1.7(-9)	-1.6(-9)	2.2(-9)

TABLE 5

h	$u_N \triangleq \frac{h^4}{480}$	Max. error in $\left[0, \frac{\pi}{2}\right]$	Max. trunc. error per unit step x $\frac{\pi}{2}$	Max. trunc. error per unit step per unit step x $\frac{\pi}{2}$
$\frac{\pi}{20}$	1.3(-6)	1.3(-6)	5.5(-7)	3.5(-6)
$\frac{\pi}{40}$	7.9(-8)	7.9(-8)	1.9(-8)	2.4(-7)
$\frac{\pi}{60}$	1.6(-8)	1.6(-8)	2.5(-9)	4.9(-8)
$\frac{\pi}{80}$	5.0(-9)	5.0(-9)	6.1(-10)	1.5(-8)
$\frac{\pi}{100}$	2.0(-9)	2.0(-9)	2.0(-10)	6.3(-9)
$\frac{\pi}{120}$	9.8(-10)	9.8(-10)	8.0(-11)	3.1(-9)
$\frac{\pi}{140}$	5.3(-10)	5.3(-10)	3.8(-11)	1.6(-9)
$\frac{\pi}{160}$	3.1(-10)	3.1(-10)	1.9(-11)	9.7(-10)
$\frac{\pi}{180}$	1.9(-10)	1.9(-10)	1.0(-11)	6.1(-10)
$\frac{\pi}{200}$	1.3(-10)	1.3(-10)	6.3(-12)	3.9(-10)

estimate of the maximum truncation error per unit step per unit step multiplied by the length of the range of integration.

It is clear from Table 5 that the control based on the error per unit step per unit step provides a better estimate of the maximum error in Numerov's method and we shall consider in § 4.4 automatic implementations of Numerov's method which control the local error per unit step and the local error per unit step per unit step. These implementations will be referred to as NUMEROV1 and NUMEROV2 respectively.

Estimates of the errors are obtained by using the divided difference form for the sixth derivative of y, that is we approximate y^{vl}_{k+1} by $6!\, f[x_{k-5},x_{k-4},x_{k-3},x_{k-2},x_{k-1},x_k,x_{k+1}]$ which in the case of evenly spaced mesh points corresponds to using backward differences. We therefore need to calculate the solution at a minimum of seven mesh points before an estimate of the truncation error can be obtained.

Suppose we have reached the mesh point x_{k+1} with a constant steplength h and that at this point the estimated error per unit step or per unit step per unit step exceeds the tolerance parameter. Then the solution must be recalculated at the new mesh point x'_{k+1} where

$$x'_{k+1} = x_k + rh$$

with $r < 1$. Thus we shall require an estimate of the solution at the mesh point x'_{k-1} where

$$x'_{k-1} = x_k - rh$$

in order to use equation (4.2) to advance the solution to the new mesh point x'_{k+1}. Thus a disadvantage of a stepchanging mechanism in Numerov's method is the requirement of an interpolation procedure when decreases in steplength are performed. It is necessary to provide an estimate of the solution at x'_{k-1} to the same degree of accuracy as that produced in estimates obtained by Numerov's method. Thus we use the calculated solution at the mesh points x_k, x_{k-1}, x_{k-2},

x_{k-3}, x_{k-4}, x_{k-5} to generate the divided difference form of the fifth degree interpolation polynomial interpolating the solution at these points.

If the estimated error per unit step or per unit step per unit step at x_{k+1} is less than the tolerance parameter we consider whether an increase in steplength is justified. The steplength strategy will be discussed in §4.4.2.

It is possible to derive a generalisation of Numerov's method which is appropriate for integration over two unequal intervals of length h and k. We consider the new method to have the form

$$y_{n+1} = Ay_n + By_{n-1} + (Cf_{n+1} + Df_n + Ef_{n-1}) + R \qquad (4.18)$$

where

$$x_n = x_{n-1} + h, \; x_{n+1} = x_n + k.$$

When h = k the method given by (4.18) reduces to that of the familiar Numerov method for a constant steplength, that is

$$A = 2, \quad B = -1, \quad C = \frac{h^2}{12} = E, \quad D = 10\,\frac{h^2}{12}$$

and R is the leading term in the local truncation error given by

$$R = \frac{-h^6}{240}\, y_{n+1}^{vl} .$$

Substituting the following Taylor expansions

$$y_n = y_{n+1} - ky'_{n+1} + \frac{k^2}{2} y''_{n+1} - \frac{k^3}{6} y'''_{n+1} + \frac{k^4}{24} y_{n+1}^{1v} - \frac{k^5}{120} y_{n+1}^{v} + \frac{k^6}{720} y_{n+1}^{vl} + \cdots$$

$$y_{n-1} = y_{n+1} - (h+k)y'_{n+1} + \frac{(h+k)^2}{2} y''_{n+1} - \frac{(h+k)^3}{6} y'''_{n+1} + \frac{(h+k)^4}{24} y_{n+1}^{1v} - \frac{(h+k)^5}{120} y_{n+1}^{v} + \frac{(h+k)^6}{720} y_{n+1}^{vl} + \cdots$$

$$f_n = y''_n = y''_{n+1} - ky'''_{n+1} + \frac{k^2}{2} y_{n+1}^{1v} - \frac{k^3}{6} y_{n+1}^{v} + \frac{k^4}{24} y_{n+1}^{vl} + \cdots$$

$$f_{n-1} = y''_{n-1} = y''_{n+1} - (h+k)y'''_{n+1} + \frac{(h+k)^2}{2} y_{n+1}^{1v} - \frac{(h+k)^3}{6} y_{n+1}^{v} + \frac{(h+k)^4}{24} y_{n+1}^{vl} + \cdots$$

in equation (4.18) and equating the coefficients of y_{n+1} and the first

six derivatives of y_{n+1} we obtain the following values for the coefficients in (4.18):

$$A = \frac{(h+k)}{h}, \quad B = -\frac{k}{h}, \quad C = \frac{1}{12}(hk+k^2-h^2)$$

$$D = \frac{(h+k)}{12h}(h^2+3hk+k^2), \quad E = \frac{k}{12h}(h^2+hk-k^2)$$

$$R = -\frac{1}{720}(4k^5+10hk^4-10h^3k^2-4h^4k)\,y_{n+1}^{v}$$

$$-\frac{1}{1440}(-5k^6-15hk^5-5h^2k^4+15h^3k^3+13h^4k^2+3h^5k)y_{n+1}^{vl} . \quad (4.19)$$

Notice that when a constant steplength is used over the interval $[x_{n-1}, x_{n+1}]$, corresponding to the method in (4.2), y_{n+1} is a fifth order approximation to the solution at x_{n+1}; when the steplength is changed from h in $[x_{n-1}, x_n]$ to k in $[x_n, x_{n+1}]$, y_{n+1} is a fourth order approximation. We shall refer henceforth to the method of (4.18) - (4.19) as the generalised Numerov method and that of (4.2) simply as the Numerov method. The generalised method has been used by Burke and Seaton (1971, p. 61) in connection with solving a set of integro-differential equations; the equations are reduced to a system of linear algebraic equations by applying the generalised Numerov method. The method used by Burke and Seaton differs from that of Robertson (1956) which uses Numerov's method over a finite range of integration with a finite number of evenly spaced mesh points. The use of the generalised Numerov method allows Burke and Seaton to consider a finite range of integration comprised of a finite number of unevenly spaced mesh points; the mesh spacing is determined prior to the numerical integration of the equations and is calculated subject to an intuitive model which is set up to provide "function values with an approximately correct distribution of nodes". (see Burke and Seaton, p. 56).

We shall discuss in §4.4.2 the suitability of the generalised

Numerov method with an automatic steplength control with regard to solving the radial Schrödinger equation given by (4.4).

§ 4.4 Implementation of Numerov's method with automatic error control

We shall discuss the programs NUMEROV1 and NUMEROV2 which are automatic implementations of Numerov's method to solve $y'' = f(x, y)$. NUMEROV1 uses a local error per unit step criterion and NUMEROV2 a local error per unit step per unit step criterion. The steplength strategy used in both versions is discussed in § 4.4.2 and we shall see that in the case of the generalised Numerov method various questions are raised as to how best to effect the change over from the Numerov to the generalised Numerov formula (or vice versa) when a change in steplength is required.

The test program which used Numerov's method to solve the single channel radial Schrödinger equation, given by (4.4), for scattering of an electron by the static potential of atomic hydrogen is a special case of that listed in Appendix 2 which uses the method of Raptis and Allison (1977). A discussion of the Raptis and Allison method is given in Chapter 5; it will suffice here to say that this method is a generalisation of Numerov's method.

The subroutine NUMOV which solves the differential equation in programs NUMEROV1 and NUMEROV2 differs for the two programs only with respect to the type of error criterion used to control the steplength in the numerical integration of the differential equation.

§ 4.4.1 Programming the Numerov algorithm

Numerov's method applied to equation (4.4) is expressed as

$$y_{k+1} = 2y_k - y_{k-1} + \frac{h^2}{12}\left\{y''_{k+1} + 10y''_k + y_{k-1}''\right\}$$

$$= 2y_k - y_{k-1} + \frac{h^2}{12}\left\{\left[\frac{\ell(\ell+1)}{x_{k+1}^2} - E + V(x_{k+1})\right]y_{k+1} + 10\left[\frac{\ell(\ell+1)}{x_k^2} - E + V(x_k)\right]y_k + \left[\frac{\ell(\ell+1)}{x_{k-1}^2} - E + V(x_{k-1})\right]y_{k-1}\right\}$$

or

$$\left(1 - \frac{h^2}{12}\lambda_{k+1}\right)y_{k+1} = 2\left(1-\frac{h^2}{12}\lambda_k\right)y_k - \left(1-\frac{h^2}{12}\lambda_{k-1}\right)y_{k-1} + h^2\lambda_k y_k \qquad (4.20)$$

where

$$\lambda_r = \frac{\ell(\ell+1)}{x_r^2} - E + V(x_r).$$

Since values of the solution calculated at previous mesh points will be required before an estimate of the local error can be made we shall use the arrays XX, F to store the values of the mesh points and the corresponding calculated solutions; F(K + 1) will correspond to the calculated solution y_{k+1} at the mesh point x_{k+1} which is stored in XX(K + 1). We shall be required to provide two starting conditions, namely F(1) and F(2); the problem of starting the solution will be considered separately in §4.4.3.

We introduce the quantities

$$Y = \left(1-\frac{h^2}{12}\lambda_{k+1}\right)F(K+1), \quad YK = \left(1 - \frac{h^2}{12}\lambda_k\right)F(K),$$

$$YPREK = \left(1-\frac{h^2}{12}\lambda_{k-1}\right)F(K-1), \quad H2VF = h^2\lambda_k F(K),$$

$$V = \lambda_k, \quad H2 = h^2.$$

If the steplength h has been used as far as x_k we shall have

$$Y = \left(1-\frac{h^2}{12}\lambda_k\right)F(K), \quad YK = \left(1-\frac{h^2}{12}\lambda_{k-1}\right)F(K-1), \quad YPREK = \left(1-\frac{h^2}{12}\lambda_{k-2}\right)F(K-2)$$

with

$$F(K) = y_k, \quad F(K-1) = y_{k-1}, \quad F(K-2) = y_{k-2}, \quad H2VF = h^2\lambda_{k-1}F(K-1).$$

In order to advance the integration one step to x_{k+1} we must update the values of YK and YPREK for use in the next step; we do this by

setting

```
H2VF  =  H2* V* F(K)
YK    =  Y
YPREK =  YK
```

where V is still λ_k. Then we use the following FORTRAN statements to calculate the solution at x_{k+1}:

```
Y  =  2.0*YK - YPREK + H2VF                    (4.21)
F(K+1)  =  Y/(1.0 - H2*V/12.0).                (4.22)
```

Notice that if we write (4.1) as

$$y'' = f(x, y) = g(x)y$$

then V is just the function g(x) so that the second derivative of the solution at x_k is given by V*F(K). The calculation of this quantity is effected within the calculation of the quantity YK. We choose to evaluate the second derivative of the solution in this way in the program text rather than use a function subprogram, as was the case in program RADISH, since the use of equation (4.21) as a means of advancing the integration one step results in a decrease in the number of arithmetic operations required per step.

Suppose the initial steplength which must be supplied by the user is h and that we have calculated the solution at the mesh points XX(1) = h, XX(2) = 2h, ---, XX(K+1) = (k+1)h. The leading term in the truncation error per step at XX(K+1) is given by

$$-\frac{h^6}{240} y_{k+1}^{vl}$$

and we estimate this error by using the divided difference form for the sixth derivative of y. Subroutine DIVDIF sets up the table of sixth divided differences of the solution at the points XX(K+1), XX(K), XX(K-1), XX(K-2), XX(K-3), XX(K-4), XX(K-5). This routine will be described in more detail in §4.4.4. DIVDIF returns the value DD which is an estimate of the sixth derivative of the solution. Notice

however that the sixth divided difference table will tend to estimate the value near the midpoint of the range of mesh points considered rather than at x_{k+1} which is the point at which the estimate is required in order to consider what action if any must be taken in the next step, that is, whether an increase or decrease in steplength is required. The use of any divided difference table which uses past values of the solution to estimate the associated derivative of the solution at a given mesh point will inevitably introduce a 'lag' in that the estimated derivative curve will lag behind the true derivative curve. However our calculations show that this lag does not affect the accuracy of the solution obtained.

An estimate of the local error per unit step is given by

$$\text{TRERR} = -\frac{h^5}{240}\,\text{DD} \tag{4.23}$$

and this quantity is calculated in subroutine NUMOV of NUMEROV1. Program NUMEROV2 calculates

$$\text{TRERR} = -\frac{h^4}{240}\,\text{DD} \ . \tag{4.24}$$

§ 4.4.2 Steplength strategy

The user is required to specify a parameter EPS for use in the subroutine NUMOV; the value of EPS is transferred to NUMOV by the common block EKLL1. In the case of NUMEROV1 EPS is the largest allowed local error per unit step and in the case of NUMEROV2 it is the largest allowed local error per unit step per unit step.

Program NUMEROV2 calculates at each step

$$\text{TOL} = \text{EPS} \times \text{Max}\left\{1,\ |y_{k+1}|\right\} \tag{4.25}$$

so as to provide an absolute or relative error criterion according as the absolute value of the calculated solution is less or greater than unity, and the calculated y_{k+1} is accepted as satisfactory if

$$|\text{TRERR}| \leq \text{TOL} \qquad (4.26)$$

where TRERR is given by equation (4.24). If this inequality is not satisfied the step is rejected, a new value of the steplength αh is found and we start again from the mesh point $x_k - \alpha h$ (see §4.3) using an interpolated value for the solution at this point. Since the local error per unit step per unit step varies approximately as h^4, we argue as in §3.2 that a steplength αh would give a truncation error of approximately α^4 TRERR, and we choose

$$\alpha = \left(\frac{0.5\ \text{TOL}}{|\text{TRERR}|}\right)^{\frac{1}{4}} .$$

When the calculated y_{k+1} has been accepted as sufficiently accurate we consider whether or not a steplength increase is justified. In order to avoid additional interpolation it is necessary to restrict increases in steplength to a factor of C = 2. Notice that in program RADISH the value of C was also taken to be 2 but C is not confined to this value, since for any value of C which is specified by the user the implementation remains valid. By considering only increases in the steplength by a factor of 2 in subroutine NUMOV we can use the stored values of YPREK and Y to obtain the solution at the next step with a steplength of 2h.

Suppose we have reached the mesh point XX(K+1) with a constant steplength h and at this point we have

$$|\text{TRERR}| \leq \text{TOL} .$$

The steplength will be doubled only if

$$16\ |\text{TRERR}| < 0.5 \cdot \text{TOL}$$

for 3 consecutive steps. Then if a steplength of 2h is to be used in the next step we shall require values for the solution at the mesh points XX(K-1) and XX(K+1) calculated with a steplength 2h so that equation (4.2) may be used to advance the solution to the next mesh point. If we define

$$Y_r = \left(1 - \frac{(2h)^2}{12}\lambda_r\right) F(r) \qquad (4.27)$$

we shall require the values of Y_{K-1} and Y_{K+1} for use in equation (4.21) where Y_{K-1} and Y_{K+1} are the new values of YPREK and YK respectively. Now YPREK is given by

$$YPREK = \left(1 - \frac{h^2}{12}\lambda_{k-1}\right) F(K-1)$$

and

$$Y_{K-1} = \left(1 - \frac{(2h)^2}{12}\lambda_{k-1}\right) F(K-1) \ .$$

Thus we have

$$Y_{K-1} = 4\ YPREK - 3F(K-1) . \qquad (4.28)$$

Similarly

$$Y_{K+1} = 4Y - 3F(K+1) . \qquad (4.29)$$

Program NUMEROV1 accepts the calculated y_{k+1} if

$$|TRERR| \leqslant TOL \qquad (4.30)$$

where TRERR and TOL are given by equations (4.23) and (4.25) respectively. If the inequality in (4.30) is not satisfied the step is rejected, as in NUMEROV2, but now α is chosen to be

$$\alpha = \left(\frac{0.5TOL}{|TRERR|}\right)^{\frac{1}{5}}$$

since the local error per unit step varies approximately as h^5. The corresponding test which must be satisfied for 3 consecutive steps before the steplength is allowed to double is

$$32\,|TRERR| < 0.5\ TOL .$$

We consider now the steplength strategy for the generalised Numerov method. If we consider a local error per unit step criterion and if the method is used to advance the solution from x_k to x_{k+1} where

$$x_k = x_{k-1} + h \text{ and } x_{k+1} = x_k + h_1 \ , \ h_1 \neq h$$

an estimate of the local error per unit step is

$$\text{TRERR1} = \frac{1}{360}(5h^3h_1 + 2h^4 - 2h_1^4 - 5hh_1^3)\text{DD} \qquad (4.31)$$

where DD is an estimate for y^v_{k+1}. Suppose that we have reached the mesh point x_{n+1} with a constant steplength h. Then at this point an estimate of the local error per unit step is

$$\text{TRERR} = \frac{-h^5}{240}\text{DD}$$

where DD is an estimate for y^{v1}_{n+1}. Suppose that

$$|\text{TRERR}| > \text{TOL}.$$

Then a decrease in steplength is required and the calculation must be restarted from the mesh point x_n with a new steplength h_1 say. However the local error per unit step which previously varied approximately as h^5 will now vary approximately as

$$[(2 + 5\alpha) - \alpha^3(5 + 2\alpha)]\, h^4 \qquad (4.32)$$

by virtue of equation (4.31) where $h_1 = \alpha h$. We now require that

$$|\text{TRERR1}| \leq \text{TOL}.$$

However a difficulty arises if we consider using the form in (4.32) to provide a value for α ; we used the argument in NUMEROV1 and NUMEROV2 that if the local error per unit step and per unit step per unit step vary approximately as h^5 and h^4 respectively then a steplength αh will give an error of approximately α^5 TRERR and α^4 TRERR where TRERR in both cases involves an estimate of the sixth derivative of the solution. But notice that for the generalised Numerov method TRERR and TRERR1 involve an estimate for the sixth and fifth derivative of the solution respectively so we cannot combine the two estimates of the local error per unit step to provide a value for α. One possibility would be to perform the necessary decrease in steplength from h to $h_1 = \alpha h$ as if the next step were taken with the Numerov formula. Thus by employing the same strategy used for

decreases in NUMEROV1 we choose

$$\alpha = \left(\frac{0.5TOL}{|TRERR|}\right)^{\frac{1}{5}} .$$

It is now possible to use the generalised Numerov formula for the next step which has length αh, followed by the Numerov formula if this step is accepted. Notice that no interpolation is required. However a steplength αh which is acceptable for the Numerov method may well prove to be too large for the generalised method since the leading term of the local truncation error in the former method is a factor of h times that in the latter. This would necessitate a further decrease in steplength to $h_2 = \beta h_1 = \alpha\beta h$ and the problem immediately arises as to how to calculate β in practice. An estimate of the local error per unit step is

$$TRERR2 = \frac{1}{360}(5h^3h_2 + 2h^4 - 2h_2^4 - 5hh_2^3)DD \qquad (4.34)$$

where DD is an estimate for the fifth derivative of the solution at $x_{n+1} = x_n + h_2$. Thus the local error per unit step which previously varied as the quantity in (4.32) will now vary approximately as

$$[(2 + 5\alpha\beta) - \alpha^3\beta^3(5 + 2\alpha\beta)]h^4.$$

We require

$$|TRERR2| \leq TOL.$$

A suitable choice for β can be obtained in terms of the computed quantity TRERR1 but involves solving a quartic equation in β. This does not constitute a practical means of estimating the new steplength. It may be that some other strategy proves to be more satisfactory and we shall give no further consideration to the method. We concentrate instead on the familiar method of Numerov.

§ 4.4.3 The initial step

We are required to specify two starting conditions namely y_o and y_1 which represent the solution values at the mesh points x_o and x_1 =

$x_o + h$. These are calculated in the main routine of NUMEROV1 and NUMEROV2. The initial step of Numerov's method calculates y_2 and may be expressed as

$$y_2 = \frac{\left(2 + \frac{5}{6} h^2 \lambda_1\right)}{\left(1 - \frac{h^2}{12} \lambda_2\right)} y_1 - \frac{\left(1 - \frac{h^2}{12}\lambda_o\right)}{\left(1 - \frac{h^2}{12}\lambda_2\right)} y_o. \qquad (4.35)$$

With the completion of this step the cumulative and truncation errors equate and the truncation error at the end of this step is h times the truncation error per unit step in the case of program NUMEROV1 and h^2 times the truncation error per unit step per unit step in the case of NUMEROV2. In Chapter 7 we give a numerical comparison of various programs including RADISH, NUMEROV1 and NUMEROV2 to solve the radial Schrödinger equation (4.4). In order to provide a reasonable comparison the same starting point x_o must be used in each program and thus the same criterion must be used for choosing x_o in each program. In § 3.5.1 the criterion for choosing x_o was such that the first neglected term in the expansions for y_o and $h\dot{z}_o$ was less than 0.1 EPS. In order that the contribution of the error from y_o to y_2 in (4.35) be controlled we ask that a times the first neglected term in the expansion for y_o be less than 0.1hEPS in program NUMEROV1 and less than $0.1h^2$EPS in NUMEROV2, where

$$a = \left(\frac{1 - \frac{h^2}{12}\lambda_o}{1 - \frac{h^2}{12}\lambda_2}\right) < 1. \qquad (4.36)$$

Similarly we ask that b times the first neglected term in the expansion for y_1 be less than 0.1h EPS, $0.1h^2$ EPS in NUMEROV1, NUMEROV2 respectively where

$$b = \left(\frac{2 + \frac{5}{6} h^2 \lambda_1}{1 - \frac{h^2}{12} \lambda_2}\right) . \qquad (4.37)$$

The criterion adopted in Chapter 7 for choosing x_o is

$$|a_5|\, x_o^{L+5} \leq 0.1 \times EPS \times EPS \qquad (4.38)$$

in the notation of § 3.5.1. The condition given by (4.38) will ensure that the appropriate criterion for choosing y_o in each method is satisfied. In order that the contribution from the error in y_1 to that in y_2 be controlled we ask that

$$|b\delta y_1| < h^i \, EPS \qquad (4.39)$$

where $\delta y_1 = a_r(x_o+h)^{L+r}$ represents the first neglected term in the expansion for y_1 with $5<r\leq 9$ and $i = 1, 2$ in NUMEROV1, NUMEROV2 respectively. The first 8 coefficients of the expansion for the potential given by (3.50) are read as data in the main routine and the coefficients $a_2, ---, a_9$ are calculated with $a_1 = 1$. If the inequality in (4.39) is not satisfied for $r = 6, ---, 9$ then a more suitable steplength $\propto h$ is found with

$$\propto = \left(\frac{0.5\, h^i \, EPS}{|b\, a_9 (x_o+h)^{L+9}|} \right)^{\frac{1}{6-i}} .$$

If the inequality in (4.39) is satisfied with the new value of the steplength for $r = s$ say $(6 \leq s < 9)$ then the accepted value of y_1 is such that the first neglected term in the expansion for y_1 is $a_s(x_o+h)^{L+s}$.

The accepted values of y_o and y_1 are passed to the routine NUMOV in the argument list of NUMOV. It may happen however that the first estimate of the local error per unit step or per unit step per unit step is such that

$$|TRERR| > TOL$$

in which case a new value for y_1 must be provided by the routine NUMOV.

§ 4.4.4 Description of programs NUMEROV1, NUMEROV2

Figure 4 below shows the relationship between the seven routines of programs NUMEROV1 and NUMEROV2 which solve the radial Schrödinger equation given by equation (4.4).

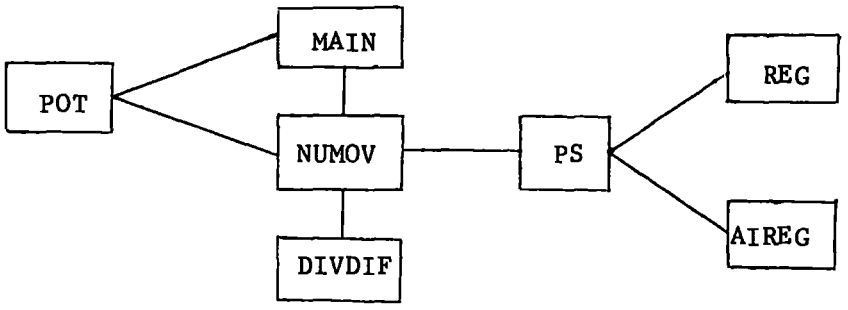

FIGURE 4

Program NUMEROV2 differs from NUMEROV1 only with respect to six FORTRAN statements which we discuss in part (ii) of § 5.7.3. So for the main part of this Section we shall describe program NUMEROV1; Appendix 2 contains a listing of the program EXPFIT1 which is an automatic implementation of the Raptis and Allison method with an error per unit step criterion. To obtain a listing of NUMEROV1 would require only a few minor modifications to EXPFIT1. Thus we have chosen to provide a listing of EXPFIT1; EXPFIT2 is the corresponding generalised version of NUMEROV2 which uses an error per unit step per unit step criterion in the Raptis and Allison method.

Double precision is used for all real variables. We shall discuss each of the routines in Figure 4 in this Section.

(i) The main routine

This calculates the values of the initial conditions y_o, y_1 as described in § 4.4.3. The first 8 coefficients in the expansion of the potential in the vicinity of the origin, given by equation (3.50) are read as data at line 21 and are stored in the array VCOEFF. The 9 coefficients in the expansion of the regular solution of (4.4) which are given by equations (1.15) of § 1.2 are calculated with $a_1 = 1$ and a_i replaced by a_{i+1}. Having read the coefficients V_1, V_2, ---, V_8 the program then reads the parameters EPS, L, E, PSIG and XF. C the factor by

which the steplength is increased is set to 2. For given values of EPS and XF an arbitrary number of calculations may be carried out for different values of the energy E and angular momentum L. Execution terminates when a negative value of L is read.

In the listing in Appendix 2, H is taken to be 0.1. This value is completely arbitrary; for comparison purposes the initial value of H in each of the programs tested for a particular problem must be the same in order to provide a meaningful comparison of the methods. It is possible to provide a more realistic value by using the power series given by equation (3.51) to estimate, a priori, the truncation error in y_2, as was done in § 3.5.1.

The common block EKLL1 is used to transfer parameters to the subroutines NUMOV, PS and POT.

(ii) Subroutine NUMOV

The subroutine NUMOV which is written in double precision form solves the differential equation

$$y'' = f(x, y).$$

NUMOV has been written so that the structure of the routine is similar to that of DEVOG. Indeed the NUMOV routine is identical to routine RAPAL described in Chapter 5 (see § 5.6) with the exception of a few statements. The input to NUMOV is as for RAPAL; NUMOV is obtained from RAPAL by deleting lines (240 - 254)*, 297 - 300, 331, 346 - 353, 371 and 373 - 378.

---

* The line numbers quoted are those appearing on the left hand side of the listing of EXPFIT1, corresponding to the sequential numbering of the lines in a file containing EXPFIT1.

In addition the labels in lines 301 - 302, 354, 379 are deleted and line 262 is replaced by the following FORTRAN statement:

```
9   CALL   DIVDIF (DD, K, J, JD)
```

(see part (iii)).

(iii) Subroutine DIVDIF

This subroutine sets up the tables of sixth divided differences of the solution at the mesh points XX(K+1), XX(K), ---, XX(K-5). The elements of the table are stored in the array D the dimensions of which are 6 by 7. The rth column, r = 1, ---, 6, of the table is stored in D(r,I), I=1, ---, 8-r. The array X(7) stores the values of the seven mesh points which are being considered:

$$X(I) = XX(K + I - 6),\ I = 1, ---, 7.$$

The estimate of the sixth derivative of the solution is given by

$$DD = \frac{D(6, 1) - D(6, 2)}{X(1) - X(7)} .$$

The arguments of DIVDIF are DD, K, J and JD; J is 1 on initial entry to DIVDIF and is subsequently set to zero to distinguish between the first and later calls. JD>0 signifies that a decrease in steplength has been performed.

Subroutine DIVDIF as listed in Appendix 2 is for the method of Raptis and Allison which requires a more complex routine for estimating the derivatives in the local truncation error; notice that the listed version of DIVDIF involves two extra parameters in the argument list. However the version of DIVDIF which we require for use in conjunction with the routine NUMOV is contained within lines 415, 417-418 and 467-505 of EXPFIT 1 with lines 417 and 482 replaced by

```
      DIMENSION X(7)
```

and

```
    1 CONTINUE
```

respectively; we also reduce the number of arguments in the call to DIVDIF by two to the four discussed above, namely DD, K, J and JD. A COMMON statement which holds the arrays XX, F and D in common enables values of D to be used in the decrease section in NUMOV.

(iv) Subroutine PS

The arguments K, JP and JCONV of PS are described in comment cards in the text of the program. The function of this routine is as described in §3.5.3; one less argument is required in the present version of PS since the mesh points and the corresponding solutions are stored in the arrays XX, F which are relayed to PS by means of a COMMON statement. This is the only difference from routine PS as used in program RADISH.

(v) The function POT

This subprogram is called by NUMOV and the main routine of the program to calculate the potential V(x). The decision as to when a phase shift should be calculated is taken in POT; when

$$\frac{L(L+1)}{x^2} < E \quad \text{and} \quad |V(x)| < \text{EPS} \left| E - \frac{L(L+1)}{x^2} \right|$$

the value IPS = 1 is returned to the calling routine.

(vi) The functions REG and AIREG

REG and AIREG have been described in §3.5.4 and they are used in exactly the same form as in program RADISH.

§4.4.5 Test runs

Programs NUMEROV1 and NUMEROV2 have been tested for the same set of problems as tested by program RADISH. Details of the test problems and results will appear in Chapter 7.

We consider here the solution of the following problem over a specified range for different values of the accuracy parameter EPS:

$$y'' = \left(\frac{2}{x^2} - k^2\right)y;\ y_o = x_o j_1(kx_o),\ y_1 = x_1 j_1(kx_1),\ x_o = 0.01 \ .$$

This problem is problem (ii) of §3.6 for which the exact solution is $y = xj_1(kx)$. We solve the problem for k = 0.1, 0.2, 0.5, 1.0, 2.0, 5.0 with EPS = 10^{-n}, n = 3, 4, 6, 8. We provide exact starting values for x_o, x_1, y_o and y_1 in the main routine along with the initial value of the steplength which is chosen to be

$$\left(\frac{240\ EPS}{|y_o^{vl}|}\right)^{\frac{1}{5}} \quad , \quad \left(\frac{240\ EPS}{|y_o^{vl}|}\right)^{\frac{1}{4}}$$

in the case of NUMEROV1, NUMEROV2 respectively where we use the approximation $y_o^{vl} = k^6 y_o$. If the initial steplength is very large it is possible for the calculation to proceed to XF (the end point of the integration which we take to be 20) without providing an estimate of the truncation error. To avoid this difficulty the main routine takes H to be either the value supplied by the user or (XF - XO)/7, whichever is the smaller.

Tables 6 and 7 in the next Section show the results obtained using NUMEROV1 and NUMEROV2 in solving this problem; results for the same problem solved by RADISH are tabulated in Table 3 of §3.7.

§4.4.6 Test results

A study of Tables 6 and 7 shows that NUMEROV2 is more effective in controlling the global error than is NUMEROV1. Notice that the number of function evaluations N required in NUMEROV2 is generally

far greater, sometimes by more than a factor or two, than the corresponding number in NUMEROV1. This is a consequence of the more stringent error requirement in NUMEROV2 when H is less than unity; the relatively few cases where N in NUMEROV1 exceeds N in NUMEROV2 occur when the steplength accepted in NUMEROV2 as sufficiently small exceeds that accepted in NUMEROV1 and both steplengths exceed unity. Note that the value of H listed in Tables 6 and 7 is the initial steplength supplied by the main routine for use in NUMOV and that this value of H is not necessarily the initial steplength accepted by NUMOV as sufficiently small.

The values of a_{max} in Tables 6 and 7 are 6.64 and 7.31 respectively which are considerably smaller than that in Table 3 for de Vogelaere's method. However the value of a_{max} does not reflect the superiority of the Numerov method with a local error per unit step per unit step criterion over that with a local error per unit step criterion; this superiority is apparent from a study of the scaled maximum errors in Tables 6 and 7. A more useful insight into the relative performance of each method in terms of its reliability in solving the problem is provided by considering the number of cases where a, which is the factor by which the scaled maximum error exceeds the corresponding EPS, exceeds unity as a percentage of the total number of cases tested. Tables 6 and 7 each provide 24 cases corresponding to 6 values of the parameter k and 4 values of EPS. From Table 6 we see that a exceeds unity in 12 cases (for EPS = 10^{-6}, 10^{-8}); thus the corresponding percentage is a_p = 50%. From Table 7 we have a_p = 25% and the superiority of NUMEROV2 over NUMEROV1 in controlling the global error is immediately apparent; the penalty incurred however is an increase in the number of function evaluations required in NUMEROV2.

Note that for Table 3 $a_p \simeq 79\%$ which far exceeds the corresponding values of Tables 6 and 7. However notice that the factor a over the range of EPS and k values exceeds the value five only at the more stringent accuracy requirements. If we denote by a_{p5} the percentage of cases where a exceeds 5 times EPS then for Table 3 $a_{p5} = 25\%$. The corresponding values for Tables 6 and 7 are $a_{p5} \simeq 4\%$ in both cases. We conclude that NUMEROV2 solves this problem more reliably than NUMEROV1 and RADISH in terms of controlling the global error.

NUMEROV 1

TABLE 6

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.38(-5)	1.93(-5)	2.856	6
	0.5	2.20(-4)	6.51(-4)	5.02(-4)	2.856	25
	1.0	5.59(-4)	4.85(-4)	5.21(-4)	2.856	45
	2.0	4.30(-4)	5.40(-4)	6.39(-4)	2.239	69
	5.0	4.79(-4)	4.88(-4)	4.85(-4)	0.621	155
10^{-4}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.38(-5)	1.93(-5)	2.856	6
	0.5	5.36(-5)	2.84(-5)	5.04(-5)	2.856	34
	1.0	6.34(-5)	5.42(-5)	6.43(-5)	2.856	55
	2.0	6.00(-5)	7.95(-5)	8.49(-5)	1.413	98
	5.0	6.30(-5)	6.74(-5)	6.96(-5)	0.392	242
10^{-6}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.25(-6)	2.70(-6)	2.60(-6)	2.856	28
	0.5	1.89(-6)	2.18(-6)	1.68(-6)	2.856	63
	1.0	2.24(-6)	1.80(-6)	1.93(-6)	1.484	118
	2.0	1.64(-6)	1.75(-6)	2.09(-6)	0.562	225
	5.0	2.05(-6)	2.22(-6)	2.30(-6)	0.156	548
10^{-8}	0.1	3.38(-8)	2.61(-8)	1.49(-8)	2.856	37
	0.2	4.12(-8)	2.14(-8)	4.55(-8)	2.856	64
	0.5	6.08(-8)	6.64(-8)	3.73(-8)	1.559	140
	1.0	4.67(-8)	3.72(-8)	3.62(-8)	0.591	304
	2.0	3.68(-8)	3.62(-8)	4.19(-8)	0.224	597
	5.0	4.19(-8)	4.10(-8)	4.11(-8)	0.062	1491

NUMEROV 2

TABLE 7

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.38(-5)	1.93(-5)	2.856	6
	0.5	4.37(-4)	3.88(-3)	3.15(-3)	2.856	8
	1.0	2.75(-4)	2.44(-4)	2.90(-4)	2.856	47
	2.0	1.27(-4)	1.82(-4)	1.94(-4)	2.739	94
	5.0	5.43(-5)	5.83(-5)	5.92(-5)	0.551	253
10^{-4}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.38(-5)	1.93(-5)	2.856	6
	0.5	4.84(-5)	5.83(-5)	4.65(-5)	2.856	35
	1.0	3.03(-5)	2.50(-5)	2.87(-5)	2.856	65
	2.0	1.46(-5)	1.62(-5)	1.95(-5)	1.540	137
	5.0	6.84(-6)	6.77(-6)	7.46(-6)	0.310	400
10^{-6}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	7.31(-6)	4.50(-6)	2.47(-6)	2.856	26
	0.5	5.98(-7)	7.12(-7)	4.62(-7)	2.856	83
	1.0	3.26(-7)	2.62(-7)	2.67(-7)	1.638	188
	2.0	1.43(-7)	1.44(-7)	1.69(-7)	0.487	418
	5.0	6.45(-8)	6.88(-8)	7.16(-8)	0.098	1304
10^{-8}	0.1	2.43(-8)	2.04(-8)	1.12(-8)	2.856	38
	0.2	1.56(-8)	8.27(-9)	2.38(-8)	2.856	73
	0.5	1.16(-8)	5.77(-9)	6.27(-9)	1.742	216
	1.0	4.03(-9)	3.27(-9)	3.63(-9)	0.518	539
	2.0	2.27(-9)	1.91(-9)	2.14(-9)	0.154	1249
	5.0	7.79(-10)	7.98(-10)	8.01(-10)	0.031	3965

CHAPTER 5

The Raptis and Allison method

Introduction

This method has been developed by Raptis and Allison (1977, preprint) specifically to solve the radial Schrödinger equation

$$y''(x) = \left[\frac{\ell(\ell+1)}{x^2} - E + V(x)\right] y(x). \qquad (5.1)$$

The method takes into account the known form of the solution of (5.1) in the asymptotic region. Some work has been done on methods for solving second order differential equations which exploit the a priori knowledge of the form of the solution (see, for example, Gautschi 1961; Gordon 1969; Lyche 1972). We shall concentrate however on the work of Raptis and Allison which is a development of some earlier work of Lyche (1972) in which it is recognised that for problems of the form

$$y^{(r)}(x) = f(x, y),\ y^{(j)}(a) = 0,\ j=0, 1, \text{---}, r-1 \qquad (5.2)$$

where the form of the solution is periodic or exponential a polynomial approximation to the solution is not always the best approximation which may be applied. The multistep method

$$\sum_{i=0}^{k} \alpha_i y_{n+i} = h^r \sum_{i=0}^{k} \beta_i f(x_{n+i}, y_{n+i}) \qquad (5.3)$$

where $\alpha_k = 1$ and α_o and β_o do not both vanish can be used to solve (5.2) and a polynomial approximation of degree p is such that the operator

$$\mathcal{L}[y(x); h] = \sum_{i=0}^{k} \alpha_i y(x+ih) - h^r \sum_{i=0}^{k} \beta_i y^{(r)}(x + ih) \qquad (5.4)$$

annihilates $y(x) = 1, x, x^2, \text{---}, x^{p+r-1}$. If instead we require that $\mathcal{L}$ annihilates functions of the form e^{wx} or $x^m e^{wx}$ which would be more appropriate in the case of solving (5.1) the penalty introduced is that the coefficients in (5.3) are dependent on the steplength h.

Lyche considers the multistep method

$$\sum_{i=o}^{k} \alpha_i(h)\, y_{n+i} \;=\; h^r \sum_{i=o}^{k} \beta_i(h)\, y_{n+i}^{(r)} \qquad (5.5)$$

with first and second characteristic polynomials given by

$$\rho(\xi) \;=\; \sum_{i=o}^{k} \alpha_i(h)\, \xi^i$$

and

$$\sigma(\xi) \;=\; \sum_{i=o}^{k} \beta_i(h)\, \xi^i .$$

Then if the operator $\mathcal{L}$ is defined by

$$\mathcal{L}[y(x);h] \;=\; \sum_{i=o}^{k} \alpha_i(h) y(x+ih) \;-\; h^r \sum_{i=o}^{k} \beta_i(h) y^{(r)}(x+ih) \qquad (5.6)$$

we can use the following result which appears in the form of a Lemma in Lyche (1972) and which is appended by Raptis and Allison (1977). The appended result reads:

Suppose h is fixed and $w \in \mathbb{C}$. Let $n \geq r$ if $w = 0$ and $n \geq 1$ otherwise. Then

$$\mathcal{L}[x^m e^{wx};\, h] = 0, \quad m = 0, 1, \text{---}, n - 1$$

and

$$\mathcal{L}[x^n e^{wx};\, h] \neq 0$$

if and only if the function ϕ given by

$$\phi(\xi) \;=\; \frac{\rho(\xi)}{\log^r(\xi)} \;-\; \sigma(\xi) \qquad (5.7)$$

has a zero of exact multiplicity s at $\xi = e^{wh}$, where $s = n$ if $w \neq 0$ and $s = n - r$ if $w = 0$.

The method of Raptis and Allison is a two step method ($k = 2$) and we derive the method in the following Section.

§ 5.1 Derivation of the Raptis and Allison method

We consider the method

$$\sum_{i=o}^{2} \alpha_i (h)\, y(x + ih) = h^2 \sum_{i=o}^{2} \beta_i (h)\, y''(x + ih) \qquad (5.8)$$

for solving (5.2) with r = 2. We have $\alpha_2(h) = 1$ and we can make use of the following consistency conditions for the method:

$$\rho(1) = \rho'(1) = 0, \quad \rho''(1) = 2\sigma(1) .$$

Thus we have

$$\alpha_o (h) = 1 = \alpha_2(h), \quad \alpha_1(h) = -2$$

$$\beta_o (h) + \beta_1 (h) + \beta_2 (h) = 1 .$$

We can evaluate the β coefficients by using Lyche's result with r = 2. The coefficients are then given by the solution of

$$\rho(e^{\pm w_i h}) = \log^2 (e^{\pm w_i h}) \, \sigma (e^{\pm w_i h}), \; i = 1, 2 . \qquad (5.9)$$

In order that the reciprocal of a root of an equation also be a root, as is required in (5.9), we must have

$$\beta_o(h) = \beta_2(h).$$

Thus the first and second characteristic polynomials in (5.9) are

$$\rho(\xi) = \xi^2 - 2\xi + 1$$

$$\sigma(\xi) = \beta_o(h)\, \xi^2 + \beta_1(h)\xi + \beta_o(h) . \qquad (5.10)$$

Substitution of (5.10) in (5.9) leads to

$$e^{2\lambda} - 2e^{\lambda} + 1 = \beta_o(h)[1 + e^{2\lambda}]\, \lambda^2 + \lambda^2 e^{\lambda}\, \beta_1(h)$$

$$e^{2\mu} - 2e^{\mu} + 1 = \beta_o(h)[1 + e^{2\mu}]\, \mu^2 + \mu^2 e^{\mu} \beta_1(h) \qquad (5.11)$$

where $\lambda = w_1 h$, $\mu = w_2 h$ and the solution of the system (5.11) is

$$\beta_o(h) = \frac{\dfrac{(1 - e^{\lambda})^2 e^{\mu}}{\lambda^2} - \dfrac{(1 - e^{\mu})^2 e^{\lambda}}{\mu^2}}{e^{\mu}(1+e^{2\lambda}) - e^{\lambda}(1+e^{2\mu})} = \beta_2(h)$$

$$\beta_1(h) = \frac{\dfrac{(1 - e^{\lambda})^2 (1 + e^{2\mu})}{\lambda^2} - \dfrac{(1 - e^{\mu})^2 (1 + e^{2\lambda})}{\mu^2}}{e^{\lambda}(1 + e^{2\mu}) - e^{\mu}(1 + e^{2\lambda})} \qquad (5.12)$$

For the application of (5.8) to the solution of the Schrödinger equation (5.1) it is convenient to take $w_1 = w$ and $w_2 = 0$ since the asymptotic form of the solution of (5.1) involves a single exponential argument.

Notice that the method may be applied to both the bound state problem (where $E < 0$ in (5.1)) and the scattering problem by taking w to be real and imaginary respectively. We are interested in the scattering problem and we set $w = ik$ in (5.12). We thus arrive at the following forms for $\beta_o(h)$ and $\beta_1(h)$:

$$\beta_o(h) = \frac{k^2h^2 - 2(1 - \cos kh)}{2k^2h^2(1 - \cos kh)} = \beta_2(h) \qquad (5.13a)$$

$$\beta_1(h) = \frac{2 + (k^2h^2 + 2)\cos kh}{k^2h^2(1 - \cos kh)} \qquad (5.13b)$$

and the corresponding operator $\mathcal{L}$ annihilates the functions

$$1,\ x,\ x^2,\ x^3,\ \sin kx,\ \cos kx$$

for $k \neq 0$.

For small values of kh, $\beta_o(h)$ and $\beta_1(h)$ are seen to have the following power series expansions:

$$\beta_2(h) = \beta_o(h) = \frac{1}{12}\left\{1 + \frac{(kh)^2}{20} + \frac{(kh)^4}{504} + \cdots\right\} \qquad (5.14a)$$

$$\beta_1(h) = \frac{1}{6}\left\{5 - \frac{(kh)^2}{20} - \frac{(kh)^4}{504} - \cdots\right\}. \qquad (5.14b)$$

Now if we take $w = 0$ we see from (5.14) that

$$\beta_o(h) = \frac{1}{12} = \beta_2(h),\ \beta_1(h) = \frac{10}{12}$$

corresponding to the use of Numerov's method in which $\mathcal{L}$ annihilates the functions $1, x, x^2, x^3, x^4, x^5$. Thus Numerov's method is seen to be a special case of (5.8).

§ 5.2 The local truncation error and its estimation for a fixed steplength

For an arbitrary function y(x) having p continuous derivatives we define the functional

$$\mathcal{L}[y(x);h] = y(x+2h) - 2y(x+h) + y(x) - h^2\left\{\beta_o(h)y''(x+2h) + \beta_1(h)y''(x+h) + \beta_o(h)y''(x)\right\} \qquad (5.15)$$

and we make use of the following Taylor expansions:

$$y(x+2h) = y(x)+2hy'(x)+\cdots+\frac{1}{p!}(2h)^p y^{(p)}(x)+\frac{1}{p!}h^{p+1}\int_0^2(2-s)^p y^{(p+1)}(x+sh)ds$$

$$y(x+h) = y(x)+hy'(x)+\cdots+\frac{1}{p!}h^p y^{(p)}(x)+\frac{1}{p!}h^{p+1}\int_0^1(1-s)^p y^{(p+1)}(x+sh)ds$$

$$y''(x+2h) = y''(x)+2hy'''(x)+\cdots+\frac{1}{(p-2)!}(2h)^{p-2}y^{(p)}(x)$$

$$+\frac{1}{(p-2)!}h^{p-1}\int_0^2(2-s)^{p-2}y^{(p+1)}(x+sh)ds$$

$$y''(x+h) = y''(x)+hy'''(x)+\cdots+\frac{1}{(p-2)!}h^{p-2}y^{(p)}x$$

$$+\frac{1}{(p-2)!}h^{p-1}\int_0^1(1-s)^{p-2}y^{(p+1)}(x+sh)ds$$

in (5.15). Then for p = 5 we have:

$$\begin{aligned}\mathcal{L}[y(x);h] = {} & h^2[1-2\beta_0(h)-\beta_1(h)]y''(x)+h^3[1-2\beta_0(h)-\beta_1(h)]y'''(x)\\ & +h^4\left[\frac{7}{12}-2\beta_0(h)-\frac{1}{2}\beta_1(h)\right]y^{1v}(x)\\ & +h^5\left[\frac{1}{4}-\frac{4}{3}\beta_0(h)-\frac{1}{6}\beta_1(h)\right]y^{v}(x)\\ & +h^6\left[\frac{1}{120}\int_0^2(2-s)^5y^{v1}(x+sh)ds-\frac{1}{60}\int_0^1(1-s)^5y^{v1}(x+sh)ds\right.\\ & \left.-\frac{1}{6}\beta_0(h)\int_0^2(2-s)^3y^{v1}(x+sh)ds-\frac{1}{6}\beta_1(h)\int_0^1(1-s)^3y^{v1}(x+sh)ds\right].\end{aligned} \tag{5.16}$$

Now if we make use of the identity

$$2\beta_0(h)+\beta_1(h) = 1$$

from § 5.1 we can express (5.16) in the simplified form

$$\begin{aligned}\mathcal{L}[y(x);h] = {} & \frac{h^4}{12}[1-12\beta_0(h)]y^{1v}(x)+\frac{h^5}{12}[1-12\beta_0(h)]y^{v}(x)\\ & +\frac{h^6}{120}\int_0^2 G(s)y^{v1}(x+sh)ds\end{aligned} \tag{5.17}$$

with

$$G(s) = (2-s)^5 - 2(1-s)_+^5 - 20\beta_0(h)(2-s)^3-20\beta_1(h)(1-s)_+^3 .$$

It is possible to show that $\beta_0(h) \geq \frac{1}{12}$ by considering the function

$$\beta_o(h) - \frac{1}{12} = \frac{6[k^2h^2 - 2(1-\cos kh)] - (1-\cos kh)k^2h^2}{12\, k^2h^2\,(1-\cos kh)} \qquad (5.18)$$

where we have substituted equation (5.13a) for $\beta_o(h)$. For all values of kh the numerator in (~~5.20~~ 5.18) is non-negative. Thus $\beta_o(h) \geqslant \frac{1}{12}$.

Now for $0 \leqslant s \leqslant 1$,

$$G(s) = -20\beta_o(h)[s^3 + 6(1-s)] + [s^5 + 10(1-s)] \qquad (5.19)$$

and for $1 \leqslant s \leqslant 2$,

$$G(s) = (2-s)^3[(2-s)^2 - 20\beta_o(h)] . \qquad (5.20)$$

Thus by using the fact that $\beta_o(h) \geqslant \frac{1}{12}$ we see from (5.19) and (5.20) that $G(s) \leqslant 0$ for $0 \leqslant s \leqslant 2$. Hence we may express (5.17) as

$$\mathcal{L}[y(x);h] = \frac{h^4}{12}[1-12\beta_o(h)]y^{1v}(x)+\frac{h^5}{12}[1-12\beta_o(h)]y^{v}(x)+h^6 c y^{vl}(x+\theta h),$$

$$0 < \theta < 2, \qquad (5.21)$$

with

$$c = \frac{1}{120}\int_0^2 G(s)ds$$

$$= \frac{1}{360}[16-210\beta_o(h)] .$$

Notice that when $\beta_o(h) = \frac{1}{12}$ equation (5.21) reduces to

$$\mathcal{L}[y(x);h] = \frac{-h^6}{240} y^{vl}(x+\theta h), \quad 0 < \theta < 2 \qquad (5.22)$$

which represents the local truncation error in Numerov's method.

Now if we use the form for $\beta_o(h)$ given by (5.14a) we see that the leading term of the local truncation error is proportional to h^6 and is given by

$$\frac{-h^6}{240}\left\{ k^2\, y^{1v}(x) + y^{vl}(x)\right\}. \qquad (5.23)$$

Notice that (5.23) vanishes for $y = \sin kx$ and $y = \cos kx$. In order to estimate the quantity in (5.23) we consider using the divided difference forms for the fourth and sixth derivatives of y, that is we approximate y^{1v}_{k+1} by $4!\, f[x_{k-3}, x_{k-2}, x_{k-1}, x_k, x_{k+1}]$ and y^{vl}_{k+1} by $6!\, f[x_{k-5}, x_{k-4}, x_{k-3}, x_{k-2}, x_{k-1}, x_k, x_{k+1}]$ which in the case of evenly spaced mesh points corresponds to using backward differences.

Thus we consider constructing a sixth divided difference table for the calculated solution from which we can extract both the fourth and sixth divided differences needed for estimating (5.23). However it was observed in §4.4 that the sixth divided difference table will tend to output an estimate for the sixth derivative of the solution at the midpoint of the range of mesh points considered, that is at x_{k-2}; similarly the fourth derivative will be estimated at the mesh point x_{k-1}. It is important however that the estimates for the fourth and sixth derivatives appearing in (5.23) are made for the same mesh point. The difficulty can be overcome in the case of evenly spaced mesh points by approximating y^{1v}_{k+1} by $4!\, f[x_{k-4}, x_{k-3}, x_{k-2}, x_{k-1}, x_k]$. But in the case of unevenly spaced mesh points the problem must still be resolved. Notice that we can express

$$k^2 y^{1v}(x) + y^{v1}(x)$$

as the fourth derivative of

$$k^2 y(x) + y''(x) \tag{5.24}$$

with respect to x. Hence it is possible to estimate (5.23) by constructing a fourth divided difference table for (5.24) over a range of five mesh points. When solving the radial Schrödinger equation given by (5.1) we use

$$k^2 y(x) + y''(x) = \left[\frac{\ell(\ell+1)}{x^2} + V(x)\right] y(x). \tag{5.25}$$

We shall discuss the necessary implementation for estimating the local error in §5.7.

§5.3 A bound for the local truncation error

If $y(x)$ is the exact solution of $y'' = f(x,y)$ and we assume that the starting values at x_n, x_{n+1}, are exact, that is

$$y_n = y(x_n), \quad y_{n+1} = y(x_{n+1})$$

$$f_n = y''(x_n), \qquad f_{n+1} = y''(x_{n+1})$$

then we have

$$\mathcal{L}[y(x_{n+1});h] = y(x_{n+2})-2y(x_{n+1})+y(x_n)-h^2\{\beta_o(h)y''(x_{n+2}) + [1-2\beta_o(h)]\, y''(x_{n+1})+\beta_o(h)y''(x_n)\}$$

$$= y(x_{n+2})-2y_{n+1}+y_n-h^2\{\beta_o(h)y''(x_{n+2})+[1-2\beta_o(h)]f_{n+1} + \beta_o(h)f_n\}. \qquad (5.26)$$

Thus the truncation error at x_{n+2} is

$$y(x_{n+2})-y_{n+2} = y(x_{n+2})-2y_{n+1}+y_n-h^2\{\beta_o(h)f_{n+2}+[1-2\beta_o(h)]f_{n+1}+\beta_o(h)f_n\}$$

$$=\mathcal{L}[y(x_{n+1});h]+h^2\beta_o(h)\{y''(x_{n+2})-f_{n+2}\}. \qquad (5.27)$$

To obtain a bound on this error we assume the usual Lipschitz condition

$$|f(x, y) - f(x,\eta)| \leq K|y-\eta|$$

for all x in the appropriate interval [a, b] and all finite y and η. We also assume a bound on the fourth, fifth and sixth derivatives of y for all x in [a, b]:

$$|y^{1v}(x)| \leq M_4, |y^{v}(x)| \leq M_5, |y^{vl}(x)| \leq M_6.$$

It is necessary, in addition, on inspection of equation (5.21) to provide a bound for each of the quantities

$$1 - 12\,\beta_o(h) \qquad (5.28a)$$

$$16 - 210\beta_o(h) \qquad (5.28b)$$

It is clear from the form of the leading term in the local truncation error that we shall require bounds of the form

$$|1 - 12\beta_o(h)| \leq Ak^2h^2$$

$$|16 - 210\beta_o(h)| \leq B$$

where A and B are constants which are suitably defined for $h \leq$ some h_o.

We may write (5.28a) as

$$1-12\beta_o(h) = \frac{k^2h^2(1-\cos kh)-6k^2h^2+12(1-\cos kh)}{k^2h^2(1-\cos kh)}$$

$$= \frac{k^2h^2\left[\frac{k^2h^2}{2} - \frac{k^4h^4}{24} - R_6\right] - 6k^2h^2 + 12\left[\frac{k^2h^2}{2} - \frac{k^4h^4}{24} + \frac{k^6h^6}{720} - R_8\right]}{k^2h^2\left[\frac{k^2h^2}{2} - \frac{k^4h^4}{24} - R_6\right]} \qquad (5.29)$$

where

$$R_6 = \frac{-k^6h^6}{6!}\cos\xi \quad , \quad R_8 = \frac{k^8h^8}{8!}\cos\eta \, , \quad 0 < \xi, \eta < kh.$$

Further simplification of equation (5.29) leads to

$$1-12\beta_o(h) = \frac{\left[-\frac{k^6h^6}{40} + \frac{k^8h^8}{720}\cos\xi - \frac{k^8h^8}{2.1680}\cos\eta\right]}{\frac{1}{2}k^4h^4\left[1 - \frac{k^2h^2}{12} + \frac{k^4h^4}{360}\cos\xi\right]}$$

$$= \frac{\left[-\frac{k^2h^2}{20} + \frac{k^4h^4}{360}\cos\xi - \frac{k^4h^4}{1680}\cos\eta\right]}{\left[1 - \frac{k^2h^2}{12} + \frac{k^4h^4}{360}\cos\xi\right]} \qquad (5.30)$$

Now for $0 \leq kh \leq \pi$ it is possible to show that the denominator of (5.30) which may be written equivalently as

$$d(kh) = \frac{2}{k^2h^2}(1-\cos kh) \qquad (5.31)$$

is a monotonic decreasing function of kh; for $0 \leq kh \leq \pi$ we have

$$d(kh) \geq \frac{4}{\pi^2} \, .$$

Thus for $0 \leq kh \leq \pi$

$$\left|1-12\beta_o(h)\right| \leq \left(\frac{k^2h^2}{20} + \frac{k^4h^4}{360} + \frac{k^4h^4}{1680}\right)\frac{\pi^2}{4}$$

$$= k^2h^2\left(\frac{1}{20} + \frac{17}{5040}k^2h^2\right)\frac{\pi^2}{4}$$

so that

$$\left|1-12\beta_o(h)\right| \leq k^2h^2\left(\frac{1}{20} + \frac{17}{5040}\pi^2\right)\frac{\pi^2}{4}$$

$$< \frac{k^2h^2}{4} \, . \qquad (5.32)$$

Similarly we write (5.28b) as

$$16-210\beta_o(h) = \frac{16k^2h^2(1-\cos kh) - 105k^2h^2 + 210(1-\cos kh)}{k^2h^2(1-\cos kh)}$$

$$= \frac{\left[-\frac{3}{2} - \frac{3}{4}k^2h^2 + \frac{2}{45}k^4h^4\cos\xi - \frac{1}{96}k^4h^4\cos\eta\right]}{\left[1 - \frac{k^2h^2}{12} + \frac{k^4h^4}{360}\cos\xi\right]} .$$

Thus for $0 \leq kh \leq \pi$,

$$\left|16-210\beta_o(h)\right| \leq \left(\frac{3}{2} + \frac{3}{4}k^2h^2 + \frac{2}{45}k^4h^4 + \frac{1}{96}k^4h^4\right)\frac{\pi^2}{4}$$

$$\leq \left(\frac{3}{2} + \frac{3}{4}\pi^2 + \frac{79}{1440}\pi^4\right)\frac{\pi^2}{4}$$

$$< 36 . \qquad (5.33)$$

Thus for $0 \leq kh \leq \pi$ we obtain the bound

$$\left|y(x_{n+2})-y_{n+2}\right| \leq \frac{h^4}{12}\left|1-12\beta_o(h)\right|\left|y^{1v}(x)\right| + \frac{h^5}{12}\left|1-12\beta_o(h)\right|\left|y^{v}(x)\right|$$

$$+\frac{h^6}{360}\left|16-210\beta_o(h)\right|\left|y^{vl}(x+\vartheta h)\right| + h^2\left|\beta_o(h)\right| K\left|y(x_{n+2})-y_{n+2}\right| .$$

We have shown previously that $\beta_o(h) \geq \frac{1}{12}$ and we seek an upper bound on $\beta_o(h)$ for $0 \leq kh \leq \pi$. $\beta_o(h)$ is a slowly varying function of kh and $\beta_o(h)$ increases for increasing kh. Thus by considering the form for $\beta_o(h)$ given by (5.13a) we have that for $0 \leq kh \leq \pi$

$$\frac{1}{12} \leq \beta_o(h) \leq \frac{\pi^2 - 4}{4\pi^2} < 0.15 . \qquad (5.34)$$

Hence

$$\left|y(x_{n+2}) - y_{n+2}\right| \leq \frac{\left(\frac{h^6k^2M_4}{48} + \frac{h^7k^2M_5}{48} + \frac{h^6M_6}{10}\right)}{\left|1 - 0.15h^2K\right|}$$

and for all $h \leq h_o < \sqrt{\frac{20}{3K}}$ we have

$$\left|y(x_{n+2}) - y_{n+2}\right| \leq h^6[k^2N_4 + N_6] \qquad (5.35)$$

where

$$N_4 = \frac{(M_4 + h_oM_5)}{48(1-0.15h_o^2K)} , \qquad N_6 = \frac{M6}{10(1-0.15h_o^2K)}$$

§5.4 Absolute stability of the method

This is an area which has not been considered by Raptis and Allison. We shall consider the application of the boundary locus

method to determine the interval of absolute stability. The locus of the boundary of the region of absolute stability is given by

$$\bar{h}(\theta) = \frac{\rho(\exp(i\theta))}{\sigma(\exp(i\theta))}$$

where

$$\rho(r) = r^2-2r+1, \quad \sigma(r) = \beta_o(h)r^2 + [1-2\beta_o(h)]r + \beta_o(h).$$

Hence we have

$$\bar{h}(\theta) = \frac{-2[1-3\beta_o(h)] + 2[1-4\beta_o(h)]\cos\theta + 2\beta_o(h)\cos 2\theta}{[1-4\beta_o(h)+6\beta_o^2(h)]+4\beta_o(h)[1-2\beta_o(h)]\cos\theta+2\beta_o^2(h)\cos 2\theta} \qquad (5.36)$$

which implies that the boundary of the region is an interval of the real axis. If we equate to zero the first derivative of $\bar{h}(\theta)$ with respect to θ we have

$$\sin\theta\left\{4\beta_o^2(h)\cos^2\theta+4\beta_o(h)[1-2\beta_o(h)]\cos\theta+[1-2\beta_o(h)]^2\right\} = 0$$

from which it follows that

$$\sin\theta = 0 \qquad (5.37a)$$

or

$$\cos\theta - \left[\frac{2\beta_o(h) - 1}{2\beta_o(h)}\right] = 0 . \qquad (5.37b)$$

The end points of the interval of absolute stability are determined by the values of θ which satisfy (5.37). If equation (5.37b) is to hold then we have

$$\cos\theta = \frac{2\beta_o(h) - 1}{2\beta_o(h)}$$

with

$$-2\beta_o(h) \leq 2\beta_o(h) - 1 \leq 2\beta_o(h)$$

since $\beta_o(h)$ is positive. It follows that (5.37b) will hold only if $\beta_o(h) \geq \frac{1}{4}$. Hence for $\beta_o(h) < \frac{1}{4}$ the endpoints of the stability interval are determined by the roots of the equation (5.37a), that is by $\theta = 0$ and $\theta = \pi$ for which

$$\bar{h}(0) = 0$$

$$\bar{h}(\pi) = \frac{-4}{1-4\beta_o(h)} \quad .$$

For $\beta_o(h) \geq \frac{1}{4}$,

$$\bar{h}(\theta) \longrightarrow \frac{-2(1-\cos\theta)}{2\beta_o(h)\cos\theta + 1-2\beta_o(h)}$$

as

$$\cos\theta \longrightarrow \frac{2\beta_o(h)-1}{2\beta_o(h)} \quad .$$

That is,

$$\bar{h}(\theta) \longrightarrow -\infty .$$

Hence for $\beta_o(h) < \frac{1}{4}$ the absolute stability region is $\left[\frac{-4}{1-4\beta_o(h)}, 0\right]$ and for $\beta_o(h) \geq \frac{1}{4}$ the region is $[-\infty, 0]$. The smallest stability interval occurs when $\beta_o(h)$ has its smallest value, that is when $\beta_o(h) = \frac{1}{12}$; the interval of absolute stability is then given by $[-6,0]$ which is the stability region for Numerov's method. The larger the value of $\beta_o(h)$ the larger is the corresponding interval of absolute stability.

§ 5.5 The cumulative error

The exact solution satisfies

$$y(x_n+2h)-2y(x_n+h)+y(x_n)=h^2\left\{\beta_o(h)y''(x_n+2h)+[1-2\beta_o(h)]y''(x_n+h)+\beta_o(h)y''(x_n)\right\}$$
$$+\mathcal{L}[y(x_{n+1});h] \quad .$$

If we ignore rounding errors the numerical values y_n satisfy

$$y_{n+2} - 2y_{n+1} + y_n = h^2\left\{\beta_o(h)f_{n+2} + [1-2\beta_o(h)]f_{n+1} + \beta_o(h)f_n\right\} .$$

Now if we assume

$$\left|y(x_\mu) - y_\mu\right| \leq h\xi, \quad \mu = 0, 1$$

with

$$h^2\beta_o(h) < \frac{1}{K} \quad \text{and} \quad 0 \leq kh \leq \pi$$

we obtain the following bound

$$|y(x_n)-y_n| = |e_n| \leq \Gamma^* [8(x_n-a^*)\S + \frac{(x_n-a^*)^2}{2}(k^2N_4+N_6)h^4]\exp[(x_n-a^*)^2\Gamma^*KB] \tag{5.38}$$

for $a \leq x_n \leq b$, where K, a^* and Γ^* are defined in §4.2.3. B is given by

$$B = |\beta_o(h)| + |1-2\beta_o(h)| + |\beta_o(h)|$$

which for $0 \leq kh \leq \pi$ is just 1 since $\frac{1}{12} \leq \beta_o(h) < 0.15$.

Thus the cumulative error is of order h^4.

§ 5.6 The method of Raptis and Allison with variable stepsize

If the solution is advanced one step from x_{n+1} to x_{n+2} by the method of Raptis and Allison estimates of the local error per unit step and per unit step per unit step are given by

$$-\frac{h^5}{240}(k^2 y_{n+1}^{lv} + y_{n+1}^{vl}) \tag{5.39}$$

and

$$-\frac{h^4}{240}(k^2 y_{n+1}^{lv} + y_{n+1}^{vl}) \tag{5.40}$$

respectively. We shall discuss in § 5.7 automatic implementations of the Raptis and Allison method, namely EXPFIT1 and EXPFIT2 which use (5.39) and (5.40) respectively as a means of controlling the global error of the method.

It was noted in § 5.2 that

$$k^2 y^{lv}(x) + y^{vl}(x)$$

may be expressed as the fourth derivative of

$$k^2 y(x) + y''(x) \tag{5.41}$$

with respect to x. We therefore need to calculate the solution at a minimum of five mesh points before an estimate of the truncation error can be obtained from a fourth divided difference table of the quantity in (5.41). If a decrease in steplength is required we shall need to have available a fifth divided difference table of the solution y(x) so that values from this table may be used in the inter-

polation procedure which is a necessary feature of steplength reduction in the Raptis and Allison method just as it is in Numerov's method (see §4.3).

We shall discuss in §5.7.2 the steplength strategy which has been adopted in programs EXPFIT1 and EXPFIT2; the strategy is precisely the same as that adopted in NUMEROV1 and NUMEROV2 respectively, the only difference being that the updated values of FORTRAN variables needed for the step after a change in steplength must be modified to take account of the h dependent coefficient β.

§5.7 Implementation of the Raptis and Allison method with automatic error control

We shall discuss programs EXPFIT1 and EXPFIT2 in this section and a listing of EXPFIT1 is provided in Appendix 2; the test program solves the single channel radial Schrödinger equation given by (5.1) for scattering of an electron by the static potential of atomic hydrogen.

The method of Raptis and Allison applied to the problem in (5.1) is certainly a valid approach for obtaining the solution in the classical region; in the non-classical region however the behaviour of the solution is difficult to predict. Thus in practice we shall use the Numerov method out to the classical turning point and thereafter use the method of Raptis and Allison whereupon the coefficients of the method will vary with the steplength h. Raptis and Allison state that the local truncation error can be used to control the interval size but there is no evidence from their study to show that this has been put into practice.* In particular they have studied the method for solving equation (5.1) with V(x) given by

$$V(x) = 500\left(\frac{1}{x^{12}} - \frac{1}{x^{6}}\right) \qquad (5.42)$$

which is a Lennard-Jones potential; they claim that a comparison with

* See footnote on page 20.

Numerov's method to solve the above problem using the same starting conditions indicates a rapid increase with the Raptis and Allison method over Numerov's method in the optimal interval size allowed from truncation error considerations. We shall give a comparison of programs NUMEROV1 and EXPFIT1 in § 5.7.5 when applied to solving (5.1) with V(x) given by (5.42).

§ 5.7.1 Programming the Raptis and Allison algorithm

The method of Raptis and Allison applied to equation (5.1) is expressed as

$$[1-h^2\beta_o(h)\lambda_{k+1}]\ y_{k+1} = 2[1-h^2\beta_o(h)\lambda_k]y_k-[1-h^2\beta_o(h)\lambda_{k-1}]y_{k-1}+h^2\lambda_k y_k \tag{5.43}$$

where

$$\lambda_r = \frac{\ell(\ell+1)}{x_r^2} - E + V(x_r).$$

The XX and F arrays are used just as in NUMEROV1 and NUMEROV2 to store the values of the mesh points and the corresponding calculated solutions. Since we initially use the method of Numerov up to the classical turning point, at which point we enter the classical region, the calculation of the initial step will be precisely as described in §4.4.3; values of F(1) and F(2) are calculated to sufficient accuracy and passed to the routine RAPAL in the argument list.

We introduce the quantity BO; BO represents the value of $\beta_o(h)$ and is $\frac{1}{12}$ up to the classical turning point and thereafter its value will be given by equation (5.13a). For small values of kh a better representation of (5.13) is

$$\beta_o(h) = \beta_2(h) = \frac{k^2h^2 - 4\sin^2\left(\frac{1}{2}kh\right)}{4k^2h^2\sin^2\left(\frac{1}{2}kh\right)} \tag{5.44a}$$

$$\beta_1(h) = \frac{4\sin^2\left(\frac{1}{2}kh\right) - k^2h^2\cos kh}{2k^2h^2\sin^2\left(\frac{1}{2}kh\right)} \qquad (5.44b)$$

and beyond the classical turning point we calculate $\beta_o(h)$ using equation (5.44a). Raptis and Allison have noted that the only modification required to computer codes that implement the Numerov algorithm is the replacement of the coefficients by subroutines which evaluate (5.44). The calculation of $\beta_o(h)$ from (5.44a) involves only a few lines of programming in RAPAL and it is in fact unnecessary to calculate $\beta_1(h)$ from (5.44b) since we can make use of the simple relation

$$2\beta_o(h) + \beta_1(h) = 1 .$$

We introduce the quantities

$$Y = (1 - h^2 BO\,\lambda_{k+1})\, F(K+1),$$

$$YK = (1 - h^2 BO\lambda_k)\, F(K),$$

$$YPREK = (1-h^2 BO\lambda_{k-1})\, F(K-1),$$

$$H2VF = h^2\lambda_k\, F(K),\ V = \lambda_k,\ H2 = h^2 .$$

If the steplength h has been used as far as x_k then in order to advance the integration one step to x_{k+1} we must update the values of YK and YPREK for use in the next step. We do this by setting

$$H2VF = H2* V* F(K) \qquad (5.45a)$$

$$YPREK = ((BO - BO1) *F(K-1) + BO1*YK)/BO \qquad (5.45b)$$

$$YK = ((BO - BO1)*F(K) + BO1*Y)/BO \qquad (5.45c)$$

where

$$BO1 = BO = \frac{1}{12}$$

in the Numerov stage of RAPAL; for the change-over step from Numerov to the Raptis and Allison method BO = $\frac{1}{12}$ and BO1 is given by (5.44a). Thereafter BO and BO1 are equal for as long as the steplength remains constant. We shall discuss the case of steplength increase and decrease in § 5.7.2 .

The following FORTRAN statements calculate the solution at x_{k+1}

```
Y      =      2.0* YK - YPREK + H2VF          (5.46)

F(K+1)=       Y/(1.0 - BO* H2 * V)  .          (5.47)
```

An estimate of the local error per unit step during the Numerov stage of RAPAL is

$$\text{TRERR} = -\frac{h^5}{240}\,\text{DD} \qquad (5.48)$$

where DD is an estimate of y^{vl}_{k+1}. During the Raptis and Allison stage of RAPAL the estimate of the local error per unit step is given by (5.48) where now DD is an estimate of the fourth derivative of $(k^2 y_{k+1} + y''_{k+1})$.

§ 5.7.2 Steplength strategy

The parameter EPS which is specified by the user is transferred to RAPAL by the common block EKLL1. EPS which is used in EXPFIT1 and EXPFIT2 is the largest allowed local error per unit step and per unit step per unit step respectively. The strategy used in EXPFIT1 and EXPFIT2 corresponds exactly to that used in NUMEROV1 and NUMEROV2 respectively. However during the Raptis and Allison stage of RAPAL the β coefficients are now dependent on the steplength h and we shall therefore need to modify the updating of the variables YPREK and YK to account for this.

We consider first the change-over from Numerov's method to the method of Raptis and Allison. Suppose we have reached the mesh point XX(K+1) using Numerov's method with a constant steplength h and at this point the classical region is entered. Then we must use the Raptis and Allison method to calculate the solution at XX(K+2) and at all subsequent mesh points. The current values of YPREK, YK and Y are

$$\text{YPREK} = (1-h^2\beta_o(h)\lambda_{k-1})\ F\ (K-1)$$

$$\text{YK} = (1-h^2\beta_o(h)\lambda_k)\ F\ (K)$$

$$\text{Y} = (1-h^2\beta_o(h)\lambda_{k+1})\ F\ (K+1)$$

where $\beta_o(h) = \frac{1}{12}$. If we define

$$Y_r = (1-h^2\beta_o'(h)\lambda_r)\ F\ (r)$$

where $\beta_o'(h)$ is given by equation (5.44a), we shall require the values of Y_K and Y_{K+1} which are the new values of YPREK and YK respectively for use in the next step. We have

$$Y_K = (1-h^2\beta_o'(h)\lambda_k)\ F\ (K)$$

and thus

$$Y_K = \frac{(\beta_o(h)-\beta_o'(h))F\ (K) + \beta_o'(h)YK}{\beta_o(h)} \qquad (5.49a)$$

Similarly

$$Y_{K+1} = \frac{(\beta_o(h) - \beta_o'(h))F\ (K+1) + \beta_o'(h)Y}{\beta_o(h)} \qquad (5.49b)$$

The FORTRAN implementation of (5.49) is given by (5.45b, c) with K replaced by K+1.

If at the mesh point x_{k+1} a decrease in the steplength h is required then the following FORTRAN statements update YPREK and YK:

```
YPREK  =  FPREK1 * (1.0 - BO*H2*V)
YK  =  F(K)*(1.0-BO*H2*V)
```

where FPREK1 is the interpolated solution at x_k-rh where rh is the new steplength ($r < 1$). BO is $\frac{1}{12}$ during Numerov's method and is given by equation (5.44a) when the Raptis and Allison method is used.

Suppose that after reaching the mesh point x_{k+1} with a constant steplength h an increase in steplength is required; if the steplength is doubled for the next step we shall require values of the solution at the mesh points XX(K-1) and XX(K+1) calculated with a steplength 2h. We redefine

$$Y_r = (1-(2h)^2 \beta_o(2h)\lambda_r)\, F(r) .$$

Now YPREK is given by

$$YPREK = (1-h^2\beta_o(h)\lambda_{k-1})\, F(K-1)$$

and

$$Y_{K-1} = (1-4h^2\beta_o(2h)\lambda_{k-1})\, F(K-1) .$$

Thus we have

$$Y_{K-1} = \frac{4\beta_o(2h)YPREK + (\beta_o(h) - 4\beta_o(2h))F(K-1)}{\beta_o(h)} .$$

Similarly

$$Y_{K+1} = \frac{4\beta_o(2h)Y + (\beta_o(h)-4\beta_o(2h))F(K+1)}{\beta_o(h)} .$$

Thus we use the following implementation to update YPREK and YK:

```
YPREK = (C2*BO1*YPREK + (BO - C2*BO1)*F(K-1))/BO
YK = (C2*BO1*Y + (BO - C2*BO1)*F(K+1))/BO
```

where C2 = C*C and C(=2) is the factor by which the steplength is increased. BO1 and BO represent $\beta_o(2h)$ and $\beta_o(h)$ respectively; the next step uses a steplength 2h so we write

```
BO = BO1 .
```

§ 5.7.3 Description of programs EXPFIT1, EXPFIT2

Figure 5 below shows the relationship between the seven routines of programs EXPFIT1 and EXPFIT2 which solve the radial Schrödinger equation (5.1); EXPFIT1 uses a local error per unit step criterion and EXPFIT 2 a local error per unit step per unit step criterion in the automatic integration of (5.1).

We shall concentrate on a description of EXPFIT1 since EXPFIT2 is obtained simply by replacing six FORTRAN statements which appear in EXPFIT1; we shall discuss the necessary modification in part (ii) of this Section. Double precision is used for all real variables.

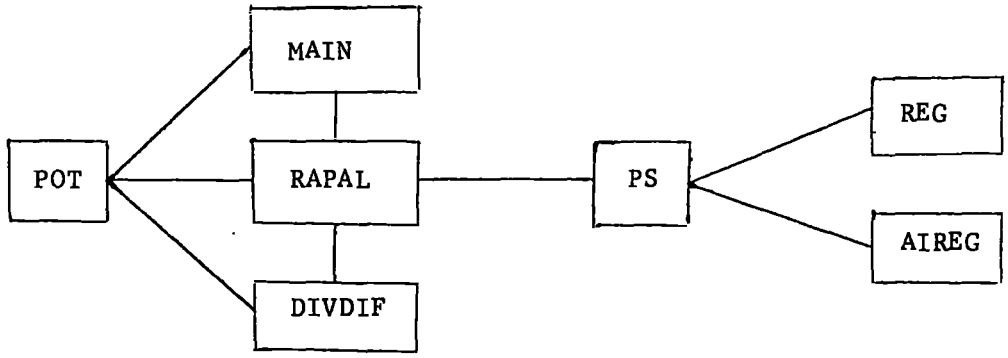

FIGURE 5

(i) The main routine

Since the initial stage of the Raptis and Allison algorithm uses Numerov's method the calculation of the initial conditions y_0 and y_1 is precisely as described in §4.4.3. The main routine in EXPFIT1 is thus identical to that in NUMEROV1 (see §4.4.4 part (i)) with the exception that the common block EKLL1 is used to transfer parameters to the subroutine DIVDIF as well as to RAPAL, PS and POT. The main routine now calls the subroutine RAPAL in place of NUMOV to perform the automatic integration of (5.1).

(ii) Subroutine RAPAL

Subroutine RAPAL uses the method of Raptis and Allison to solve

$$y'' = f(x,y)$$

using a local error per unit step criterion. The parameters which must be supplied to RAPAL as input data are:

H : the initial steplength,

XO, X, YO, Y1 : the starting point along with the next mesh point X = XO+H and the values of the solution at these points.

XF : The calculation terminates somewhere between XF-H and XF unless earlier termination has occurred because of convergence of the phase shift to the required accuracy.

Lines (216-226)* calculate the values of YPREK and YK for use

* The line numbers quoted are those appearing on the left hand side of the listing of EXPFIT1 in Appendix 2.

in the initial step of Numerov's method and lines 233 - 305 implement the Raptis and Allison algorithm to advance the calculation one step of length H, evaluate the truncation error estimate and decide whether or not to alter the steplength and/or the β coefficients in the numerical method of solution. The initial stage of the calculation uses Numerov's method with constant β coefficients up to the classical turning point beyond which the β coefficients vary with the interval h in the Raptis and Allison method.

The decisions to change over from Numerov's method to the method of Raptis and Allison and to calculate a phase shift are taken within the function subprogram POT. Both these decisions are characterised by the parameters IPS which is assigned the value 1 in POT when the classical region is reached. When this condition is encountered lines 246-250 of RAPAL calculate $\beta_0(h)$ for subsequent use in the Raptis and Allison method and at line 281 the subroutine PS is called to calculate the phase shift; the current value of x and the calculated phase shift are then printed. The IF condition at line 240 ensures that the first seven steps are taken with the Numerov method regardless of the value of IPS.

If at line 271 it is decided that the truncation error is too large control is transferred to lines 324-360 for the steplength decrease and necessary updating of YPREK and YK and thence to the beginning of the main loop at line 233; if the initial H provided by the main routine to RAPAL proves too large then lines 160-170 recalculate Y1 with the new steplength.

If the steplength is to be increased line 290 transfers control to lines 364-384 for the steplength increase and necessary updating of YPREK and YK; line 384 then transfers control to the beginning of the main loop.

Exit from the Raptis and Allison loop occurs (see line 283) if the subroutine PS indicates, by setting JCONV = 1, that the phase shift has been calculated to the required accuracy; otherwise termination occurs when it is noted (at line 236) that the next step would take the calculation beyond XF. In either case, as in DEVOG in program RADISH, the user is informed of the reason for termination and information on the number of steps carried out and the number of increases and decreases of steplength is also printed.

The corresponding subroutine required for use in EXPFIT2 may be obtained by replacing lines 169, 263, 289 and 325 of RAPAL by

```
IF (DTERM. GT. 0.5DO*H*H*EPS) GO TO 22,
TRERR = -H**4*DD/2.4D2,
DTRERR = DTRERR*C**4
```

and

```
C1 = (0.5DO*TOL/DTRERR)**0.25DO .
```

In addition lines 90 and 94 of the main routine must be replaced by

```
IF(DTERM.LT.H*H*EPS) GO TO 17,
```

and

```
16 C1 = (0.5DO*H*H*EPS/DTERM1)**0.25DO .
```

Similarly if the above FORTRAN statements replace the corresponding statements in NUMEROV1 we obtain program NUMEROV2.

The test program EXPFIT1 is based on the Schrödinger equation in the precise form of (3.2) for positive energy E.

(iii) Subroutine DIVDIF

The parameters which must be supplied for calls to subroutine DIVDIF are DD, K, J, JD, IPS and JB. The first four parameters are described in §4.4.4 (part (iii)). In addition the value of DD serves to distinguish between calls made to DIVDIF for the purpose

of calculating an estimate of the derivative in the truncation error estimate (see § 5.2) and calls made to extract the values needed for the interpolation process in the case of a decrease in steplength during the Raptis and Allison method; DD is set to zero immediately after the values for interpolation have been found. The parameter IPS has been discussed in part (ii) of this Section. JB is set to zero during the Numerov stage of the algorithm and is set to 1 when the classical region is reached (when IPS = 1). On the next step which uses the method of Raptis and Allison the value of JB is 2 and this indicates to DIVDIF that the fourth divided difference table of (k^2y+y'') must be set up; thereafter JB has the value 3.

The array D1(4,4) is introduced to store relevant values in the case of a steplength decrease during the Raptis and Allison method. A detailed description of DIVDIF is provided by comment cards in the listing of EXPFIT1 (see Appendix 2).

(iv) Subroutine PS and functions POT, REG, AIREG

The above have been discussed in § 4.4 (parts (iv)-(vi)) and are used in precisely the same form here, the only exception being that POT initialises the calculation of the phase shift by setting IPS = 1 when the classical region is reached. (The calculation of the phase shift could be initiated, if desired, in the so called asymptotic region, simply by introducing another parameter, IPS2 say, in calls to the function POT; the calculation of the β coefficients would be characterised by the value assigned to IPS). POT is called by DIVDIF, RAPAL and the main routine of EXPFIT1.

§ 5.7.4 Test runs

Programs EXPFIT1 and EXPFIT2 have been tested for the same set of problems as tested by programs RADISH, NUMEROV1 and NUMEROV2 and details of the results for these test problems will appear in Chapter 7.

We consider here the solutions of the following problems over

the range $[x_o, 20]$.

(i) $y'' = \left(\frac{2}{x^2} - k^2\right) y$; $y_o = x_o j_1(kx_o)$, $y_1 = x_1 j_1(kx_1)$, $x_o = 0.01$

for k = 0.1, 0.2, 0.5, 1.0, 2.0, 5.0 with EPS = 10^{-n}, n=3, 4, 6, 8.

(ii) $y'' = \left[\frac{\ell(\ell+1)}{x^2} - k^2 + 500\left(\frac{1}{x^{12}} - \frac{1}{x^6}\right)\right] y$; $y_o=0$, $y_1=y(x_1)$, $x_1=x_o+h$, $x_o=0.7$, h=0.01, for ℓ=0, 2, 4 with k=3.0 and ℓ=0, 5, 10 with k=10.0.

Problem (i) corresponds to problem (ii) of § 3.6 for which the exact solution is $y = xj_1(kx)$; exact starting values for x_o, x_1, y_o and y_1 are used and the initial value of the steplength is chosen to be

$$\min\left[\left(\frac{240\ \text{EPS}}{k^6|y_o|}\right)^{\frac{1}{5}}, \left(\frac{20 - x_o}{7}\right)\right]$$

in the case of EXPFIT1 and

$$\min\left[\left(\frac{240\ \text{EPS}}{k^6|y_o|}\right)^{\frac{1}{4}}, \left(\frac{20 - x_o}{7}\right)\right]$$

in the case of EXPFIT2. The results for this problem using EXPFIT1 and EXPFIT2 are tabulated in Table 8 and Table 9 respectively of § 5.7.5. We have removed the restriction in both programs that the first seven steps be performed with the Numerov method; since the potential function V(x) is zero for this problem the Raptis and Allison stage is entered when

$$\frac{2}{x^2} - k^2 < 0$$

is satisfied and for small values of k with a large (> 1) initial steplength (acceptable for the Numerov stage of the algorithm) the Raptis and Allison stage may be entered fairly quickly, perhaps even at the third step.

Problem (ii) corresponds to the test problem considered by Raptis and Allison (1977, preprint) where V(x) corresponds to a Lennard-Jones potential. They specify $x_o = 0.7$ and an initial steplength of 0.01 and a table of results is provided which shows the phase shift obtained accurate to three decimal places for the various values of k and ℓ using the Numerov method and the method

of Raptis and Allison. They also list the 'final interval' and the number of steps used by each method. It is not clearly stated what starting conditions were used but we have chosen to provide $y_o = 0$ and $y_1 = 10^{-8}$, with an initial steplength of 0.01 and initial mesh point $x_o = 0.7$ as in Raptis and Allison. Table 10 shows the results obtained when NUMEROV1 and EXPFIT1 are used to solve this problem; we show for each value of k and ℓ which is tested the corresponding phase shift δ to three decimal places along with h_{max}, the largest interval used and N, the number of steps required to solve the problem using Numerov's method and the method of Raptis and Allison. The integration is terminated when the phase shift has converged to the required accuracy; the integration is performed using EPS = 10^{-6} and PSIG = 10^{-4}.

§ 5.7.5 Test results

The relevant statistics which may be extracted from Tables 8 and 9 are respectively

$$a_{max} = 47.3, \quad a_p \simeq 70\%, \quad a_{p5} \simeq 26\%$$

and

$$a_{max} = 47.3, \quad a_p \simeq 52\%, \quad a_{p5} \simeq 17\% .$$

We see that EXPFIT2 is more effective in controlling the global error than is EXPFIT1; however the value of a_{max} far exceeds that in Tables 6 and 7 where the method of Numerov is used and a comparison of the corresponding values of a_p and a_{p5} also seems to favour the method of Numerov with respect to control of the global error. Against this if we compare Tables 7 and 9 we see a substantial decrease in the number of function evaluations as used in EXPFIT2 compared with NUMEROV2 sometimes by a factor which is greater than three. Similarly a comparison of Tables 6 and 8 favours EXPFIT1 over NUMEROV1 in terms of the number of function evaluations and the factor by

which this number is reduced in EXPFIT1 is sometimes greater than two. In all but one case (when EPS = 10^{-8}, k = 0.1) the number of function evaluations used in the programs incorporating the Raptis and Allison method is consistently less than that in the programs incorporating Numerov's method. We maintain that those cases where Numerov's method appears to give a better control of the global error than the method of Raptis and Allison are a consequence of NUMEROV1 and NUMEROV2 accepting a smaller initial value of the steplength; also for small values of k and large values of the corresponding initial steplength accepted, the Raptis and Allison stage may be entered fairly quickly, as noted in § 5.7.4, with the result that the steplength increases with higher frequency than it would otherwise do in the Numerov stage of the algorithm. In order to fully appreciate the advantages to be gained in using the method of Raptis and Allison a more useful comparison might be to provide each of the methods with the same initial value of the steplength which is acceptable to both methods and then keep this steplength fixed throughout the range of integration. A comparison of the actual errors then incurred clearly shows that the method of Raptis and Allison is superior to that of Numerov; the Raptis and Allison method has the effect of damping out the error in the later stages of the calculation so that the largest error occurs in the early stage of the integration, typically in $[x_o, 5]$.

In particular we consider the case corresponding to EPS = 10^{-3}, k = 2.0. We have not included in Tables 8 and 9 results for this case; unfortunately the initial value of the steplength accepted by EXPFIT1 and EXPFIT2 (2.239 and 2.739 respectively) on the basis that the appropriate error criterion is satisfied is totally misleading since a study of the actual errors incurred using the large

initial steplengthsshows them to be of the same order of magnitude as the computed solution and in some cases the order of magnitude exceeds that of the computed solution. We chose to provide the same initial value of the steplength h to subroutines NUMOV and RAPAL and arranged for h to remain fixed throughout the range of integration. Then with h = 0.25 for k = 2.0, EPS = 10^{-3} we obtained the following scaled maximum errors for d = 5, 10, 20 respectively in NUMOV:

8.31(-5), 1.03(-4), 1.20(-4)

and

2.99(-5), 1.53(-5), 7.87(-6)

in RAPAL. We see from these results the increased effectiveness of the method of Raptis and Allison over Numerov's method for solving the problem at hand for a fixed steplength.

Table 10 shows the results obtained for problem (ii). It is immediately apparent that the method of Raptis and Allison is able to use larger values of the steplength thus resulting in a decrease in the number of function evaluations required. In addition the method of Raptis and Allison uses considerably less time (sometimes by a factor which is near a half) than Numerov to perform the numerical integration and extraction of a phase shift. We have not compared actual errors which result from each method since it is observed that the equation given in problem (ii) is essentially unstable; this is reflected by a small change in the starting conditions giving rise to a substantial change in the computed solution.

TABLE 8

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	3.82(-5)	2.856	6
	0.5	4.37(-4)	2.36(-4)	2.88(-4)	2.856	6
	1.0	3.04(-4)	3.34(-4)	1.95(-4)	2.856	31
	2.0	*	*	*		
	5.0	2.65(-3)	1.33(-3)	6.93(-4)	0.621	58
10^{-4}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	2.19(-5)	2.856	6
	0.5	5.07(-4)	5.28(-4)	3.77(-4)	2.856	17
	1.0	2.46(-4)	1.43(-4)	7.30(-5)	2.856	35
	2.0	1.24(-4)	6.31(-5)	3.22(-5)	1.413	59
	5.0	4.95(-5)	2.70(-5)	1.48(-5)	0.392	97
10^{-6}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	4.24(-5)	2.856	8
	0.5	2.82(-5)	1.44(-5)	7.32(-6)	2.856	35
	1.0	2.84(-5)	1.40(-5)	7.19(-6	1.484	63
	2.0	6.18(-6)	3.15(-6)	1.66(-6)	0.562	112
	5.0	2.73(-6)	1.45(-6)	7.57(-7)	0.156	204
10^{-8}	0.1	2.39(-8)	2.02(-8)	1.25(-8)	2.856	43
	0.2	4.01(-8)	1.96(-8)	1.26(-8)	2.856	58
	0.5	4.91(-8)	2.51(-8)	2.21(-8)	1.559	96
	1.0	5.11(-8)	2.52(-8)	1.27(-8)	0.591	179
	2.0	4.37(-8)	2.38(-8)	2.24(-8)	0.224	274
	5.0	1.77(-7)	1.15(-8)	6.62(-9)	0.062	525

TABLE 9

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	3.82(-5)	2.856	6
	0.5	4.37(-4)	3.36(-4)	2.88(-4)	2.856	6
	1.0	3.43(-3)	1.72(-3)	9.37(-4)	2.856	22
	2.0	*	*	*		
	5.0	1.06(-4)	5.96(-5)	3.59(-5)	0.551	80
10^{-4}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	2.19(-5)	2.856	6
	0.5	4.37(-4)	3.36(-4)	2.88(-4)	2.856	6
	1.0	2.30(-4)	1.22(-4)	6.45(-5)	2.856	36
	2.0	6.46(-5)	3.29(-5)	1.74(-5)	1.540	62
	5.0	1.15(-5)	6.55(-6)	3.52(-6)	0.310	136
10^{-6}	0.1	3.17(-6)	2.87(-6)	1.99(-6)	2.856	6
	0.2	4.73(-5)	3.59(-5)	2.19(-5)	2.856	6
	0.5	2.46(-5)	1.18(-5)	5.97(-6)	2.856	37
	1.0	3.85(-7)	3.48(-7)	2.01(-7)	1.638	112
	2.0	1.56(-7)	8.23(-8)	5.04(-8)	0.487	177
	5.0	3.17(-8)	2.40(-7)	1.34(-7)	0.098	360
10^{-8}	0.1	2.01(-8)	1.56(-8)	9.53(-9)	2.856	45
	0.2	1.52(-8)	8.60(-9)	1.70(-7)	2.856	67
	0.5	7.26(-9)	1.36(-7)	1.45(-7)	1.742	141
	1.0	4.10(-9)	1.78(-8)	1.06(-8)	0.518	262
	2.0	1.50(-8)	1.04(-8)	6.15(-9)	0.154	479
	5.0	4.70(-9)	3.19(-9)	1.21(-8)	0.031	1024

TABLE 10

k	ℓ	NUMEROV δ	NUMEROV h_{max}	NUMEROV N	RAPTIS AND ALLISON δ	RAPTIS AND ALLISON h_{max}	RAPTIS AND ALLISON N
3.0	0	-0.590	0.045	609	-0.590*	0.364	260
3.0	2	-1.288	0.046	593	-1.288	0.091	276
3.0	4	0.144	0.046	615	0.144	0.091	344
10.0	0	-0.431	0.012	1446	-0.431*	0.122	343
10.0	5	-0.298	0.012	1454	-0.299	0.042	444
10.0	10	0.378	0.012	1417	0.378	0.023	698

* The values of the phase shift presented by Raptis and Allison (1977) for $(k, \ell) = (3.0, 0)$ and $(10.0, 0)$ should be negated.

CHAPTER 6

A variable-step variable-order Adams method

Introduction

The N.A.G. routine D02AHF (N.A.G. Library, 1974) integrates a system of first order ordinary differential equations over a specified range, using a variable-step variable-order Adams method (Shampine/Gordon, 1975). The routine obtains an estimate of the local error at each step and varies the order and steplength automatically to keep this estimate below a prescribed tolerance level. We shall discuss in § 6.1 the particular Adams Bashforth-Adams Moulton method used which lends itself to an efficient implementation in routine D02AHF. A detailed discussion of all the various features incorporated in D02AHF would prove rather lengthy and we shall describe only the basic features such as the initial stage of the integration, the error estimation and the strategies adopted for changing the steplength and the order. For a more detailed description of the algorithm employed in D02AHF the reader is referred to Siemieniuch (1972).

Our reason for using D02AHF is twofold; first as a means of solving

$$y'' = f(x, y) \qquad (6.1)$$

where

$$f(x, y) = \left[\frac{\ell(\ell+1)}{x^2} - E + V(x)\right] y(x)$$

as a method in its own right in program NAGMOD, and second to incorporate its use in programs RADISH, NUMEROV1, NUMEROV2, EXPFIT1 and EXPFIT2 to generate high accuracy parallel solutions to those calculated by the methods of de Vogelaere, Numerov and Raptis and Allison. Program NAGMOD uses a modified version of the routine D02AHF to compute the solution of (6.1) where of course (6.1) is

treated as the pair of coupled first order differential equations

$$y_1' = y_2$$
$$y_2' = f(x, y_1) \qquad (6.2)$$

where we have substituted $y_1 = y$ in (6.1). Details of the modifications required to D02AHF specific to our needs appear in § 6.2.

In order to study the actual errors which accumulate in routines DEVOG, NUMOV and RAPAL during the course of integration in cases where the exact solution of the problem is not known we use D02AHF to provide solutions to high accuracy at the same points which have been chosen automatically in these routines; thus we may regard the N.A.G. solution as an 'exact' solution of the problem.

§ 6.1 Description of D02AHF

The routine D02AHF uses the kth order Adams Bashforth formula combined with the (k+1)st order Adams Moulton formula in the PECE mode to calculate the solution of the first order equation

$$y' = f(x, y), \quad y(x_o) = y_o \qquad (6.3)$$

or systems of such equations. The solution of (6.3) at the mesh point x_{n+1} may be written as

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))dx. \qquad (6.4)$$

The Adams Bashforth formula of order k uses a polynomial $P_{k,n}(x)$ to interpolate the values $f_n, f_{n-1}, \text{---}, f_{n-k+1}$ where $f_i = f(x_i, y_i)$ and is given by

$$p_{n+1} = y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P_{k,n}(x)dx . \qquad (6.5)$$

This is an explicit formula for the so called predicted value p_{n+1} of the solution at x_{n+1}. The Adams Moulton formula of order k+1 uses the same values which are used in (6.5) along with p_{n+1} and is

given by

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P_{k+1,n}(x)dx \qquad (6.6)$$

where

$$P_{k+1,n}(x_{n+1-j}) = f_{n+1-j},\ j = 1, ---, k$$

$$P_{k+1,n}(x_{n+1}) = f^P_{n+1} = f(x_{n+1},P_{n+1}) . \qquad (6.7)$$

This is an implicit formula which yields a corrected value for the solution at x_{n+1}.

One representation of the interpolating polynomial $P_{k,n}(x)$ is the divided difference form:

$$P_{k,n}(x) = f[x_n]+(x-x_n)f[x_n,x_{n-1}]+---$$
$$\vdots$$
$$+(x-x_n)(x-x_{n-1})---(x-x_{n-k+2})f[x_n,x_{n-1},---,x_{n-k+1}] \qquad (6.8)$$

which in the case of evenly spaced mesh points reduces to the backward difference form:

$$P_{k,n}(x) = f_n+\frac{(x-x_n)}{h}\nabla f_n+---+\frac{(x-x_n)(x-x_{n-1})---(x-x_{n+2-k})}{(k-1)!\ h^{k-1}}\nabla^{k-1}f_n \qquad (6.9)$$

where h is the steplength. The routine D02AHF is based on the divided difference form (6.8) of the interpolating polynomial. Most steps in the integration are taken in groups of constant steplength and constant order and these stages will use (6.8) in the reduced form (6.9); the form (6.8) proves convenient particularly for error estimation when estimates at different orders are required.

§ 6.1.1 Efficient implementation of the Adams methods

An efficient implementation of the Adams Bashforth-Adams Moulton method described above has been fully discussed in Shampine/Gordon (1975, Chapter 5). Introducing the quantities

$$h_i = x_i - x_{i-1},$$

$$s = \frac{x - x_n}{h_{n+1}},$$

$$\psi_i(n+1) = h_{n+1} + h_n + \cdots + h_{n+2-i}, \quad i \geq 1$$

$$\alpha_i(n+1) = \frac{h_{n+1}}{\psi_i(n+1)}, \quad i \geq 1$$

$$\beta_1(n+1) = 1$$

$$\beta_i(n+1) = \frac{\psi_1(n+1)\psi_2(n+1)\cdots\psi_{i-1}(n+1)}{\psi_1(n)\psi_2(n)\cdots\psi_{i-1}(n)}, \quad i > 1$$

$$\phi_1(n) = f[x_n] = f_n,$$

$$\phi_i(n) = \psi_1(n)\psi_2(n)\cdots\psi_{i-1}(n) f[x_n, x_{n-1}, \cdots, x_{n-i+1}], \quad i > 1 \tag{6.10}$$

we see that a typical term of $P_{k,n}(x)$ is

$$(x-x_n)(x-x_{n-1})\cdots(x-x_{n-i+2}) f[x_n, x_{n-1}, \cdots, x_{n-i+1}] = c_{i,n}(s)\phi_i^*(n)$$

where

$$c_{i,n}(s) = \begin{cases} 1 & , i = 1 \\ \dfrac{sh_{n+1}}{\psi_1(n+1)} = s & , i = 2 \\ \left(\dfrac{sh_{n+1}}{\psi_1(n+1)}\right)\left(\dfrac{sh_{n+1}+\psi_1(n)}{\psi_2(n+1)}\right)\cdots\left(\dfrac{sh_{n+1}+\psi_{i-2}(n)}{\psi_{i-1}(n+1)}\right) & , i \geq 3 \end{cases} \tag{6.11}$$

and

$$\phi_i^*(n) = \beta_i(n+1)\phi_i(n). \tag{6.12}$$

Thus

$$P_{k,n}(x) = \sum_{i=1}^{k} c_{i,n}(s)\phi_i^*(n) \tag{6.13}$$

and

$$P_{n+1} = y_n + \int_{x_n}^{x_{n+1}} \left(\sum_{i=1}^{k} c_{i,n}(s)\phi_i^*(n) \right) dx$$

$$= y_n + h_{n+1} \sum_{i=1}^{k} \phi_i^*(n) \int_0^1 c_{i,n}(s)\,ds. \tag{6.14}$$

Finally we obtain the following form for the predicted value p_{n+1}:

$$p_{n+1} = y_n + h_{n+1} \sum_{i=1}^{k} g_{i,1} \phi_i^*(n) \qquad (6.15)$$

where

$$g_{i,q} = \begin{cases} \frac{1}{q} & , i = 1 \\ \frac{1}{q(q+1)} & , i = 2 \\ g_{i-1,q} - \alpha_{i-1}(n+1) g_{i-1,q+1} & , i \geq 3 \end{cases} \qquad (6.16)$$

The corrected value is given by (6.6) and we may write

$$P_{k+1,n}(x) = P_{k,n}(x) + (x-x_n)(x-x_{n-1})\cdots(x-x_{n-k+1}) f^p[x_{n+1}, \cdots, x_{n-k+1}] \qquad (6.17)$$

where we have introduced a superscript p on the divided difference associated with $P_{k+1,n}(x)$. Equation (6.17) may be expressed as

$$P_{k+1,n}(x) = P_{k,n}(x) + c_{k+1,n}(s) \phi_{k+1}^p(n+1)$$

and integration of the above equation with respect to the variable s yields

$$y_{n+1} = y_n + h_{n+1} \int_0^1 P_{k,n}(x_n + s h_{n+1}) ds + h_{n+1} \int_0^1 c_{k+1,n}(s) \phi_{k+1}^p(n+1) ds$$

$$= p_{n+1} + h_{n+1} g_{k+1,1} \phi_{k+1}^p(n+1) \; . \qquad (6.18)$$

If a corrector of order k is used then equation (6.18) is replaced by (Shampine/Gordon, 1975, p.101)

$$y_{n+1} = p_{n+1} + h_{n+1} g_{k,1} \phi_{k+1}^p(n+1), \qquad (6.19)$$

which corresponds to taking one less term in (6.18). The Milne error estimate for the algorithm is then

$$h_{n+1} (g_{k+1,1} - g_{k,1}) \phi_{k+1}^p(n+1) \; . \qquad (6.20)$$

The method uses formulae of orders one up to thirteen and considers only halving and doubling of the steplength in cases where the steplength must be decreased and increased respectively. The $g_{k,1}$ coefficients are stored as fractional constants for k = 1, ---, 10 and

for k = 11, ---, 14 they are computed from the recurrence relation (6.16); notice that for a constant steplength $\alpha_i(n+1) = \frac{1}{i}$. The summation in (6.15) is performed in reverse order starting with the highest order divided difference in order to minimise machine round-off error and the current value of the independent variable x is computed by subtracting (ST-1) times the current value of the steplength H from the value of x at the end of the integration range, where ST is an integer variable and is the number of steps left to the end of the range for the current value of the steplength. The derivatives of the predicted and corrected values p_{n+1} and y_{n+1} are evaluated using an auxilliary routine AUX (N.A.G. Library, 1974) which is supplied by the user. Differences associated with $P_{k,n}(x)$ and $P_{k+1,n}(x)$ of (6.5) and (6.6) must also be evaluated; several of the computed differences are retained for the next step and the computations are organised so that they are as economical of storage as possible (see Shampine/Gordon, 1975, Chapter 5).

§6.1.2 Strategies for order and steplength selection

The estimate given in (6.20) of the local truncation error will in general overestimate the quantity since the Milne error estimate requires the predictor and corrector formulae to be of the same order. Suppose we have reached the mesh point x_n. The routine D02AHF accepts the predicted value at $x_{n+1} = x_n+h$ as sufficiently accurate if the absolute value of the error in each component of the predicted value of the solution is less than 0.1 times the error allowed in a mixed error test; if G(I), I=1,---,N is the real array which contains the error bounds specified by the user for each of the N components of the predicted solution vector YP(I) and the estimate of the local truncation error in each component is E(I) then a mixed error test

requires

$$\left|E(I)\right| \leq \quad G(I) \times (1+ \left|YP(I)\right|) \tag{6.21}$$

in order that the predicted value be acceptable. This error test is specified by an integer variable T which is set to 1 on initial entry to D02AHF. The values T = 2 and T = 3 give an absolute and relative error test respectively, that is

$$\left|E\ (I)\right| \leq \quad G(I) \tag{6.22}$$

and

$$\left|E\ (I)\right| \leq \quad G(I) \times \left|YP(I)\right| \ . \tag{6.23}$$

If the error criterion in (6.21) is not satisfied then the steplength is halved and the integer ST is doubled; in addition the difference table associated with $P_{k,n}(x)$ must be retabulated and an efficient algorithm (Krogh, 1973) is incorporated in D02AHF specifically for this purpose. The corresponding algorithm for doubling the steplength is also incorporated in D02AHF. However the use of these algorithms as described by Krogh (1973) can sometimes lead to a catastrophic growth in errors, particularly in the case of halving the steplength (see Hall and Watt, 1976, Chapter 6) and Krogh's modification of the halving algorithm is employed; when used on the next two steps after halving it has the effect of smoothing the differences.

The decisions to change the order and double the steplength are performed simultaneously after the corrected value of the solution has been obtained. The estimates of the local truncation error are weighted so as to simulate a doubling of the steplength; thus to increase the order from k to k+1 and the steplength from h to 2h the error in each component of the corrected value of the solution must have absolute value less than $(0.1)2^{-(k+2)}$ times the error allowed in a mixed error test. An increase in order from k to k+1 is only performed if at least k+1 steps at order k have been taken. The counter QO is used for this purpose. At the beginning of the integration

QO is set to -1; thus two steps must be acceptable before increasing the order from one to two in the initial stage of the algorithm. The integer variable Q denotes the current order. QO is increased by one after each successful step and for values of QO $\geqslant$ Q we can consider either increasing the order by one or doubling the steplength, or both. After suitable modification of the steplength and/or the order, QO is reset to zero. No such restriction is placed on decreases in steplength.

Using the corrected differences estimates of the local truncation error for the orders k-1, k and k+1 are computed where k is the current order of integration. The order in the next step is selected according to the minimum value of the weighted error estimates obtained at these three orders. If the steplength is doubled then the integer variable ST must be halved; however in cases where ST is odd immediately before an impending increase in steplength no increase in steplength is performed until the criterion for steplength doubling is again satisfied with ST even. The justification for this strategy is that an important feature of the algorithm is the automatic choice of an initial steplength such that any subsequent halving or doubling of this steplength over the specified range of integration will cause the integration to terminate exactly at X + HO where X is the initial mesh point supplied by the user and HO is the length of the integration range.

Finally we consider an additional test which is introduced to guard against the order becoming too high. With the retabulation of the difference table it may be the case that for large values of the order k some of the higher order differences have become unreliable; for a well behaved function the kth divided difference usually decreases in magnitude as the order k increases. Thus a sudden increase in the magnitude of the kth divided difference for a

particular value of k signals either that there is a discontinuity in the derivative of the solution or that round-off errors have accumulated to such a level that we can no longer consider the divided difference table to be reliable. Routine D02AHF reduces the order to k-1 if such a condition is encountered; if the calculated differences are still unacceptable the order is reduced by one successively until the differences become acceptable. We also note that as the order k increases the regions of absolute stability decrease fairly rapidly and a further advantage of the lower order methods is that they involve less computation.

§ 6.1.3 The initial stage

Routine D02AHF calculates EPS which is the smallest positive real number such that

$$1.0 + EPS > 1.0$$

and SMAX which is the largest integer such that SMAX and -SMAX can be represented on the computer. According as the integer variable T is 1, 2 or 3 on initial entry to D02AHF the error test given by (6.21), (6.22) or (6.23) is adopted for testing the error in each component of the predicted solution vector.

The array GMA(I), I-1,---,14 is set up and contains the values $g_{k,1}$, k=1,---,14. Routine D02AHF chooses its own initial value of the steplength to ensure that the range of integration HO is made up of ST intervals of length H where ST is an integral power of two and H is chosen to be as near as possible to the initial value of the steplength supplied by the user. Before estimating the first step of the algorithm the value of T is negated, Q is set to 1 and QO is set to -1. It is now possible to proceed with the calculation of the predicted and corrected values for the solution at the first and

subsequent mesh points.

An error indicator IFAIL is used by the routine; if the integration has been successfully completed the value IFAIL = 0 will be returned. The values IFAIL = 1, 2 indicate respectively that the number of steps required to complete the interval exceeds SMAX and that H=0 or H> HO on a second or subsequent entry to the routine.

§ 6.2 A modified version of D02AHF

For the purpose of using routine D02AHF to provide high accuracy parallel solutions to those calculated by the methods of de Vogelaere, Numerov and Raptis and Allison we use D02AHF in the form described in § 6.1. In Chapter 7 we shall compare the reliability and efficiency of the routines DEVOG, NUMOV and RAPAL in controlling the global error in the solution of (6.1) and in extracting a phase shift for a particular value of the energy E and angular momentum ℓ. In order to include the routine D02AHF in this comparison we shall require that it incorporates an error per unit step criterion and that the error test

$$\left|\frac{E(I)}{H}\right| < G(I) \times \text{Max}\left\{.1, |YP(I)|\right\}$$

for each component of the predicted solution vector YP is satisfied. Of course the same initial steplength and mesh point must be supplied in DEVOG, NUMOV and RAPAL in order to provide a meaningful comparison. We shall refer henceforth to the modified version of D02AHF as DNGMOD and to the program which uses DNGMOD to solve (6.1) as NAGMOD. DNGMOD requires as starting values the solution and its first derivative at the initial mesh point.

In order to compute a phase shift a decision must be taken within NAGMOD as to when such a calculation should begin. The decision is actually taken within the auxilliary routine AUX and the calculation of a phase shift is signalled when the integer parameter IPS is set

to 1 in AUX. Just as in program RADISH the phase shift is calculated in routine PS which is called by routine DNGMOD; if the phase shift has converged to the required accuracy the integer parameter JCONV is set to 1 in DNGMOD and when this condition is encountered ST is reset to 1 which has the effect of automatically terminating the calculation in DNGMOD. A message is printed to the effect that the phase shift has converged to the required accuracy along with information concerning the number of function evaluations performed during the course of the integration. Figure 6 below shows the relationship between the six routines of program NAGMOD.

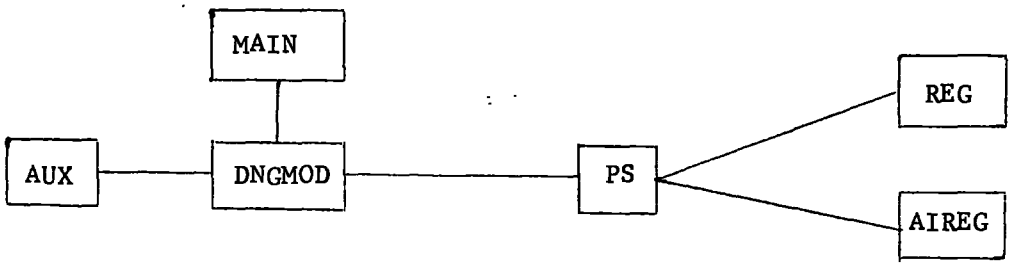

Figure 6

§ 6.3 Test runs

Results for the performance of program NAGMOD in solving equation (6.1) in the equivalent form of two first order coupled differential equations for a range of values of E and ℓ will be described in Chapter 7.

We consider here the solution of problem (i) of §5.7.4 in its equivalent form of two first order coupled differential equations. The exact solution is $y = xj_1(kx)$ and exact starting values for x_o, y_o and $z_o = \left.\frac{dy}{dx}\right|_{x = x_o}$ are used. The initial value of the steplength is chosen to be

$$h = \min\left[\frac{EPS}{|z_o|}, \frac{EPS}{|f_o|}\right]$$

where f_o may be easily calculated and EPS is different to that calcul-

ated internally in routine DNGMOD and takes the values EPS = 10^{-n}, n = 3, 4, 6, 8 for a range of values of k. The above estimate of the initial steplength incorporates an error per unit step criterion and is based on the assumption that the error of a first order method is h times that of a zero order method. The results for this problem using NAGMOD are tabulated in Table 11 of § 6.4.

§ 6.4 Test results

We see from Table 11 that routine DNGMOD does an extremely good job of controlling the global error and we note that DNGMOD has a number of sophisticated features which are absent from routine DEVOG, NUMOV and RAPAL. The value of a_{max} is 0.572 and consequently a_p and a_{p5} are both zero. However the penalty introduced in gaining such high accuracy is reflected by the large number of function evaluations, most noticeably for the lower energy values; even at the high energy values the number of function evaluations far exceeds the corresponding number in Table 7 for program NUMEROV2. Since the method starts with a formula of first order the initial value of the steplength is necessarily small particularly when the accuracy requirement is high. The value of H which is tabulated is that which is accepted by DNGMOD as sufficiently accurate for the first order method; the value supplied by the user will invariably be decreased in order that the new steplength adhere to the steplength strategy of DNGMOD (see § 6.1.3). Notice that for EPS = 10^{-8} with k = 1.0, 2.0, 5.0 routine DNGMOD returns the error indicator IFAIL = 1; the initial value of H acceptable in DNGMOD is so small that the computer is unable to handle the large number of steps required to complete the interval. The typical behaviour of routine DNGMOD is for the order to increase fairly rapidly in the initial phase of the integration once an appropriate steplength

at each order has been found.

One might argue a case for using NAGMOD with an error per step criterion in view of the exceedingly small (relative to EPS) values of the scaled maximum error, a, which appear in columns 3 - 5 of Table 11 throughout the range of EPS. However a test run using NAGMOD incorporating an error per step criterion to solve the problem of § 6.3 proves to be totally unsatisfactory with respect to control of the global error. The initial steplength is chosen to be

$$h = \min\left[\left(\frac{EPS}{|z_o|}\right)^{\frac{1}{2}}, \left(\frac{EPS}{|f_o|}\right)^{\frac{1}{2}}\right]$$

and the initial steplength accepted by routine DNGMOD is tabulated in Table 12 along with the scaled maximum errors for the above problem. The relevant statistics are

$$a_{max} = 271,\ a_p \simeq 88\%,\ a_{p5} \simeq 67\%$$

and we note that in the majority of cases a exceeds EPS by a factor far in excess of five throughout the range of EPS.

TABLE 11

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	1.23(-4)	7.54(-5)	3.77(-5)	0.122(-2)	136
	0.2	1.50(-4)	7.53(-5)	3.77(-5)	0.122(-2)	129
	0.5	2.35(-5)	1.18(-5)	1.40(-5)	0.610(-3)	169
	1.0	3.32(-6)	1.66(-6)	2.11(-4)	0.305(-3)	272
	2.0	1.77(-6)	1.53(-4)	1.28(-4)	0.305(-3)	419
	5.0	1.66(-4)	1.17(-4)	1.10(-4)	0.152(-3)	648
10^{-4}	0.1	2.71(-6)	1.66(-6)	8.32(-7)	0.305(-3)	185
	0.2	3.32(-6)	1.66(-6)	8.32(-7)	0.305(-3)	198
	0.5	4.42(-7)	2.21(-7)	9.21(-7)	0.152(-3)	229
	1.0	5.71(-8)	1.50(-5)	5.72(-5)	0.763(-4)	294
	2.0	2.23(-6)	4.41(-5)	2.21(-5)	0.381(-4)	468
	5.0	8.21(-6)	1.00(-5)	2.07(-5)	0.191(-4)	864
10^{-6}	0.1	8.87(-11)	5.45(-11)	2.73(-11)	0.953(-5)	251
	0.2	8.08(-12)	4.05(-12)	2.03(-12)	0.477(-5)	273
	0.5	1.20(-12)	3.16(-12)	8.12(-9)	0.238(-5)	305
	1.0	2.60(-10)	7.85(-8)	5.27(-8)	0.119(-5)	412
	2.0	2.32(-8)	6.94(-8)	5.17(-8)	0.596(-6)	644
	5.0	3.03(-8)	1.52(-8)	7.68(-9)	0.298(-6)	1267
10^{-8}	0.1	5.53(-12)	3.40(-12)	1.70(-12)	0.149(-6)	261
	0.2	8.68(-12)	4.34(-12)	2.17(-12)	0.745(-7)	321
	0.5	1.16(-12)	2.24(-12)	3.12(-11)	0.186(-7)	379
	1.0	*	*	*		
	2.0	*	*	*		
	5.0	*	*	*		

* IFAIL = 1

TABLE 12

EPS	k	d = 5	d = 10	d = 20	INITIAL H	N
10^{-3}	0.1	3.14(-2)	1.93(-2)	9.64(-3)	0.390(-1)	58
	0.2	2.33(-2)	1.17(-2)	5.83(-3)	0.195(-1)	83
	0.5	1.05(-2)	5.25(-3)	2.63(-3)	0.976(-2)	109
	1.0	1.05(-2)	5.79(-3)	3.05(-3)	0.976(-2)	123
	2.0	2.35(-3)	1.18(-3)	1.29(-3)	0.488(-2)	201
	5.0	1.95(-3)	9.74(-4)	6.57(-4)	0.488(-2)	517
10^{-4}	0.1	8.55(-3)	5.25(-3)	2.63(-3)	0.976(-2)	94
	0.2	3.40(-3)	1.70(-3)	8.52(-4)	0.488(-2)	123
	0.5	3.40(-3)	1.70(-3)	8.52(-4)	0.488(-2)	116
	1.0	8.10(-4)	4.05(-4)	2.03(-4)	0.244(-2)	233
	2.0	4.29(-4)	2.15(-4)	1.73(-4)	0.244(-2)	282
	5.0	4.83(-5)	4.14(-5)	4.81(-5)	0.122(-2)	557
10^{-6}	0.1	1.23(-4)	7.54(-5)	3.77(-5)	0.122(-2)	134
	0.2	1.50(-4)	7.53(-5)	3.77(-5)	0.122(-2)	128
	0.5	9.33(-7)	1.96(-6)	5.88(-6)	0.610(-3)	176
	1.0	2.32(-5)	1.16(-5)	5.82(-6)	0.610(-3)	230
	2.0	1.77(-6)	1.01(-6)	6.45(-7)	0.305(-3)	507
	5.0	9.27(-7)	7.19(-7)	8.80(-7)	0.305(-3)	964
10^{-8}	0.1	2.71(-6)	1.66(-6)	8.32(-7)	0.305(-3)	183
	0.2	4.41(-7)	2.21(-7)	1.10(-7)	0.152(-3)	181
	0.5	4.42(-7)	2.21(-7)	1.11(-7)	0.152(-3)	263
	1.0	5.71(-8)	4.32(-8)	2.19(-8)	0.763(-4)	383
	2.0	4.89(-8)	2.52(-8)	1.36(-8)	0.763(-4)	617
	5.0	3.93(-9)	5.62(-9)	6.91(-9)	0.381(-4)	1153

CHAPTER 7

Numerical comparison of RADISH, NUMEROV2, EXPFIT2, NAGMOD

Introduction

We have developed six programs, namely RADISH, NUMEROV1, NUMEROV2, EXPFIT1, EXPFIT2 and NAGMOD which incorporate numerical methods with automatic error control for the solution of

$$y''(x) = \left[\frac{L(L+1)}{x^2} - E + V(x) \right] y(x) \quad , \qquad (7.1)$$

We are now in a position to undertake a numerical comparison of the performance of the methods of de Vogelaere, Numerov, Raptis and Allison and the Adams Bashforth-Adams Moulton method of Chapter 6, when used to integrate (7.1) with automatic error control. We shall investigate the reliability of each of the numerical methods with respect to control of the global error and we are also interested in the relative efficiency of the programs in calculating the phase shift for problem (7.1) for a range of values of energy E, angular momentum L and the potential function V(x).

We saw in Chapters 4 and 5 that NUMEROV2 and EXPFIT2 do a considerably better job of controlling the global error in the numerical integration stage of the calculation than do NUMEROV1 and EXPFIT1 respectively and for this reason we exclude NUMEROV1 and EXPFIT1 from our comparison.

In order that the comparisons be meaningful we must provide as far as possible the same starting conditions for the solution of (7.1) and in § 7.1 we discuss the modifications which must be performed in order that the programs may be reasonably compared. We describe the test problems in § 7.2 and the results and conclusions of our numerical comparison are discussed in § 7.3.

§7.1 Criteria for comparison.

The initial values which must be supplied to programs RADISH and NAGMOD are approximations for the solution and its first derivative, y_o and z_o, at the starting point x_o; programs NUMEROV2 and EXPFIT2 require approximations for the solution y_o at x_o and y_1 at the next point x_1 where $x_1 = x_o+h$ with h the initial steplength. The same values of x_o and y_o must be used by each program and we arrange that the second starting condition is such that the accuracy attained in the computed solution at the end of the first step meets the necessary requirement particular to the method of solution being used. To this end the criterion for choosing x_o, in the notation of §3.5, is

$$|a_5|x_o^{L+5} \leq 0.1 \times EPS \times EPS \tag{7.2}$$

(see §4.4.3). y_o is then computed as

$$y_o = \sum_{i=1}^{4} a_i x_o^{L+i} . \tag{7.3}$$

The first 8 coefficients of the expansion for the potential V(x) given by (3.50), are read as data in the main routine of each program and the coefficients a_2, ---, a_9 are calculated with $a_1 = 1$. In order that the term hz_o be given sufficiently accurately we take

$$z_o = \sum_{I=1}^{J} (L + I)\, a_I x_o^{L+I-1}, \qquad 4 \leq J \leq 8$$

where J is the first value of I (>4) for which the following condition holds

$$h(L+I)\,|a_{J+1}|\,x_o^{L+J} \leq 0.1 \times EPS \times EPS. \tag{7.4}$$

This criterion is also used for choosing z_o in NAGMOD. The criterion for choosing y_1 in NUMEROV2 and EXPFIT2 has already been discussed in §4.4.3.

Thus the following modifications must be made to program RADISH in order that it be used in the numerical comparison; 8 coefficients

of the expansion for the potential must be supplied to the main routine and x_o, y_o, z_o must be chosen as described above.

In order to study the actual errors which accumulate in routines DEVOG, NUMOV, RAPAL and DNGMOD during the course of the integration we shall use routine D02AHF to provide high accuracy parallel solutions to those calculated by the methods of de Vogelaere, Numerov, Raptis and Allison and Adams Bashforth-Adams Moulton respectively. Now routine D02AHF will require the value of z_o and the main routines of RADISH, NUMEROV2, EXPFIT2 and NAGMOD must provide the value of z_o for this purpose.

We shall provide the programs with the same value of XF, the endpoint of the range of integration; it is not necessary however when solving (7.1) to insist that the integration end exactly at XF. In the case of NUMEROV2 and EXPFIT2 the integration will stop somewhere between XF-H and XF and in the case of RADISH it will stop between XF-H and XF+H. The integration in NAGMOD will stop precisely at XF since the steplength within routine DNGMOD is in fact chosen to meet this requirement.

We shall solve a set of test problems which are determined by the form of the potential V(x) in equation (7.1). One aim is to produce a set of tables of the form of Table 12 which records the scaled maximum error over $[x_o, d]$ where d = 5, 10, 20. The values a_{max}, a_p and a_{p5} which may be extracted from the tables reflect the reliability of a particular algorithm in solving the test problem for a range of values of energy and angular momentum subject to different error requirements. Our second aim is to compare the phase shifts computed by our programs with the published values and to compare for each program the CPU times required by the computer to perform the numerical integration and extraction of the phase shift.

We have tried as far as possible to write programs RADISH, NUMEROV2

and EXPFIT2 to the same specification. The steplength strategy is the same for each of the numerical methods employed in these programs; increases are limited to a factor of two and decreases must be allowed wherever necessary and the appropriate decrease factor chosen. Decreases in NAGMOD however are restricted to halving only. The steplength criteria adopted in RADISH, NUMEROV2 and EXPFIT2 have been seen to work quite adequately in each program although it may well be that these criteria could be further improved.

§ 7.2 Test problems.

We consider the numerical solution of (7.1) over the range $[x_o, 20]$ with an initial steplength of 0.1 (this value is completely arbitrary) supplied by the master driver to the integration routine (DEVOG, NUMOV, RAPAL or DNGMOD) with the potential given by one of the following:

$$\text{(i)} \qquad V(x) = -2\left(1+\frac{1}{x}\right)e^{-2x}$$

$$\text{(ii)} \qquad V(x) = -2\left(\frac{1}{x}+\frac{3}{4}+\frac{1}{4}\,x+\frac{1}{8}x^2\right)e^{-x}$$

$$\text{(iii)} \qquad V(x) = -\frac{2}{x}\,e^{-x}$$

for the following range of values of $k(E = k^2)$:

$$k = 0.1,\ 0.2,\ 0.5,\ 1.0,\ 2.0,\ 5.0$$

with L = 1, and EPS = 10^{-n}, n = 3, 4, 6, 8.

For each of (i) - (iii) we have drawn up tables which for programs RADISH, NUMEROV2, EXPFIT2 and NAGMOD display the value of EPS, k, the scaled maximum error $\mathcal{A}^{a}$ over 5, 10 and 20 units, H the initial value of the steplength accepted as sufficiently accurate and N the number of function evaluations performed over $[x_o, 20]$. The values of a_{max}, a_p and a_{p5} in each table are also shown. Those tables for (i) - (iii) are listed in § 7.3; the potentials of (i), (ii) and (iii) are respectively the effective potential between an electron and the ground state

of a hydrogen atom, the effective potential between an electron and the 2s state of a hydrogen atom and the screened coulomb potential.

In addition we solve (7.1) for a range of values of E and L with V(x) given by (i) - (iii) where the values of E and L are chosen to allow comparison of the computed phase shifts δ_L with the published values. The range of values of E and L are tabulated along with the published value of the phase shift and the computed phase shifts obtained from RADISH, NUMEROV2, EXPFIT2 and NAGMOD. These calculations are performed for EPS = 10^{-6}, PSIG = 10^{-4}. Also tabulated are the CPU times required by the four programs to perform the numerical integration of (7.1) and to extract a phase shift. Some other test problems have also been considered, namely the solution of (7.1) by each program with V(x) given by (iv) - (vi) below:

$$\text{(iv)} \qquad V(x) = -2e^{-x}$$

$$\text{(v)} \qquad V(x) = 2\left(\frac{1+1}{x}\right)e^{-2x}$$

$$\text{(vi)} \qquad V(x) = 2\left(\frac{1+1}{x}\right)e^{-2x} - \frac{\alpha(x)}{x^4}, \quad \alpha(x) = \frac{9}{2} - \frac{2}{3}e^{-2x}\left(x^5 + \frac{9}{2}x^4 + 9x^3 + \frac{27}{2}x^2 + \frac{27}{2}x + \frac{27}{4}\right)$$

The results have been tabulated as for (i) - (iii) but we shall not present the tables here.

All calculations are performed on the NUMAC (Northumbrian Universities Multiple Access Computer) IBM 370/168 computer and all real variables used in the calculations are in double precision form.

§ 7.3 Test results.

Tables 13 - 15 display the scaled maximum errors over the range of integration $[x_o, 20]$ for (i) - (iii) respectively using program RADISH. We notice that a, the factor by which the scaled maximum error exceeds EPS exceeds unity at each of the tolerances tested particularly at the more stringent ones. The maximum value of a, a_{max}, which exceeds 5.0 is recorded for EPS = 10^{-8} and for EPS in the range 10^{-n}, n = 3, 4, 6 a does not exceed 2.5.

Tables 16 - 18 and 19 - 21 display the scaled maximum errors over $[x_o, 20]$ using programs NUMEROV2 and EXPFIT2 respectively for (i) - (iii). The values of a_{max} obtained using NUMEROV2 are less than 5.0 and consequently a_{p5} is zero. a exceeds unity only at the more stringent tolerances (EPS = 10^{-6}, 10^{-8}) but the percentage of such cases which is reflected by the value of a_p is very small. A comparison of the number of function evaluations performed by RADISH and NUMEROV2 for each of (i) - (iii) shows that NUMEROV2 is more efficient in this respect. NUMEROV2 and RADISH both control an error which is proportional to h^4 where h is the steplength for NUMEROV2 and 2h is the corresponding steplength for RADISH; the coefficient of the error in NUMEROV2 is smaller by a factor which exceeds five and this might explain why the steplength increases more rapidly in NUMEROV2. The values of a_{max} obtained using EXPFIT2 are higher than those of NUMEROV2; again a exceeds unity only at the more stringent tolerances and a_p, although slightly higher than in NUMEROV2, remains small. Note that the values of H listed for NUMEROV2 and EXPFIT2 for a particular test problem are the same; this is to be expected since EXPFIT2 initially uses the method of Numerov until a change-over to the Raptis and Allison method is effected. This change-over subsequently speeds up the calculation by allowing a more rapid increase in the steplength. This is reflected by a substantial decrease in the number of function evaluations performed using EXPFIT2 over NUMEROV2 particularly in cases where the energy is large; EXPFIT2 often uses less than one third of the corresponding number of function evaluations of NUMEROV2.

Tables 22 - 24 display the scaled maximum errors over $[x_o, 20]$ using program NAGMOD. a_{max} is less than unity and consequently a_p and a_{p5} are both zero. The values of H accepted by routine DNGMOD as sufficiently accurate are exceedingly small and are approximately one order of magnitude smaller than EPS; for EPS = 10^{-8} the routine

consistently returns the error indicator IFAIL=1. While program NAGMOD does an extremely good job of controlling the global error the excessive number of function evaluations which must be performed particularly at the larger tolerances (EPS = 10^{-3}, 10^{-4}) is clearly undesirable. We have rerun program NAGMOD incorporating instead an error per step criterion (this will relax the condition for choosing an initial value of H) for the solution of (7.1) with the potential given by (i) - (vi). We still find that a large number of function evaluations must be performed although the number is considerably less than in NAGMOD with an error per unit step criterion particularly for EPS = 10^{-3}, 10^{-4}. At the larger energies and at the more stringent tolerances the number of function evaluations performed is considerably less than in RADISH and NUMEROV2. The values of a_{max} however for (i) - (vi) are large and vary from approximately seven to seventy five when an error per step criterion is used in NAGMOD and in addition the ratio of a_{p5} to a_p is fairly large; a number of cases occur where a exceeds EPS by a factor which is far in excess of 5.0. Thus we see that NAGMOD with an error per step criterion does a relatively poor job of controlling the global error.

Tables 25, 27 and 29 list the published and computed values of the phase shift δ_L for test problems (i), (ii) and (iii) respectively for a range of values of energy E and angular momentum L. The CPU times (in seconds) required to perform the numerical integration and calculation of the phase shift for test problems (i), (ii) and (iii) are listed in Tables 26, 28 and 30 respectively.

In most cases the phase shifts produced by the four programs agree within themselves to the number of figures quoted in the published values; where there are small discrepancies our results are confirmed by calculations which we have carried out with more stringent accuracy

requirements (e.g. EPS = 10^{-8}, PSIG = 10^{-5}). In particular if we use a more stringent accuracy requirement in RADISH and EXPFIT2 for problem (iii) then the agreement between the phase shifts obtained using RADISH, EXPFIT2 and the remaining programs is much improved. Of the four programs, EXPFIT2 performs by far the least number of function evaluations in the numerical integration stage of the calculation particularly at the higher energies and we might expect EXPFIT2 to be the most efficient with respect to computer time considerations. NAGMOD uses considerably more function evaluations than does RADISH which in turn uses more than NUMEROV2.

The CPU times which are listed in Tables 26, 28 and 30 are subject to variations of approximately $\pm$ 5% due to a time sharing environment but it is possible to ascertain which of the programs is the most efficient in terms of the amount of CPU time required to perform the numerical integration and subsequent extraction of the phase shift. At small energies the CPU times are comparable for RADISH and NUMEROV2 (being slightly in favour of RADISH) and we see that EXPFIT2 is less efficient than NUMEROV2 in this respect. At the larger energies ($E > 1$) RADISH uses less time than EXPFIT2 and considerably less (often by a factor which is near a half) time than NUMEROV2. In all cases EXPFIT2 performs less function evaluations than does NUMEROV2; however the overheads in EXPFIT2 are higher than in NUMEROV2 because of the additional complexity of routine DIVDIF in EXPFIT2. For more complicated forms of the potential we might expect EXPFIT2 to be the most efficient of the programs with respect to computing time. As expected, the CPU times required by NAGMOD far exceed those of RADISH, NUMEROV2 and EXPFIT2.

We ask the question: which program gives the most reliable and efficient solution of (7.1) for the range of data considered? Program NAGMOD with an error per unit step criterion undoubtedly gives

the best control of the global error. However the excessive number of function evaluations which must be performed disqualifies NAGMOD from the point of view of efficiency. Of the remaining three programs NUMEROV2 and EXPFIT2 give the best control of the global error. If one is interested primarily in efficiency then EXPFIT2 requires the least number of function evaluations but RADISH is the more efficient with respect to computer time considerations. This presumably is due to the method of calculating the truncation error estimates which in EXPFIT2 is a time consuming process; subroutine DIVDIF might well be improved to give a more efficient process for the calculation of the derivative required in the truncation error estimate. Each of the programs give reliable values for the phase shift and we would recommend the use of program RADISH to the user who is interested in extracting a phase shift with the minimum expenditure of computing time. RADISH also gives a good control of the global error for EPS = 10^{-3}, 10^{-4}, 10^{-6}; the user interested in extracting phase shifts will not normally request high accuracy in the solution. For the user who is primarily interested in the control of the global error in the numerical integration stage of the calculation we recommend the use of program NUMEROV2 or EXPFIT2; at the most stringent tolerance tested, that is EPS = 10^{-8}, NUMEROV2 gives a slightly better control of the global error than does EXPFIT2 but for the lower (realistic) tolerances either of the two programs is recommended. If in addition phase shifts are to be extracted the choice between the two programs is more clear-cut. With respect to the efficient use of computing time we recommend the use of NUMEROV2 for low (<1, say) energy values and EXPFIT2 for the higher energy values.

TABLE 13

RADISH for problem(i)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	4.58(-5)	2.39(-5)	1.17(-5)	0.1000	38
	0.2	4.66(-5)	2.32(-5)	5.64(-5)	0.1000	38
	0.5	4.52(-5)	5.22(-4)	7.06(-4)	0.1000	44
	1.0	6.57(-4)	9.21(-4)	1.65(-3)	0.1000	64
	2.0	5.65(-4)	8.71(-4)	1.06(-3)	0.1000	106
	5.0	1.19(-3)	1.32(-3)	1.33(-3)	0.1000	202
10^{-4}	0.1	7.18(-5)	3.48(-5)	1.81(-5)	0.0667	52
	0.2	7.24(-5)	3.36(-5)	1.69(-5)	0.0667	56
	0.5	7.37(-5)	3.72(-5)	1.13(-4)	0.0663	78
	1.0	5.06(-5)	1.05(-4)	1.60(-4)	0.0573	114
	2.0	7.44(-5)	1.04(-4)	1.14(-4)	0.0686	190
	5.0	1.03(-4)	1.10(-4)	1.18(-4)	0.0537	388
10^{-6}	0.1	1.61(-6)	8.66(-7)	3.50(-7)	0.0086	126
	0.2	1.74(-6)	8.47(-7)	1.30(-6)	0.0086	140
	0.5	1.79(-6)	8.91(-7)	8.18(-7)	0.0085	226
	1.0	2.23(-6)	1.44(-6)	1.58(-6)	0.0083	344
	2.0	1.52(-6)	9.90(-7)	8.51(-7)	0.0082	630
	5.0	8.94(-7)	6.38(-7)	5.81(-7)	0.0071	1432
10^{-8}	0.1	2.85(-8)	1.48(-8)	6.76(-9)	0.0013	314
	0.2	3.10(-8)	1.56(-8)	8.90(-9)	0.0013	364
	0.5	4.60(-8)	2.32(-8)	1.90(-8)	0.0013	602
	1.0	8.86(-8)	4.44(-8)	4.38(-8)	0.0013	998
	2.0	9.70(-8)	4.84(-8)	2.69(-8)	0.0013	1702
	5.0	3.38(-8)	2.18(-8)	1.80(-8)	0.0009	3496

$a_{max} = 9.70$, $a_p = 75\%$, $a_{p5} \simeq 8\%$

TABLE 14

RADISH for problem (ii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	1.97(-5)	8.32(-6)	3.36(-6)	0.1000	40
	0.2	2.33(-5)	1.29(-5)	7.37(-5)	0.1000	42
	0.5	1.32(-4)	9.97(-5)	7.11(-4)	0.1000	46
	1.0	4.87(-4)	9.68(-4)	1.06(-3)	0.1000	62
	2.0	6.40(-4)	8.64(-4)	9.00(-4)	0.1000	108
	5.0	1.17(-3)	1.18(-3)	1.29(-3)	0.1000	202
10^{-4}	0.1	3.80(-5)	1.82(-5)	9.17(-6)	0.0795	58
	0.2	4.09(-5)	2.07(-5)	1.11(-5)	0.0784	58
	0.5	4.92(-5)	2.71(-5)	8.63(-5)	0.0751	76
	1.0	6.04(-5)	3.86(-5)	8.03(-5)	0.0729	122
	2.0	3.27(-5)	7.37(-5)	9.80(-5)	0.0741	198
	5.0	9.84(-5)	1.06(-4)	1.09(-4)	0.0530	386
10^{-6}	0.1	1.84(-6)	8.81(-7)	4.18(-7)	0.0123	142
	0.2	1.80(-6)	8.45(-7)	4.21(-7)	0.0121	150
	0.5	2.52(-6)	1.55(-6)	1.35(-6)	0.0115	196
	1.0	1.95(-6)	1.77(-6)	1.57(-6)	0.0095	294
	2.0	1.35(-6)	1.06(-6)	9.97(-7)	0.0090	596
	5.0	9.16(-7)	6.34(-7)	5.65(-7)	0.0071	1434
10^{-8}	0.1	8.32(-9)	5.42(-9)	2.58(-9)	0.0017	366
	0.2	8.49(-10)	2.47(-9)	1.23(-8)	0.0017	398
	0.5	3.24(-8)	1.80(-8)	2.60(-8)	0.0016	580
	1.0	4.28(-8)	2.82(-8)	2.18(-8)	0.0015	840
	2.0	7.74(-8)	4.11(-8)	2.42(-8)	0.0014	1676
	5.0	3.45(-8)	2.12(-8)	1.72(-8)	0.0009	3520

$a_{max} = 7.74$, $a_p \eqsim 54\%$, $a_{p5} \eqsim 4\%$.

TABLE 15

RADISH for problem (iii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	7.24(-5)	3.74(-5)	1.86(-5)	0.1000	38
	0.2	7.22(-5)	3.59(-5)	6.35(-5)	0.1000	38
	0.5	6.53(-5)	4.46(-4)	6.19(-4)	0.1000	44
	1.0	7.45(-4)	9.27(-4)	1.37(-3)	0.1000	62
	2.0	5.95(-4)	8.51(-4)	9.34(-4)	0.1000	106
	5.0	1.17(-3)	1.31(-3)	1.33(-3)	0.1000	202
10^{-4}	0.1	5.18(-5)	2.86(-5)	1.45(-5)	0.0548	56
	0.2	5.17(-5)	2.71(-5)	5.16(-5)	0.0548	60
	0.5	5.77(-5)	2.69(-5)	1.53(-4)	0.0553	76
	1.0	6.73(-5)	1.09(-4)	1.61(-4)	0.0570	110
	2.0	7.35(-5)	1.08(-4)	1.17(-4)	0.0698	188
	5.0	1.03(-4)	1.13(-4)	1.17(-4)	0.0538	382
10^{-6}	0.1	2.36(-6)	1.20(-6)	5.33(-7)	0.0079	128
	0.2	2.45(-6)	1.20(-6)	6.46(-7)	0.0079	140
	0.5	2.37(-6)	1.17(-6)	1.27(-6)	0.0080	226
	1.0	2.48(-6)	1.54(-6)	1.38(-6)	0.0081	348
	2.0	1.47(-6)	1.00(-6)	8.78(-7)	0.0084	620
	5.0	9.01(-7)	6.39(-7)	5.77(-7)	0.0071	1434
10^{-8}	0.1	8.62(-8)	4.38(-8)	2.14(-8)	0.0012	320
	0.2	8.80(-8)	4.43(-8)	2.20(-8)	0.0012	364
	0.5	8.99(-8)	4.53(-8)	2.26(-8)	0.0012	604
	1.0	1.04(-7)	5.18(-8)	4.06(-8)	0.0013	974
	2.0	9.23(-8)	4.61(-8)	2.57(-8)	0.0013	1726
	5.0	3.38(-8)	2.15(-8)	1.77(-8)	0.0009	3514

$a_{max} = 10.4, \quad a_p \simeq 71\%, \quad a_{p5} \simeq 21\%$

TABLE 16

NUMEROV2 for problem(i)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	4.90(-6)	3.19(-6)	1.65(-6)	0.1000	24
	0.2	4.74(-6)	3.10(-6)	3.27(-5)	0.1000	24
	0.5	7.27(-6)	3.92(-4)	7.05(-4)	0.1000	33
	1.0	5.07(-5)	5.15(-5)	6.82(-5)	0.1000	57
	2.0	2.72(-5)	4.21(-5)	4.63(-5)	0.1000	102
	5.0	5.20(-5)	5.59(-5)	5.92(-5)	0.1000	198
10^{-4}	0.1	3.00(-7)	2.68(-7)	1.48(-7)	0.1000	31
	0.2	3.17(-7)	3.60(-7)	4.37(-5)	0.1000	31
	0.5	1.85(-6)	5.11(-5)	1.29(-5)	0.1000	49
	1.0	2.67(-5)	2.53(-5)	3.58(-5)	0.1000	73
	2.0	1.46(-5)	1.96(-5)	2.00(-5)	0.1000	127
	5.0	5.76(-6)	6.28(-6)	6.55(-6)	0.0688	351
10^{-6}	0.1	8.18(-8)	2.89(-8)	5.61(-8)	0.0548	70
	0.2	8.11(-8)	2.76(-8)	1.69(-6)	0.0550	76
	0.5	7.58(-8)	2.83(-7)	3.85(-7)	0.0565	122
	1.0	3.30(-7)	2.75(-7)	3.49(-7)	0.0618	211
	2.0	1.56(-7)	1.87(-7)	1.55(-7)	0.0667	421
	5.0	6.74(-8)	7.25(-8)	7.53(-8)	0.0189	1067
10^{-8}	0.1	6.26(-10)	6.33(-10)	3.34(-10)	0.0174	175
	0.2	6.72(-10)	1.49(-9)	2.61(-8)	0.0174	187
	0.5	4.27(-10)	4.33(-9)	1.86(-8)	0.0179	336
	1.0	3.47(-9)	3.22(-9)	4.70(-9)	0.0197	630
	2.0	1.69(-9)	1.90(-9)	1.88(-9)	0.0160	1246
	5.0	7.56(-10)	8.00(-10)	8.75(-10)	0.0057	3222

$a_{max} = 2.61, \quad a_p \simeq 12\%, \quad a_{p5} = 0$

TABLE 17

NUMEROV2 for problem (ii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	2.35(-5)	2.88(-5)	1.10(-5)	0.1000	26
	0.2	2.82(-5)	4.41(-5)	1.35(-4)	0.1000	26
	0.5	1.09(-4)	1.01(-4)	3.89(-4)	0.1000	28
	1.0	2.58(-5)	4.20(-5)	4.49(-5)	0.1000	56
	2.0	4.02(-5)	4.02(-5)	3.95(-5)	0.1000	102
	5.0	5.23(-5)	5.60(-5)	5.73(-5)	0.1000	198
10^{-4}	0.1	4.94(-6)	4.66(-6)	1.45(-6)	0.1000	34
	0.2	5.45(-6)	6.15(-6)	4.34(-5)	0.1000	34
	0.5	1.52(-5)	8.63(-6)	2.29(-5)	0.1000	42
	1.0	1.71(-5)	3.17(-5)	3.98(-5)	0.1000	61
	2.0	1.50(-5)	1.68(-5)	1.78(-5)	0.1000	129
	5.0	4.61(-6)	4.88(-6)	5.01(-6)	0.0650	376
10^{-6}	0.1	2.16(-7)	1.49(-7)	1.37(-7)	0.0782	77
	0.2	2.31(-7)	2.03(-7)	1.07(-7)	0.0790	85
	0.5	5.06(-7)	2.68(-7)	3.60(-7)	0.1000	98
	1.0	1.98(-7)	1.90(-7)	1.84(-7)	0.1000	198
	2.0	1.38(-7)	1.18(-7)	1.18(-7)	0.1000	413
	5.0	5.78(-8)	6.07(-8)	6.23(-8)	0.0182	1103
10^{-8}	0.1	2.41(-9)	1.58(-9)	2.22(-9)	0.0248	207
	0.2	2.55(-9)	1.88(-9)	2.61(-8)	0.0251	227
	0.5	5.49(-9)	2.92(-9)	5.23(-9)	0.0274	296
	1.0	3.93(-9)	3.65(-9)	3.83(-9)	0.0278	557
	2.0	2.09(-9)	2.21(-9)	2.28(-9)	0.0193	1164
	5.0	7.60(-10)	8.73(-10)	9.31(-10)	0.0055	3188

$a_{max} = 2.61$, $a_p \simeq 4\%$, $a_{p5} = 0.$

TABLE 18

NUMEROV2 for problem (iii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	1.04(-5)	4.62(-6)	2.31(-6)	0.1000	24
	0.2	1.07(-5)	4.77(-6)	3.46(-5)	0.1000	24
	0.5	9.31(-6)	2.45(-4)	7.29(-4)	0.1000	31
	1.0	4.39(-5)	4.64(-5)	6.12(-5)	0.1000	56
	2.0	2.78(-5)	4.22(-5)	4.48(-5)	0.1000	102
	5.0	5.33(-5)	5.68(-5)	5.89(-5)	0.1000	198
10^{-4}	0.1	2.72(-6)	1.37(-6)	6.92(-7)	0.1000	31
	0.2	2.78(-6)	1.29(-6)	4.55(-5)	0.1000	31
	0.5	3.17(-6)	6.06(-5)	3.48(-5)	0.1000	48
	1.0	2.20(-5)	2.26(-5)	2.97(-5)	0.1000	71
	2.0	1.25(-5)	1.55(-5)	1.59(-5)	0.1000	133
	5.0	5.65(-6)	6.11(-6)	6.37(-6)	0.0684	353
10^{-6}	0.1	1.68(-7)	8.35(-8)	4.52(-8)	0.0497	72
	0.2	1.70(-7)	7.18(-8)	3.74(-7)	0.0498	78
	0.5	1.72(-7)	2.01(-6)	1.20(-6)	0.0506	128
	1.0	3.36(-7)	2.62(-7)	3.16(-7)	0.0530	209
	2.0	1.61(-7)	2.24(-7)	1.77(-7)	0.0538	410
	5.0	6.37(-8)	6.82(-8)	7.06(-8)	0.0186	1084
10^{-8}	0.1	2.77(-9)	1.17(-9)	3.90(-10)	0.0156	189
	0.2	2.76(-9)	1.21(-9)	4.40(-9)	0.0157	218
	0.5	2.59(-9)	1.34(-8)	1.30(-8)	0.0159	328
	1.0	3.93(-9)	3.21(-9)	4.27(-9)	0.0167	617
	2.0	1.76(-9)	2.16(-9)	2.28(-9)	0.0169	1189
	5.0	7.18(-10)	7.82(-10)	8.37(-10)	0.0056	3303

$a_{max} = 2.01, \quad a_p \simeq 8\% \quad a_{p5} = 0$

TABLE 19

EXPFIT2 for problem(i)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	4.90(-6)	3.19(-6)	1.65(-6)	0.1000	24
	0.2	4.74(-6)	2.98(-6)	2.94(-6)	0.1000	25
	0.5	2.65(-6)	2.31(-5)	8.50(-5)	0.1000	27
	1.0	4.79(-5)	2.68(-5)	7.82(-5)	0.1000	33
	2.0	2.02(-5)	3.68(-5)	2.93(-5)	0.1000	43
	5.0	8.35(-6)	1.69(-5)	1.24(-5)	0.1000	59
10^{-4}	0.1	3.00(-7)	2.68(-7)	1.48(-7)	0.1000	31
	0.2	3.17(-7)	2.73(-7)	5.62(-7)	0.1000	31
	0.5	3.35(-7)	1.27(-5)	6.18(-6)	0.1000	38
	1.0	3.91(-6)	9.56(-6)	1.28(-5)	0.1000	45
	2.0	1.29(-6)	4.53(-6)	3.12(-6)	0.1000	69
	5.0	2.06(-6)	2.22(-6)	1.93(-6)	0.0688	107
10^{-6}	0.1	8.18(-8)	2.89(-8)	5.77(-8)	0.0548	70
	0.2	8.11(-8)	3.01(-8)	1.38(-7)	0.0550	71
	0.5	7.18(-8)	2.13(-7)	7.33(-7)	0.0565	90
	1.0	7.20(-8)	1.25(-7)	1.01(-7)	0.0618	125
	2.0	3.82(-7)	1.90(-7)	9.87(-8)	0.0667	171
	5.0	1.59(-8)	1.56(-8)	8.84(-7)	0.0189	312
10^{-8}	0.1	6.26(-10)	6.69(-10)	4.56(-10)	0.0174	176
	0.2	6.72(-10)	7.00(-10)	1.29(-7)	0.0174	177
	0.5	4.02(-10)	2.66(-9)	3.90(-8)	0.0179	255
	1.0	3.67(-9)	4.16(-8)	2.21(-8)	0.0197	329
	2.0	7.78(-9)	6.61(-9)	1.17(-8)	0.0160	490
	5.0	1.00(-9)	4.15(-9)	3.08(-9)	0.0057	823

$a_{max} = 12.9$, $a_p \simeq 17\%$, $a_{p5} \simeq 4\%$.

TABLE 20

EXPFIT2 for problem(ii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	2 35(-5)	2.88(-5)	1.10(-5)	0.1000	26
	0.2	2.82(-5)	4.39(-5)	2.44(-5)	0.1000	26
	0.5	1.09(-4)	9.36(-5)	7.43(-5)	0.1000	27
	1.0	1.86(-5)	2.38(-5)	5.80(-5)	0.1000	32
	2.0	2.60(-5)	4.15(-5)	3.68(-5)	0.1000	38
	5.0	1.80(-5)	7.68(-6)	1.43(-5)	0.1000	57
10^{-4}	0.1	4.94(-6)	4.66(-6)	1.09(-6)	0.1000	34
	0.2	5.45(-6)	5.87(-6)	3.23(-6)	0.1000	35
	0.5	1.53(-5)	6.22(-6)	1.32(-5)	0.1000	36
	1.0	2.07(-6)	1.52(-6)	5.09(-6)	0.1000	49
	2.0	1.91(-6)	2.79(-6)	2.12(-6)	0.1000	66
	5.0	1.54(-6)	7.45(-7)	9.14(-7)	0.0650	105
10^{-6}	0.1	2.16(-7)	1.49(-7)	1.29(-7)	0.0782	78
	0.2	2.31(-7)	2.03(-7)	1.07(-7)	0.0790	81
	0.5	5.08(-7)	2.71(-7)	3.30(-7)	0.1000	81
	1.0	1.62(-7)	1.16(-7)	7.49(-7)	0.1000	107
	2.0	1.48(-7)	2.55(-7)	1.49(-7)	0.1000	157
	5.0	2.55(-8)	1.27(-8)	2.07(-8)	0.0182	294
10^{-8}	0.1	2.41(-9)	1.58(-9)	2.03(-9)	0.0248	206
	0.2	2.55(-9)	1.92(-9)	2.51(-8)	0.0251	212
	0.5	5.50(-9)	2.02(-9)	1.93(-8)	0.0274	247
	1.0	2.35(-9)	1.93(-9)	5.21(-9)	0.0278	331
	2.0	5.27(-9)	1.18(-8)	8.90(-9)	0.0193	469
	5.0	1.24(-8)	6.33(-9)	3.80(-9)	0.0055	788

$a_{max} = 2.51$, $a_p \triangleq 17\%$, $a_{p5} = 0$

TABLE 21

EXPFIT2 for problem (iii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	1.04(-5)	4.62(-6)	2.30(-6)	0.1000	24
	0.2	1.07(-5)	4.82(-6)	2.12(-6)	0.1000	25
	0.5	1.04(-5)	3.14(-5)	9.43(-5)	0.1000	27
	1.0	3.88(-5)	3.39(-5)	6.12(-5)	0.1000	31
	2.0	1.77(-5)	2.54(-5)	3.42(-5)	0.1000	42
	5.0	8.17(-6)	1.55(-5)	1.18(-5)	0.1000	58
10^{-4}	0.1	2.72(-6)	1.37(-6)	6.92(-7)	0.1000	31
	0.2	2.78(-6)	1.29(-6)	8.08(-7)	0.1000	31
	0.5	2.89(-6)	1.35(-5)	6.81(-6)	0.1000	37
	1.0	3.89(-6)	1.20(-5)	1.15(-5)	0.1000	45
	2.0	8.55(-6)	7.84(-6)	4.76(-6)	0.1000	63
	5.0	1.64(-6)	2.07(-6)	1.80(-6)	0.0684	107
10^{-6}	0.1	1.68(-7)	8.35(-8)	2.16(-8)	0.0497	73
	0.2	1.70(-7)	7.18(-8)	5.22(-7)	0.0498	74
	0.5	1.74(-7)	2.89(-7)	3.64(-6)	0.0506	92
	1.0	3.37(-7)	2.16(-7)	2.11(-7)	0.0530	123
	2.0	1.40(-7)	8.37(-8)	6.16(-8)	0.0538	167
	5.0	1.05(-7)	1.15(-7)	6.63(-8)	0.0186	405
10^{-8}	0.1	2.77(-9)	1.17(-9)	5.81(-10)	0.0156	189
	0.2	2.76(-9)	1.21(-9)	9.07(-8)	0.0157	193
	0.5	2.84(-9)	3.21(-9)	1.58(-8)	0.0159	261
	1.0	2.56(-9)	1.82(-8)	1.25(-8)	0.0167	338
	2.0	1.67(-9)	5.16(-9)	9.00(-9)	0.0169	476
	5.0	5.58(-10)	3.82(-9)	3.37(-9)	0.0056	841

$a_{max} = 9.07$, $a_p \simeq 17\%$, $a_{p5} \simeq 4\%$

TABLE 22

NAGMOD for problem (i)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	9.88(-9)	4.89(-9)	1.98(-9)	0.606(-3)	254
	0.2	1.05(-8)	5.16(-9)	2.55(-9)	0.606(-3)	326
	0.5	5.50(-8)	1.59(-7)	7.85(-5)	0.606(-3)	206
	1.0	1.43(-8)	7.03(-8)	9.43(-5)	0.607(-3)	302
	2.0	2.96(-8)	1.14(-4)	1.28(-4)	0.607(-3)	376
	5.0	8.60(-5)	4.26(-5)	3.58(-5)	0.304(-3)	708
10^{-4}	0.1	7.91(-10)	6.03(-10)	4.07(-10)	0.761(-4)	285
	0.2	2.29(-10)	1.32(-10)	1.95(-9)	0.761(-4)	297
	0.5	1.05(-9)	1.80(-8)	5.07(-8)	0.380(-4)	298
	1.0	2.27(-9)	1.95(-5)	5.97(-5)	0.380(-4)	368
	2.0	7.99(-6)	3.93(-5)	1.97(-5)	0.380(-4)	464
	5.0	8.19(-7)	9.70(-6)	1.54(-5)	0.381(-4)	802
10^{-6}	0.1	1.43(-8)	9.42(-9)	3.02(-9)	0.596(-6)	318
	0.2	7.07(-10)	9.13(-10)	1.50(-8)	0.596(-6)	392
	0.5	6.40(-10)	2.40(-8)	4.40(-8)	0.596(-6)	364
	1.0	2.65(-10)	5.01(-8)	1.20(-7)	0.596(-6)	478
	2.0	2.35(-8)	3.28(-8)	1.09(-7)	0.596(-6)	624
	5.0	5.13(-8)	2.55(-8)	1.27(-8)	0.298(-6)	1262
10^{-8}	0.1	*	*	*		
	0.2	*	*	*		
	0.5	*	*	*		
	1.0	*	*	*		
	2.0	*	*	*		
	5.0	*	*	*		

$a_{max} = 0.597$, $a_p = 0 = a_{p5}$

* IFAIL = 1

TABLE 23

NAGMOD for problem(ii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	2.06(-9)	1.05(-9)	1.76(-9)	0.604(-3)	306
	0.2	1.13(-8)	5.80(-7)	3.28(-6)	0.604(-3)	226
	0.5	3.62(-9)	5.85(-8)	3.76(-5)	0.605(-3)	256
	1.0	2.95(-8)	9.78(-6)	2.65(-4)	0.606(-3)	269
	2.0	8.88(-6)	9.63(-5)	1.21(-4)	0.303(-3)	362
	5.0	1.50(-4)	1.30(-4)	1.14(-4)	0.304(-3)	687
10^{-4}	0.1	6.95(-12)	5.94(-10)	9.26(-9)	0.380(-4)	316
	0.2	1.42(-7)	6.83(-7)	3.63(-6)	0.380(-4)	286
	0.5	3.55(-9)	3.99(-7)	3.45(-5)	0.380(-4)	310
	1.0	4.68(-11)	5.54(-7)	7.03(-5)	0.380(-4)	421
	2.0	4.43(-6)	1.20(-5)	6.00(-6)	0.380(-4)	461
	5.0	6.61(-6)	8.64(-6)	1.17(-5)	0.381(-4)	808
10^{-6}	0.1	7.60(-10)	1.69(-8)	5.32(-8)	0.595(-6)	388
	0.2	6.72(-13)	7.24(-10)	3.55(-7)	0.595(-6)	382
	0.5	3.42(-11)	1.65(-9)	1.24(-7)	0.596(-6)	374
	1.0	1.01(-10)	3.04(-7)	2.41(-7)	0.596(-6)	487
	2.0	4.13(-8)	5.45(-8)	5.37(-8)	0.596(-6)	625
	5.0	1.52(-8)	7.60(-9)	5.46(-9)	0.298(-6)	1279
10^{-8}	0.1	*	*	*		
	0.2	*	*	*		
	0.5	*	*	*		
	1.0	*	*	*		
	2.0	*	*	*		
	5.0	*	*	*		

$a_{max} = 0.703$, $a_p = 0 = a_{p5}$

* IFAIL = 1

TABLE 24

NAGMOD for problem (iii)

EPS	k	d = 5	d = 10	d = 20	H	N
10^{-3}	0.1	1.89(-8)	1.00(-8)	4.87(-9)	0.607(-3)	228
	0.2	1.87(-8)	1.06(-8)	5.78(-9)	0.607(-3)	222
	0.5	1.80(-8)	9.11(-9)	8.59(-6)	0.607(-3)	222
	1.0	1.71(-8)	2.12(-5)	1.84(-4)	0.607(-3)	250
	2.0	1.61(-5)	2.09(-4)	2.70(-4)	0.303(-3)	323
	5.0	8.13(-5)	4.03(-5)	2.92(-5)	0.304(-3)	673
10^{-4}	0.1	8.29(-9)	1.04(-8)	9.66(-9)	0.380(-4)	264
	0.2	8.92(-9)	1.59(-8)	7.16(-8)	0.380(-4)	250
	0.5	1.30(-10)	3.89(-8)	3.39(-6)	0.380(-4)	308
	1.0	8.35(-8)	2.12(-5)	4.92(-5)	0.380(-4)	326
	2.0	6.40(-6)	2.62(-5)	1.37(-5)	0.380(-4)	426
	5.0	6.14(-6)	9.15(-6)	9.49(-6)	0.381(-4)	794
10^{-6}	0.1	7.42(-13)	1.78(-11)	3.11(-10)	0.596(-6)	392
	0.2	6.98(-11)	1.10(-9)	2.97(-8)	0.596(-6)	366
	0.5	1.34(-13)	5.73(-11)	8.38(-9)	0.596(-6)	446
	1.0	1.44(-9)	1.08(-7)	5.14(-8)	0.596(-6)	479
	2.0	8.08(-8)	6.36(-8)	1.20(-7)	0.596(-6)	569
	5.0	2.65(-8)	1.32(-8)	8.99(-9)	0.596(-6)	1226
10^{-8}	0.1	*	*	*		
	0.2	*	*	*		
	0.5	*	*	*		
	1.0	*	*	*		
	2.0	*	*	*		
	5.0	*	*	*		

$a_{max} = 0.492, \quad a_p = 0 = a_{p5}$

* IFAIL = 1

TABLE 25

Problem (i)

E	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD	PUBLISHED * RESULT
1.0	0	1.274	1.275	1.275	1.275	1.275
4.0	0	0.834	0.834	0.834	0.834	0.834
9.0	0	0.645	0.645	0.645	0.645	0.645
16.0	0	0.536	0.536	0.534	0.536	0.536
25.0	0	0.463	0.463	0.463	0.463	0.463
0.16	1	0.0146	0.0146	0.0146	0.0146	0.0147
0.25	1	0.0260	0.0260	0.0260	0.0260	0.0260
0.5	1	0.0584	0.0584	0.0584	0.0584	0.0584
0.8	1	0.0924	0.0924	0.0924	0.0924	0.0924
0.16	2	0.0005	0.0005	0.0005	0.0005	0.0005
0.25	2	0.0014	0.0014	0.0014	0.0014	0.0014
0.5	2	0.0055	0.0056	0.0055	0.0056	0.0056

Values of tan δ_0, δ_1, δ_2

*L = 0 : Moiseiwitsch and O'Brien, 1970 (p. 194, Table 1)

L = 1,2: Burke and Smith, 1962 (p. 472, Table 1).

TABLE 26

Problem (i)

E	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD
1.0	0	0.022	0.038	0.050	0.142
4.0	0	0.036	0.060	0.059	0.147
9.0	0	0.042	0.082	0.064	0.199
16.0	0	0.058	0.119	0.062	0.233
25.0	0	0.066	0.152	0.085	0.285
0.16	1	0.025	0.029	0.046	0.185
0.25	1	0.026	0.030	0.039	0.139
0.5	1	0.030	0.033	0.043	0.145
0.8	1	0.032	0.036	0.039	0.147
0.16	2	0.034	0.042	0.051	0.145
0.25	2	0.031	0.039	0.060	0.160
0.5	2	0.030	0.037	0.044	0.156

CPU time in seconds to calculate δ_0, δ_1, δ_2.

TABLE 27

Problem (ii).

k	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD	PUBLISHED* RESULT
0.5	0	2.770	2.770	2.770	2.770	2.745
0.831	0	2.224	2.224	2.224	2.224	2.219
1.225	0	1.808	1.808	1.806	1.808	1.812
1.803	0	1.432	1.432	1.432	1.432	1.437
2.345	0	1.208	1.208	1.208	1.208	1.208

Values of δ_0

* Mott and Massey, 1965 (p. 123, Table b).

TABLE 28

Problem (ii)

k	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD
0.5	0	0.032	0.035	0.054	0.176
0.831	0	0.035	0.044	0.067	0.187
1.225	0	0.045	0.070	0.060	0.205
1.803	0	0.065	0.109	0.082	0.201
2.345	0	0.086	0.159	0.118	0.302

CPU time in seconds to calculate δ_0

TABLE 29

Problem (iii)

k	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD	PUBLISHED* RESULT
0.5	0	8.4463	8.4485	8.4463	8.4463	8.4276
1.0	0	1.9285	1.9286	1.9252	1.9286	1.9284
2.0	0	1.0052	1.0054	1.0054	1.0054	1.0054
3.0	0	0.73780	0.73782	0.73782	0.73781	0.73781
4.0	0	0.59870	0.59881	0.59856	0.59881	0.59880
5.0	0	0.51046	0.51058	0.51052	0.51058	0.51057
1.0	1	0.24792	0.24793	0.24789	0.24793	0.24793
2.0	1	0.33453	0.33456	0.33454	0.33455	0.33455
3.0	1	0.32778	0.32788	0.32788	0.32788	0.32788
4.0	1	0.30614	0.30625	0.30620	0.30625	0.30624
5.0	1	0.28386	0.28394	0.28387	0.28393	0.28393

Values of $\tan \gamma_0$, $\tan \gamma_1$

* Holt and Santoso, 1972 (p. 505, Tables 4, 5).

TABLE 30

Problem (iii)

k	L	RADISH	NUMEROV2	EXPFIT2	NAGMOD
0.5	0	0.024	0.033	0.051	0.153
1.0	0	0.032	0.044	0.048	0.169
2.0	0	0.038	0.078	0.065	0.191
3.0	0	0.057	0.123	0.076	0.218
4.0	0	0.080	0.169	0.082	0.302
5.0	0	0.094	0.214	0.097	0.346
1.0	1	0.039	0.050	0.054	0.179
2.0	1	0.056	0.079	0.074	0.180
3.0	1	0.068	0.121	0.105	0.219
4.0	1	0.079	0.139	0.100	0.293
5.0	1	0.104	0.180	0.119	0.323

CPU time in seconds to calculate γ_0, γ_1.

CHAPTER 8

Numerical solution of coupled homogeneous second-order differential equations

Introduction

We are interested in solving a set of coupled homogeneous second order differential equations of the form (1.12). The method of de Vogelaere (1955) has been used in program SCAT by Chandra (1973) which is generally available, from C.P.C. Program Library, to users to integrate the coupled equations

$$\frac{d^2}{dr^2} U_i = \sum_{j=1}^{NP} \left[\frac{l_i(l_i+1)}{r^2} \delta_{ij} - k_i^2 \delta_{ij} + 2V_{ij} \right] U_j + \sum_{\beta=1}^{NBND} R_{i\beta} \lambda_\beta, \quad i=1,---,NP \quad (8.1)$$

where, in the notation of Chandra, U_i is the radial wave function of the projectile in the ith scattering channel, r is the independent variable, NP is the number of coupled equations and NBND is the number of inhomogeneous terms $R_{i\beta}$ in each channel which arise from the inclusion of exchange terms in the model of the scattering process; the Lagrange multipliers λ_β ensure that the continuum wave function of the scattered electron is orthogonal to all the wave functions of the occupied orbitals of the target system. (see Burke and Chandra, 1972).

We describe in § 8.1 Chandra's program SCAT which uses the method of de Vogelaere to solve (8.1) over a predetermined mesh and in § 8.2 we shall consider the solution of (8.1) with $R_{i\beta} = 0$, $\forall i,\beta$ using a modification of SCAT which uses de Vogelaere's method with a variable steplength. We describe in § 8.3 the test runs which were performed using SCAT and its modification CHMOD. The test results are given in § 8.4 and finally in § 8.5 we consider some additional modifications to program SCAT which might be desirable.

§ 8.1. Chandra's program SCAT

SCAT solves (8.1) over the range [o, r_a] by use of outward integ-

ration over the range $[o, r_o]$ (Chandra's region 1) and inward integration over the range $[r_o, r_a]$ (Chandra's region 2) where r_o is specified by the user and is such that the sum of the short-range attractive potential terms and of k^2 is comparable with the centrifugal barrier term in (8.1). If we assume that there are NA open channels so that there are NB = NP - NA closed channels, SCAT generates in region 1 a (NP x NTOT1)-dimensional solution matrix F^1 of hopefully linearly independent solutions where NTOT1 = NP+NBND by integrating outwards NP different times the equations

$$\sum_{k=1}^{NP} L_{ik} F^1_{kj} = 0, \; i, j = 1, ---, NP \tag{8.2a}$$

and

$$\sum_{k=1}^{NP} L_{ik} F^1_{k,NP+\beta} = R_{i\beta}, \; i=1, ---, NP, \; \beta=1, ---, NBND \tag{8.2b}$$

subject respectively to the following boundary conditions at XX the first mesh point

$$F^1_{ij}(XX) = \delta_{ij}(XX)^{\ell_i+1}, \quad \frac{d}{dr} F^1_{ij}\Big|_{r=XX} = \delta_{ij}(\ell_i+1)(XX)^{\ell_i}, i,j=1,---,NP \tag{8.3a}$$

and

$$F^1_{i,NP+\beta}(XX) = 0 = \frac{d}{dr} F^1_{i,NP+\beta}\Big|_{r=XX}, \; i = 1, ---, NP, \; \beta=1,---,NBND \tag{8.3b}$$

where

$$L_{ij} \equiv \left[\frac{d^2}{dr^2} - \frac{\ell_i(\ell_i+1)}{r^2} + k_i\right]\delta_{ij} - 2V_{ij} .$$

If this outward integration were continued into region 2 the solution would contain components of $e^{|k_i| r}$, i = NA + 1, ---, NP in the closed channels and for large values of the independent variable r these terms would dominate the physical solutions which are oscillatory in nature. Thus SCAT generates in region 2 a (NP x NTOT2) - dimensional solution matrix F^2 of linearly independent solutions, where NTOT2 =

NP + NA + NBND, by integrating inward NP times the equations

$$\sum_{k=1}^{NP} L_{ik} F^2_{kj} = 0, \quad i = 1, \text{---}, NP, \quad j = 1, \text{---}, NP + NA \qquad (8.4a)$$

and

$$\sum_{k=1}^{NP} L_{ik} F^2_{k,\ NP+NA+\beta} = R_{i\beta}, \quad i = 1, \text{---}, NP, \quad \beta = 1, \text{---}, NBND \qquad (8.4b)$$

subject respectively to the following boundary conditions at r_a

$$F^2_{ij}(r_a) = \mathcal{F}_{ij}(r_a), \quad \frac{d}{dr} F^2_{ij}\Big|_{r=r_a} = \frac{d}{dr}\mathcal{F}_{ij}\Big|_{r=r_a}, \quad i=1,\text{---},NP, j=1,\text{---},NP+NA \qquad (8.5a)$$

and

$$F^2_{i,\ NP+NA+\beta}(r_a)=0= \frac{d}{dr} F^2_{i,NP+NA+\beta}\Big|_{r=r_a}, i=1,\text{---},NP, \beta=1,111,\text{---},NBND \qquad (8.5b)$$

where $\mathcal{F}_{ij}$ are a linearly independent set of 2NP solutions of the asymptotic form of (8.1) (see Chandra, 1973) and these are generated by program ASYM (Norcross, 1969), the main subroutine of which is used in conjunction with program SCAT to solve (8.1).

The solutions in regions 1 and 2 must now be matched at the point r_o so that the solutions and their first derivatives are continuous at r_o. Then

$$U_{ij} = \left\{ \begin{array}{ll} \sum_{k=1}^{NTOT1} F^1_{ik} a_{kj} & , \quad 0 \le r \le r_o \\ \sum_{k=1}^{NTOT2} F^2_{ik} a_{NTOT1+k,j} & , \quad r_o \le r \le r_a \end{array} \right\} \begin{array}{l} i = 1,\text{---},NP \\ j = 1,\text{---},NA \end{array} \qquad (8.6)$$

gives a complete set of NA linearly independent solutions for each channel. The coefficients a_{ij}, i = 1, ---, NTOT1, NTOT+1, ---, NTOT1+NTOT2, j=1, ---, NA are obtained by solving a set of inhomogeneous linear simultaneous algebraic equations (given in the matrix form in Table 1 of Chandra, 1973, p. 424) and the reactance and cross section matrices thus obtained.

We shall describe the routines used by program SCAT, in particular routine DEVGL, which performs the numerical integration of (8.1) using de Vogelaere's method over a predetermined mesh. The program in its published form is limited by the dimensions of the arrays to six coupled channels each with three inhomogeneous terms.

§ 8.1.1 Master driver

The numerical integration is performed over a predetermined mesh; regions 1 and 2 are each made up of a number of intervals and as the integration crosses each interval the steplength is changed by a predetermined factor. The input data relating to the choice of mesh points is read by the master driver and the full set of input data read by the master driver for each problem is as follows:

IBUG(I), I = 1, ---, 14 — This array is responsible for the printing of intermediate results obtained during the calculation; if IBUG(I)=0 $\forall$ I then no intermediate results are printed.

HX — The initial steplength supplied by the user.

XX — The value of the first mesh point supplied by the user.

RA — The inward integration is started at r_a; program ASYM calculates the solutions at RA.

NIX1 — The number of intervals in region 1.

NIX — Total number of intervals in the whole range (if NIX1 = NIX no inward integration is performed and RA is the matching point).

IHX(I), I=1, ---, NIX — This array stores the integral multiples of HX; the steplength in the ith interval is HX * IHX(I).

IRX(I), I = 1, ---, NIX — This array stores the cumulative number of mesh points for each interval. IRX(I) should be even.

DELE — This parameter controls the methods of solution to be used by program ASYM.

MAXP — The maximum value of k in the asymptotic expansion of the potential:

$$V_{ij} = \sum_{k=1}^{MAXP} C_{ij}^{(k)} \; r^{-k-1} \qquad (8.7)$$

INPHS — A logical variable which controls the inclusion of a term in the asymptotic solution at RA (see Chandra, 1973, p. 420).

NP — Total number of coupled equations.

NA — Number of open channels.

NBND — Number of bound-state orbitals giving rise to inhomogeneous terms.

LP(I), I = 1, ---, NP — This array stores the angular momenta for NP equations.

IRFN — ≤ 0 signifies that the linearly independent solutions are not to be combined.
>0 signifies that the solutions are to be combined to form the final radial scattering wave functions.

IPH — = 1 signals that the next set of data should be read in
= 2 signals the end of the input data.

NPL — = 0 signals that the CN and VV matrices for the potential should be read in.

WP(I), I = 1, ---, NP — This array stores the energy values in atomic units for NP equations, first for NA open

channels and then for NB=NP-NA closed channels.

ZZ — The net charge.

CN(I,J,K), I=1, ---, NP; J=1, ---, NP; K=1, ---, MAXP — This array stores the coefficients of r^{-k-1} in the asymptotic expansion of the potential given by (8.7).

VV(I,J), I=1, ---, $\frac{NP(NP+1)}{2}$; J=1, ---, IRA2 — This array stores the upper triangular part of the symmetric coupling potential matrix $2V_{ij}$. I is the sequence number of the upper triangular element of V_{ij} starting from the first row, and J is the index number of the mesh point (including the half intervals) at which the potential is evaluated; IRA = 2*IRX(NIX).

BND(I,J,K), I=1, ---, NBND; J=1, ---, NP; K=1, ---, IRA2 — If NBND $>$ 0 the array stores the value of the jth channel wave function of the ith bound-state orbital at the kth mesh point (including half intervals).

The above data is transferred to the routines of program SCAT in the main via a number of common blocks (see listing of program SCAT) and also through the arguments of SCATER which is called by the master driver. In particular the common block BLCK1 transfers the mesh point data, that is HX, NIX1, IHX, IRX, IRA=IRX(NIX), IRA2=2*IRA, RA, to various routines in SCAT. BLCK2 transfers the data specifying the differential equations, namely NP, NA, NBND, LP(I), WP(I) and BLCK3 contains VV(I,J), CN(I,J,K), ZZ along with IRT(I,J) which is a symmetric matrix which stores the sequence number of the upper triangular elements of the potential matrix (starting from the first row). Details of all the common blocks used may be found in SCAT.

§ 8.1.2 Subroutine SCATER

This is the main routine whose arguments are INPHS, DELE, MAXP, XX, IRFN. After printing the input data and checking for the compatability of the mesh point data SCATER calls routine INTEG which in turn calls INTOUT to perform the outward integration in region 1 and routines ASYSM and INTIN to calculate the asymptotic boundary conditions for the solutions at RA and to perform the inward integration in region 2 respectively. With the completion of the numerical integration routine SCATER then calls routines OVRLP (if NBND > 0), MATCH, XSEC and RADFUN (if IRFN > 0). These routines will be described in later Sections.

§ 8.1.3 Subroutine INTEG

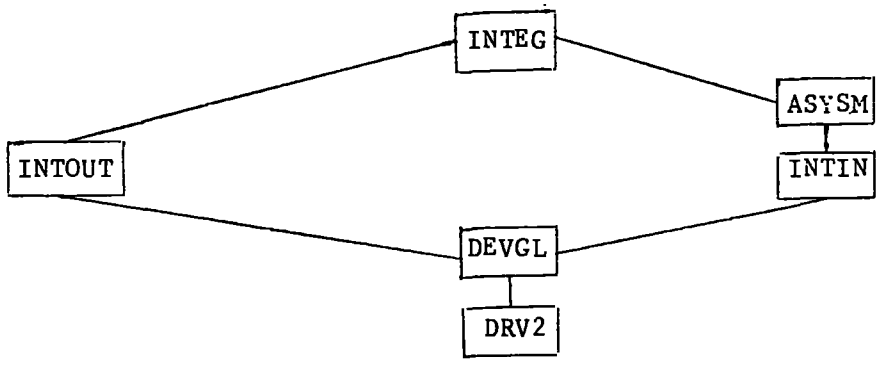

Figure 7

Figure 7 above shows the various routines called by routine INTEG and their relation to one another. The arguments in the call to routine INTEG are INPHS, DELE, MAXP, XX; the latter is the only argument in the call to INTOUT which integrates the equations (8.1) from XX the first mesh point to the matching point r_o subject to the initial boundary conditions given by (8.3) and assuming that the solution is regular at the origin. De Vogelaere's method is used to generate NP+NBND linearly independent solutions for each channel over a predetermined mesh. Consider for example test run no.2 of

Chandra's paper which uses the following input data:

HX = 0.02585 = XX, RA = 7.238

NIX1 = 1 NIX = 3

IHX(I) = 1 2 4

IRX(I) = 80 120 150

NP = 2, NA = 2, NBND = 1

LP(I) = 1 3

WP(I) = 0.5 0.5

Then INTOUT integrates (8.1) over region 1 which comprises 79 steps of length HX over the interval [XX, XX + 79(HX)] = [0.02585, 2.068]. The solutions are regular at the origin and use is made of this fact in subroutine SIMPSN (see 8.1.5). INTOUT uses the method of de Vogelaere in routine DEVGL with constant steplength to perform the numerical integration using method (b) of starting (see § 2.1) with $x_{-1} = XX - \frac{HX}{2}$. Routine DEVGL will be discussed in greater length in §8.1.4. For each of the NTOT1 boundary conditions INTOUT integrates (8.1) for each of the NP equations; once the initial step of the de Vogelaere algorithm has been computed in INTOUT routine DEVGL is called repeatedly (79 times in all) to advance the solution one step of length HX until the integration has reached the end point of region 1. Notice that the values of the potential and the second derivative of the solution are required at equal intervals of length HX in [XX, r_o] and also at the half intervals; the potential values are supplied as data by the user and the second derivatives are calculated by routine DRV2. The mesh points are stored in an array W(K), K=1, ---, IRX(NIX1) and the corresponding solutions are stored in the array FUN(I,K), I=1,---,NP,K=1,---IRX(NIX1). Both arrays are written on disc and are held in the common block BLCK7. The solutions at r_o and the corresponding first derivatives are stored in the FB1 and

FDR1 matrices in the common block BLCK8.

Subroutine ASYSM calculates sets of 2NP solutions and their derivatives at RA; these are stored respectively in the FX and FX1 matrices in common block BLCK5.

Routine INTIN whose only argument is MAXP is then called to integrate (8.1) over region 2 from RA to the matching point r_o subject to the initial boundary conditions at RA, as calculated by ASYSM . INTIN uses the method of de Vogelaere with constant steplength to integrate (8.1) for each of the NTOT2 boundary conditions; the initial step uses method (b) of starting where x_{-1} in the case of test run no. 2 of Chandra's paper is $RA + \frac{4HX}{2}$. Region 2 is made up of two intervals the first of which, [4.136, 7.238], uses a steplength of -4HX and the second, [2.068, 4.136], uses a steplength of -2HX until the integration has been advanced to r_o = 2.068 the end point of region 2. The first step is calculated in INTIN and thereafter DEVGL is called to generate the numerical solution at each mesh point. The mesh points and solutions of region 2 are again stored in the W and FUN arrays respectively and the solutions and their corresponding first derivatives at r_o are stored in the FB2 and FDR2 matrices respectively in the common block BLCK9. Note that if NIX1=NIX no inward integration is performed and RA becomes the matching point.

§ 8.1.4 Subroutines DEVGL and DRV2

The following arguments appear in calls to routine DEVGL:

H — The current value of the steplength

HS — H*H

H1 — The previous value of the steplength after a change in steplength

LSWT — = 1, signifies that the computation should proceed for the given H

= 2, signifies that a change in steplength has occurred

and that the computation should proceed with the modifications which are required to take account of this.

K2 The index number of the bound-state orbital which is to be used to generate the K2-th inhomogeneous solution.

K31, K32 The index numbers of the mesh points corresponding to the mid and end points of the particular step; suppose the step is of length H starting from the point X. Then K31, K32 corresponds to the index numbers of the points $X+\frac{H}{2}$, X+H respectively.

The routine is written to perform the numerical integration of (8.1) over one step of length H using de Vogelaere's method; each step is effected by a separate call to DEVGL and the user emerges from routines INTOUT and INTIN having completed the numerical integration over regions 1 and 2 respectively. If on entry to DEVGL, LSWT=1 then DEVGL uses essentially equations (3.13) - (3.20) of §3.1 with H, HH, H2 replaced by $\frac{H}{2}$, H, $\frac{HS}{4}$ respectively to advance the solutions one step. If LSWT=2 a change in steplength has occurred and values for the next step must be updated; the factor by which the steplength has changed is given by C = H/H1. No attempt is made to estimate the truncation error in the numerical calculation and the solutions are calculated at a set of predetermined mesh points.

The second derivatives of the solutions are calculated using routine DRV2. Suppose the solution is to be advanced from the mesh point X to X+H. Then the second derivative of the solutions at the intermediate point $X + \frac{H}{2}$ is determined by a call to routine DRV2 whose arguments are K2 and K31. Similarly for the point X + H the arguments used in the calling sequence of DRV2 are K2 and K32.

§ 8.1.5 Subroutines OVRLP and SIMPSN

Routine OVRLP is called by SCATER only when NBND>0 to initialise the

parameters for the calculation of the so called 'overlap integrals' which are necessitated by satisfaction of the following orthogonality condition (see Chandra, 1973)

$$\sum_{k=1}^{NTOT1} I^1_{\beta k}\, a_{kj} + \sum_{k=1}^{NTOT2} I^2_{\beta k}\, a_{NTOT1+k,j} = 0, \quad j=1, \text{---}, NA, \quad \beta=1, \text{---}, NBND$$

where

$$I^1_{\beta k} = \sum_{i=1}^{NP} \int_o^{r_o} R_{i\beta} F^1_{ik} dr, \quad k=1, \text{---}, NTOT1, \quad \beta=1, \text{---}, NBND$$

$$I^2_{\beta k} = \sum_{i=1}^{NP} \int_{r_o}^{r_a} R_{i\beta} F^2_{ik} dr, \quad k = 1, \text{---}, NTOT2, \quad \beta=1, \text{---}, NBND.$$

The values of $R_{i\beta}$ are stored in the BND array for each of the IRA2 mesh points and the solutions F^1_{ik} and F^2_{ik} are stored in the array FUN. I^1 and I^2 are the overlap integrals of the NTOT1 linearly independent solutions of region 1 and NTOT2 solutions of region 2 with the NBND bound-state orbitals. Routine OVRLP calls routine SIMPSN to calculate I^1 and I^2 which are stored in the arrays OVLP1(I, J,), OVLP2(I, K), I = 1, ---, NBND, J = 1, ---, NTOT1, K = 1, ---, NTOT2, which together with the BND array are stored in the common block BLCK4.

Subroutine SIMPSN uses Simpson's one-third rule to evaluate the integrals I^1 and I^2. For test run no. 2 of Chandra's paper the first call to SIMPSN by OVRLP calculates I^1_{11}, I^1_{12}, I^1_{13} for region 1 using the solutions at the origin together with those stored in the FUN array at the next 80 mesh points. Similarly the second call to SIMPSN calculates I^2_{11}, I^2_{12}, I^2_{13}, I^2_{14}, I^2_{15} for region 2.

§ 8.1.6 Subroutines MATCH, XSEC and RADFUN

These routines are called by SCATER in turn. MATCH sets up the matching equations at the point r_o (see Chandra, 1973, p. 424) and calls the routine MA01A to solve the resultant set of inhomogeneous

linear simultaneous algebraic equations; solving these equations NA times enables the full (NTOT1 + NTOT2) x NA solution matrix for the unknown coefficients a to be obtained. Relations (15), (16) of Chandra (1973, p. 422) may be used in particular to separate and store the NA x NA elements of the reactance matrix K (in the notation of Chandra) in the KMT matrix.

Routine XSEC which is subsequently called by SCATER evaluates the S(scattering), T(transition) and CSM (cross section) matrices of BLCK11 and XSEC makes use of routine MA01A to invert a matrix and routine MLTY to multiply two square matrices.

Subroutine RADFUN is finally called by SCATER if IRFN>0 and the NTOT1 linearly independent solutions of region 1 are linearly combined with the NTOT2 linearly independent solutions of region 2 for each channel to form NA radial scattering functions for NA open channels.

§ 8.2 CHMOD - a modification of Chandra's program.

Chandra's program SCAT is written in FORTRAN IV and may be run in its original form on an ICL 1907 computer. Only very trivial modifications were required in order to run SCAT on the IBM 370/168 computer. The numerical integration stage of any quantum mechanical scattering problem consumes a considerable proportion of the CPU time required to solve the problem in full. Thus any saving in CPU time during the numerical integration stage would certainly be desirable and we consider the effect of replacing routine DEVGL in SCAT by a coupled channel version of routine DEVOG which uses an automatic error control to select steps as large as possible at each step of the numerical integration. The introduction of DEVOG necessitates the calculation of the potential function within the program at the mesh points as they are generated by routine DEVOG; program SCAT by contrast

requires the user to specify a priori the mesh spacing for the numerical integration and in addition the values of the potential and the bound-state orbitals at these points must be supplied as data to the master driver routine of SCAT.

A number of modifications to the routines of SCAT both trivial and non-trivial will be required in order that the modified version CHMOD solves (8.1) automatically. The master driver reads the following data:

IBUG (I), I = 1, ---, 14, DELE, MAXP, INPHS, ZZ, NP, NA, NBND, LP(I), I = 1, ---, NP, IRFN, IPH, WP(I), I = 1, ---, NP

as in program SCAT; in addition HO, XO, RO and RA which now comprise BLCK1 must be supplied and they are respectively the initial steplength for the integration in region 1, the first mesh point in region 1, the matching point and the point at which the inward integration is started. EPS and C must also be supplied by the user and they are the tolerance parameter and the factor by which the steplength is allowed to increase; EPS and C are transferred to routine DEVOG by means of a common block CEPS. The elements of the CN array are also specified and routine SCATER is called.

Routine SCATER is simplified since we discard the compatability check on the mesh point data and we also dispense with the setting up of the IRT matrix. The arguments of SCATER are INPHS, DELE, IRFN, and SCATER calls INTEG, MATCH, XSEC and RADFUN in turn (we are considering the equations to be homogeneous so that routine OVRLP and hence SIMPSN are not required). MATCH, XSEC and RADFUN are used in the same form as in SCAT with only minor changes to the common blocks and to the dimensions of the arrays.

The major changes have been made to the numerical integration routines namely, INTEG, INTOUT, INTIN, DEVGL, DRV2 the first three of which are replaced by INTEG; DEVGL has been replaced by a coupled

channel version of DEVOG, and DRV2 includes the calculation of the potential at the required mesh points.

§ 8.2.1 Numerical integration routines of CHMOD.

The routines INTEG, INTOUT and INTIN of SCAT are replaced by just routine INTEG in CHMOD; INTEG now provides the initial values required for the outward and inward integrations and two calls are made to DEVOG for the integrations in regions 1 and 2. DEVOG has been modified to integrate NP second order coupled differential equations subject to a set of K3 initial boundary conditions over a specified range of integration [XO, XF]. Program SCAT performs the outward and inward integrations within routines INTOUT and INTIN respectively by integrating the NP equations over all the channels for each of K3 initial boundary conditions in turn (K3 = NTOT1, NTOT2 in INTOUT, INTIN respectively). However in order to incorporate an automatic steplength selection it is necessary to perform the integration over all the channels for each of the boundary conditions simultaneously and a large number of the variables of DEVOG as used in program RADISH must be replaced by two dimensional arrays. Routine DEVOG stores the mesh points in the W(K) array and the solutions are stored in FUN(I,J,K) where I ranges over the number of equations, J over the number of boundary conditions and K is the index number of the mesh point including the half interval points. The first call to DEVOG performs the automatic outward integration from XO to RO and the second call performs the automatic inward integration from RA to the matching point RO. DEVOG is written so that the integration ends exactly at the end point of the interval which is specified by the user.

The arguments which must be supplied to routine DEVOG are the initial steplength H (although strictly speaking each step in DEVOG

is of length 2H), K2 the number of inhomogeneous terms, K3 the total number of linearly independent solutions generated over the range of integration and XF the end point of the range of integration. For the sake of simplicity we have taken K2 to be zero; if K2 is non zero then the values of the inhomogeneous terms (corresponding to the values of the BND array in SCAT) at the mesh points generated by DEVOG must also be calculated. K3 is set to be NTOT1 in routine INTEG before the first call to DEVOG is made for the outward integration and on exit K3 is set to be KO the number of steps carried out during the range of integration; the mesh points and the corresponding solutions are then stored in the WO(K) and FUNO(I,J,K) arrays, I = 1, ---, NP, J = 1, ---, K3, K = 1, ---, KO. If in order to reach r_o the end point of the interval of outward integration the steplength must be reduced from h_1 to h_2 say, then the initial steplength for the inward integration is taken to be $-h_1$. Before the second entry to DEVOG K3 is set to be NTOT2 in routine INTEG and on exit K3 is set to be KI, the number of steps carried out in the inward integration from RA to r_o; the mesh points and the corresponding solutions are then stored in the WI(K) and FUNI(I, J, K) arrays respectively, I = 1, ---, NP, J = 1, ---, K3, K = 1, ---, KI.

Routine DRV2 whose arguments are K2, K3 calculates the value of the potential at the current mesh point using the explicit analytic expression; the second derivative of the solution at this point may now be returned to routine DEVOG.

A detailed description of routines DEVOG and DRV2 as used in CHMOD for test run no. 1 of Chandra's paper is provided by comment cards in the listing in Appendix 3.

§ 8.3 Test runs.

(a) Programs SCAT and CHMOD were run for the single channel case

of electron-hydrogen elastic scattering in the static approximation for the range of angular momentum and energy values listed in Table 13 of § 7.3. The integration in both programs was started at XO = 0.01 with an initial steplength of H = 0.01. In the case of program SCAT the steplength was kept fixed throughout the range of integration [XX, RO]; program CHMOD uses the same initial conditions with EPS = 10^{-6} and the steplength is varied automatically so that EPS is the largest allowed error per unit step. The results obtained for the phase shifts using SCAT and CHMOD were compared with the published results in Table 31. Notice that SCAT and CHMOD return the value of the tangent of the phase shift. The effect of varying the matching point was studied for RA = RO = 8, 10, 12, 15 atomic units.

(b) Programs SCAT and CHMOD were run for test run no. 1 in Chandra's paper which calculates the reactance matrix (the K matrix in the notation of Chandra) for electron-hydrogen scattering in the strong-coupling approximation when only the 1s and 2s atomic states are included in the eigenfunction expansion, with exchange neglected. The input data specifying the above problem is as follows for program SCAT:

INPHS = T, DELE = 0.0, MAXP = 1, IRFN = 1

HX = 0.0304572 = XX, RA = 13.70574

NIX1 = 1, NIX = 4

IHX(I) = 1, 2, 4, 8

IRX(I) = 50, 90, 130, 150

NP = 2 = NA, NBND = 0

LP(I) = 0, 0

WP(I) = 0.5, 0.125

The explicit analytic expressions for the potential in (8.1) are

$$2V_{11}(x) = -2\left(1+\frac{1}{x}\right)^{-2x}, \quad 2V_{12}(x) = \frac{4\sqrt{2}}{27}(2+3x)e^{\frac{-3x}{2}}$$

$$2V_{22}(x) = -2\left(\frac{1}{x} + \frac{3}{4} + \frac{1}{4}x + \frac{1}{8}x^2\right)e^{-x}$$

(see Burke and McCarroll, 1962).

Program CHMOD uses the same initial starting point and steplength but the steplength is varied automatically during the outward and inward integrations. CHMOD is run for the above problem for a range of values of EPS = 10^{-n}, n = 3, 4, 5, 6, 8 and the elements of the K matrices thus obtained are compared with the K matrix obtained using SCAT.

§ 8.4 Test results

(a) The phase shifts δ produced by SCAT and CHMOD agree exactly with each other up to the number of figures quoted in the published result and the phase shifts for SCAT and CHMOD are unchanged over the range of values of RO which are tested. Table 31 below shows the values of tan δ_0, δ_1, δ_2 obtained using SCAT, CHMOD along with the published results of Burke and Smith (1962) and Moiseiwitsch and O'Brien (1970). In addition the number of function evaluations performed by CHMOD over the range [XO, RO] is provided for RO = 8, 10.

TABLE 31

E	ℓ	SCAT,CHMOD	PUBLISHED RESULT	NO OF FN EVALS BY CHMOD RO = 8	RO = 10
1.0	0	1.273	1.275	144	184
4.0	0	0.833	0.834	222	272
9.0	0	0.644	0.645	352	438
16.0	0	0.535	0.536	414	514
25.0	0	0.462	0.463	496	616
0.16	1	0.0146	0.0147	112	124
0.25	1	0.0260	0.0260	140	142
0.5	1	0.0584	0.0584	138	160
0.8	1	0.0927	0.0924	162	170
0.16	2	0.0005	0.0005	110	118
0.25	2	0.0014	0.0014	124	138
0.5	2	0.0055	0.0056	136	162

Program SCAT performs 1600 and 2000 function evaluations for each value of E, ℓ tested when RO = 8 and RO = 10 respectively. We thus see a drastic reduction in the number of function evaluations performed by CHMOD when compared with SCAT. For the low energy values ($E < 1$) CHMOD uses less than 10% of the number of function evaluations used by SCAT and even at the high energies this percentage is approximately 30%. Of course the user of SCAT might decide that region 1 should contain more than one interval, for example he might choose to provide the following input data relating to the mesh points:

HX = 0.01, XX = 0.01, RA = 8 = RO

NIX1 = 4

IHX(I) = 1, 2, 4, 8

IRX(I) = 100, 150, 200, 250.

SCAT would then perform 500 function evaluations which is considerably less than the corresponding number required when the initial steplength remains fixed throughout [XO, 8]. With this particular choice however of input data CHMOD will still require considerably less function evaluations than SCAT and this would hopefully result in less CPU time being required for the numerical integration stage of the problem.

(b) This test run provides us with a more convincing case for using CHMOD over SCAT in terms of reducing the CPU time required for the numerical integration; this test run uses a predetermined mesh based on the user's intuition about the approximate behaviour of the solution. The K matrix produced by SCAT is

$$K = \begin{bmatrix} 1.138853 & 3.854753(-1) \\ 3.854697(-1) & -3.256686(-1) \end{bmatrix}$$

and SCAT performs 300 function evaluations during the numerical

integration stage of the problem. Program CHMOD produces the following K matrices for the range of values of EPS:

(i) $EPS = 10^{-3}$, $K = \begin{bmatrix} 1.135174 & 3.848934(-1) \\ 3.845751(-1) & -3.259578(-1) \end{bmatrix}$

(ii) $EPS = 10^{-4}$, $K = \begin{bmatrix} 1.138839 & 3.854565(-1) \\ 3.854490(-1) & -3.256349(-1) \end{bmatrix}$

(iii) $EPS = 10^{-5}$, $K = \begin{bmatrix} 1.138745 & 3.854520(-1) \\ 3.854437(-1) & -3.256742(-1) \end{bmatrix}$

(iv) $EPS = 10^{-6}$, $K = \begin{bmatrix} 1.138980 & 3.854975(-1) \\ 3.854974(-1) & -3.256579(-1) \end{bmatrix}$

(v) $EPS = 10^{-8}$, $K = \begin{bmatrix} 1.138996 & 3.855010(-1) \\ 3.855010(-1) & -3.256568(-1) \end{bmatrix}$

The following table shows the number of function evaluations performed by CHMOD in producing the K matrices of (i) - (v).

EPS	NO. OF FN. EVALS.
10^{-3}	86
10^{-4}	130
10^{-5}	144
10^{-6}	258
10^{-8}	754

The value $EPS = 10^{-8}$ is extremely stringent but we may regard the elements of the K matrix produced by this value of EPS to be 'exact' since CHMOD for the main part of the range of integration uses steplengths which are far smaller than the smallest steplength used by SCAT. For $EPS = 10^{-4}$ the elements of the K matrix produced by CHMOD are in good agreement with those produced by SCAT; moreover the number of function evaluations performed by CHMOD is less than half that performed by SCAT. Even with $EPS = 10^{-6}$ CHMOD requires 86% of the number of function evaluations performed by SCAT. We have emphasised previously that the numerical integration of the differential equations

(8.1) uses a considerable proportion of the CPU time which is required for the solution of the problem and in view of the tremendous saving in terms of the number of function evaluations to be gained using CHMOD in preference to SCAT we would advocate that routine DEVGL be replaced by DEVOG in program SCAT. Of course program CHMOD in its present form is purely a research code and would require a certain amount of fine tuning and optimisation before being made generally available to users as a preferable alternative to SCAT.

§ 8.5 Discussion.

As noted in Chandra (1973) the accuracies of de Vogelaere's algorithm (and in addition Simpson's rule) as used in SCAT depend implicitly upon the mesh point data supplied by the user. We have shown in the previous Section that by incorporating our own routine DEVOG into program SCAT in place of routine DEVGL we are able to produce accurate results in CHMOD while at the same time using a considerably reduced number of function evaluations in the numerical integration stage of the calculation. In addition the potential and bound-state orbitals which were previously supplied by the user to SCAT at a set of predetermined mesh points are now determined as the calculation proceeds at the mesh points which are generated automatically by routine DEVOG in program CHMOD.

We have already noted that program CHMOD is by no means optimised. CHMOD in its present form is limited by the dimensions of the arrays to five coupled channels each with two inhomogeneous terms. However this restriction could be easily lifted and the efficiency of program CHMOD improved with respect to computer storage considerations.

If inhomogeneous terms are to be included in the scattering model

then program CHMOD will require an auxilliary routine which calculates these terms at the required mesh points. In addition a modification of routine SIMPSN in SCAT is necessitated; previously in program SCAT the use of routine SIMPSN required that an even number of steps be performed over each interval of regions 1 and 2 using the de Vogelaere algorithm. With the introduction of a variable stepsize in the algorithm Simpson's one third rule lends itself to the calculation of the overlap integrals; consider the evaluation of the overlap integrals over one step of the range of integration $[x_r, x_{r+1}]$ say, where $x_{r+1} = x_r + h$. Now the de Vogelaere algorithm produces fourth order approximations to the solutions at x_r and x_{r+1} and a third order approximation to the solution at the intermediate point $x_{r+\frac{1}{2}}$. Thus Simpson's rule, which is a fourth order quadrature method, uses the points x_r, $x_{r+\frac{1}{2}}$, x_{r+1} to evaluate the quadrature of the bound-state orbitals with the solutions over the step $[x_r, x_{r+1}]$.

The question of maintaining the linear independence of the solutions generated in regions 1 and 2 is an important one and it would be nice to have some sort of check within CHMOD to ascertain whether this linear independence is violated and furthermore to take appropriate action if this is the case. The problem arises because of the finite word length of computers. A recent article by Scott and Watts (1977) considers the problem of maintaining linear independence in the solutions of linear two-point boundary value problems and a computer code SUPORT (Scott and Watts, 1975) using superposition coupled with an orthonormalisation procedure and a variable -step Runge-Kutta-Fehlberg integration scheme produces the solution of such problems. Consider now the numerical integration in CHMOD. The range of integration is in two parts; CHMOD provides a (2NPxNTOT1) - dimensional matrix $\mathcal{A}$ of initial boundary conditions (at XO) the columns of which are linearly independent

for the outward integration in region 1 and a (2NPxNTOT2) - dimensional matrix β of initial boundary conditions (at RA) the columns of which are also linearly independent for the inward integration in region 2. The corresponding solution matrices F^1 and F^2 are thus generated in regions 1 and 2 and the final solutions are obtained by solving the so called matching equations thus providing NA solutions for each channel. We shall consider the problem of how to maintain the linear independence of the solutions F^1 in region 1. We start the solution using the matrix α of initial conditions and our aim is to ensure that the NTOT1 columns of α remain linearly independent over the range [XO, r_o]; one way to guarantee numerical independence of the solutions is to keep them nearly mutually orthogonal over the range of integration. Scott and Watts (1977) use the modified Gram-Schmidt procedure to orthonormalise the solutions of the homogeneous and particular equations for the particular linear boundary value problem. This method might also be used in Chandra's program to replace the old set of vectors α_{old} by a new orthonormal set α_{new}:

$$\alpha_{old} = \alpha_{new} P$$

where P is an (NTOT1 x NTOT1) upper triangular matrix defined by the modified Gram-Schmidt formulation. Alternatively the matrix α_{old} could be orthogonalised using the Householder and singular value decomposition processes (see Nash, 1973). We shall require a test which determines when and the frequency with which this orthogonalisation process must be performed. Suppose that at some point in the integration in region 1 the columns of α start to lose their numerical linear independence; then if a singular value decomposition of the matrix α were performed at this point (e.g. using the numerical procedure of Golub and Kahan, 1965) the condition would be indicated by the presence of one or more zero (relative to some tolerance) singular

values of α. Thus proceeding in this manner throughout region 1 we would be able to obtain NTOT1 linearly independent solutions at r_o. Similarly in region 2 we would obtain NTOT2 linearly independent solutions at r_o by ensuring that the columns of β throughout region 2 remain linearly independent. The elements of the reactance matrix could then be obtained in the usual way by solving the resultant set of inhomogeneous linear equations. Note that if NBND>0 the overlap integral terms must be evaluated over the various intervals defined by the orthonormalisation points at which the condition of continuity of the solutions must be imposed. The implementation of the orthonormalisation procedure referred to above would of course be expensive in terms of computing time but the advantages and disadvantages of such an implementation would be worthy of investigation. Program CHMOD would be more suitable than would SCAT for the introduction of such a procedure since CHMOD solves the set of coupled equations simultaneously over the number of channels and the number of boundary conditions.

SUGGESTIONS FOR FUTURE WORK

We have studied the ability of programs RADISH, NUMEROV2, EXPFIT2 and NAGMOD which incorporate the methods of de Vogelaere, Numerov, Raptis and Allison with a variable stepsize and a variable-step variable-order Adams method to solve

$$y'' = f(x, y) \tag{1}$$

where

$$f(x, y) = \left[\frac{L(L+1)}{x^2} - E + V(x)\right] y(x) \quad .$$

It is important to note that the conclusions of § 7.3 relate to the relative reliability and efficiency of the programs (as they are written) and not only to the methods themselves to solve equation (1). The work of Chapters 1 - 8 suggests areas for further study which we shall discuss below.

The following fourth order Runge Kutta method

$$\left.\begin{aligned} k_o &= h^2 f(x_o, y_o) \\ k_1 &= h^2 f\left(x_o + \frac{h}{2}, y_o + \frac{h}{2} y_o' + \frac{k_o}{8}\right) \\ k_2 &= h^2 f\left(x_o + h, y_o + hy_o' + \frac{k_1}{2}\right) \\ y_1 &= y_o + hy_o' + \frac{1}{6}(k_o + 2k_1) \\ hy_1' &= hy_o' + \frac{1}{6}(k_o + 4k_1 + k_2) \end{aligned}\right\} \tag{2}$$

solves (1) directly using a total of three function evaluations per step. It would be interesting to include in our numerical comparison of Chapter 7 a program which uses the method in (2) with automatic error control. Unfortunately the method does not possess a natural error estimator. Shampine and Watts (1971) have compared a number of different error estimators for the local truncation error of Runge-Kutta methods; one such estimator is that of England (1969) which for the method in (2) might be formulated as

$$r_1 = \delta_0 k_0 + \delta_1 k_1 + \delta_2 k_2 + \delta_3 k_3 + \delta_4 k_4 + \delta_5 k_5 \qquad (3)$$

with

$$k_3 = h^2 f(x_1, y_1)$$

$$k_4 = h^2 f\left(x_1 + \frac{h}{2}, y_1 + \frac{h}{2} y_1' + \frac{k_3}{8}\right)$$

$$k_5 = h^2 f(x_1 + \alpha h, y_o + \beta h y_o' + \gamma_o k_o + \gamma_1 k_1 + \gamma_2 k_2 + \gamma_3 k_3 + \gamma_4 k_4)$$

where the parameters α, β, γ_i, $i = 0, 1, ---, 4$, δ_i, $i = 0,1,---,5$ are to be chosen so that the error in the solution at x_1 is given by

$$y(x_1) - y_1 = r_1 + O(h^6).$$

It may transpire however that it is not possible to obtain a fourth order method and a fifth order error estimator with six function evaluations. An investigation of the feasibility of using (3) as an estimate of the local truncation error would be instructive albeit lengthy. The general feeling is that Runge-Kutta methods, certainly for systems of first order equations, 'particularly the 4th-order versions, are rather expensive for high-accuracy work, but very competitive with other methods when only a few figures are required' (Walsh, 1974). It would be interesting to see how the method of de Vogelaere compares with the Runge-Kutta method in (2) when low accuracies in the solution are requested.

We have seen in Chapters 6 and 7 that program NAGMOD with an error per unit step criterion in routine DNGMOD does an extremely good job of controlling the global error in the numerical integration stage of the calculation but that it also uses an excessive number of function evaluations in the process. NAGMOD uses routine DNGMOD for the numerical integration and this routine is a modified version of the N.A.G. routine D02AHF which uses a variable order Admas method based on Krogh's algorithm (1973). Routine D02AHF is soon to be withdrawn from the N.A.G. Library and replaced by routine D02QAF which

is based on Hindmarsh's (1974) code GEAR REVISION 3. A comparison of DO2AHF and the GEAR code has been made by Gladwell (1979, private correspondence) and it is reported that the GEAR code generally uses less than half the number of function evaluations of DO2AHF. The GEAR code is particularly well suited for the solution of stiff systems of first order differential equations but contains as an option an automatic implementation of the variable order Adams Bashforth - Adams Moulton methods for the solution of non-stiff problems. The major difference however between DO2AHF and the GEAR code lies in the adopted steplength strategy; the former only changes step by halving or doubling while the latter chooses an optimum value of the steplength when a decrease or increase in steplength is required, and in addition uses interpolation to output the solution at points which are specified by the user. This would probably account for the decrease in the number of function evaluations which are performed. However it might be more appropriate to solve (1) by employing a variable-step variable-order routine which solves (1) directly; we would expect such a routine to be more efficient and we would welcome the design of a code specifically for this purpose.

We discussed in Chapter 8 Chandra's program SCAT to solve a set of coupled homogeneous second order differential equations and a modification thereof in program CHMOD. Several modifications such as those discussed in § 8.5 could be applied to CHMOD with an aim to optimising the program with respect to computer time and storage considerations.

The iterative Numerov method has been studied by Allison (1970) and compared with de Vogelaere's method and the matrix Numerov method. It would be interesting to investigate the performance of the method of Raptis and Allison when incorporated into a matrix and iterative Raptis and Allison method.

REFERENCES

ALLISON, A.C., "The numerical solution of coupled differential equations arising from the Schrödinger equation", J.Comput. Phys., 6, 1970, pp. 378-391.

ASH, J.H., "Analysis of multistep methods for special second-order ordinary differential equations" Ph.D. thesis, University of Toronto, 1969.

BASAVAIAH, S., and BROOM, R.F., "Characteristics of in-line Josephson tunneling gates", IEEE Trans. on Magnetics, MAG-11, 1975, pp. 759-762.

BLATT, J.M., "Practical points concerning the solution of the Schrödinger equation", J.Comput. Phys., 1, 1967, pp. 382-396.

BURKE, P.G., and CHANDRA, N., "Electron-molecule interactions. III. A pseudo-potential method for e^- - N_2 scattering", J.Phys. B: Atom.Molec. Phys., 5, 1972, pp. 1696-1711.

BURKE, V.M., and McCARROLL, R., "Electron scattering by atomic hydrogen in the 1s, 2s or 2p state: II", Proc. Phys. Soc., 80, 1962, pp. 422-431.

BURKE, P.G., and SEATON, M.J., "Numerical solutions of the integrodifferential equations of electron-atom collision theory", Meth. Comput. Phys., 10, 1971, pp. 1 - 80.

BURKE, P.G., and SMITH, K., "The low-energy scattering of electrons and positrons by hydrogen atoms", Rev.Mod.Phys., 34, 1962, pp. 458-502.

CHANDRA, N., "A general program to study the scattering of particles by solving coupled inhomogeneous second-order differential equations", Compute Phys.Commun., 5, 1973, pp. 417-429.

CHANDRASEKHAR, S., and BREEN, F.H., "The motion of an electron in the Hartree field of a hydrogen atom", Astrophys. J., 103, 1946, pp. 41-46.

COLEMAN, J.P., and MOHAMED, J., "On de Vogelaere's method for y''=f(x,y)", Math. Comp., 32, 1978, pp. 751-762.

COLEMAN, J.P., and MOHAMED, J., "De Vogelaere's method with automatic error control", Computer Phys. Commun., 1979, to appear.

CORBATÓ, F.J., and URETSKY, J.L., "Generation of spherical Bessel functions in digital computers", J.Assoc. Comp.Mach., 6, 1959, pp. 366-375.

C.P.C. PROGRAM LIBRARY, Queen's University of Belfast, N.Ireland, 1973.

DE VOGELAERE, R., "A method for the numerical integration of differential equations of second order without explicit first derivatives", J.Res. NBS, 54, 1955, pp. 119-125.

ENGLAND, R., "Error estimates for Runge-Kutta type solutions to systems of ordinary differential equations", Comput.J., 12, 1969, pp. 166-170.

FROESE, C., "Numerical solution of the Hartree-Fock equations", Can.J.Phys., 41, 1963, pp.1895-1910.

GAUTSCHI, W., "Numerical integration of ordinary differential equations based on trigonometric polynomials", Numer.Math., 3, 1961. pp.381-397.

GOLUB, G., and KAHAN, W., "Calculating the singular values and pseudo-inverse of a matrix", SIAM J.Numer.Anal., 2, 1965, pp. 205-224.

GORDON, R.G., "New method for constructing wavefunctions for bound states and scattering", J.Chem. Phys., 51, 1969, pp. 14-25.

HALL, G., and WATT, J.M., Modern Numerical Methods for Ordinary Differential Equations, Clarendon Press, Oxford, 1976.

HENRICI, P., Discrete Variable Methods in Ordinary Differential Equations, Wiley, New York, 1962.

HILDEBRAND, F.B., Introduction to Numerical Analysis, McGraw-Hill, New York, 1956.

HINDMARSH, A.C., "GEAR: ordinary differential equation system solver", Lawrence Livermore Laboratory Report UCID-30001 Rev. 3, Livermore, California, 1974.

HOLT, A.R., and SANTOSO, B., "A Fredholm integral equation method for scattering phase shifts", J.Phys. B: Atom. Molec.Phys., 5, 1972, pp. 497-507.

HULL, T.E., ENRIGHT, W.H., FELLEN, B.M., and SEDGWICK, A.E., "Comparing numerical methods for ordinary differential equations", SIAM J.Numer. Anal., 9, 1972, pp. 603-637.

KOPAL, Z., Numerical Analysis, Chapman and Hall, London, 1955.

KROGH, F.T., "Algorithms for changing the step size", SIAM J.Numer. Anal., 10, 1973, pp.949-965.

LAMBERT, J.D., Computational Methods in Ordinary Differential Equations, Wiley, London, 1973.

LAMBERT, J.D., "The intial value problem for ordinary differential equations: a survey", in

LAMBERT contd.: A Survey of Numerical Analysis 1976, edited by D.A.H. Jacobs, Academic Press, London, 1977.

LAUNAY, J. - M., "Body-fixed formulation of rotational excitation: exact and centrifugal decoupling results for CO - He", J.Phys.B: Atom.Molec.Phys., 9, 1976, pp.1823-1838.

LESTER, W.A., "De Vogelaere's method for the numerical integration of second-order differential equations without explicit first derivatives: application to coupled equations arising from the Schrödinger equation", J.Comput.Phys., 3, 1968, pp.322-326.

LESTER, W.A., "Calculation of cross sections for rotational excitation of diatomic molecules by heavy particle impact: solution of the close-coupled equations", in Methods in Computational Physics, edited by B. Alder, S.Fernbach, and M.Rotenberg, Academic Press, New York and London, Volume 10, 1971, pp. 211-241.

LESTER, W.A., and BERNSTEIN, R.B., "Computational procedure for the close-coupled rotational excitation problem: scattering of diatomic molecules by atoms", J.Chem.Phys., 48, 1968, pp. 4896-4904.

LYCHE, T., "Chebyshevian multistep methods for ordinary differential equations", Numer. Math., 19, 1972, pp. 65-75.

McDOUGALL, J., "The motion of electrons in the static fields of hydrogen and helium", Proc. Roy. Soc. A, 136, 1932, pp. 549-558.

MOISEIWITSCH, B.L., and O'BRIEN, T.J., "Application of Fredholm theory to elastic scattering," J.Phys.B: Atom.Molec. Phys., 3, 1970, pp. 191-197.

MOTT, N.F., and MASSEY, H.S.W., The Theory of Atomic Collisions, Oxford, 1965.

N.A.G. LIBRARY — D02AHF (Mark 5), 1974.

NASH, J.C. — "A one-sided transformation method for the singular value decomposition and algebraic eigenproblem", Comput. J., 18, 1973, pp. 74-76.

NORCROSS, D.W., — "Asymptotic solution of coupled equations for electron scattering", Computer Phys. Commun., 1, 1969, pp. 88-96.

O'SHEA, B.B. — "Algorithms for the solution of systems of coupled second-order ordinary differential equations", Ph.D. thesis, University of London, 1971.

O'SHEA, B.B. — "Algorithms for the solution of systems of coupled second-order ordinary differential equations", J.Comput. Appd. Math., 4, 1978 pp. 11-17.

RAPTIS, A., and ALLISON, A.C., "Exponential-fitting methods for the numerical solution of the Schrödinger equation", preprint, 1977; subsequently published in Computer Phys. Commun., 14, 1978, pp. 1-5.

RIEHL, J.P., DIESTLER, D.J., and WAGNER, A.F., "Comparison of perturbation and direct numerical integration techniques for the calculation of phase shifts for elastic scattering", J.Comput.Phys., 15, 1974, pp. 212-225.

ROBERTSON, H.H., — "Phase calculations for nuclear scattering on the Pilot Ace", Proc. Camb.Phil.Soc., 52, 1956, pp. 538-545.

SCOTT, M.R., and WATTS, H.A., "SUPORT - A computer code for two-point boundary-value problems via orthonormalization", Rept. SAND-75-0198, Sandia Laboratories, Albuquerque, New Mexico, 1955.

SCOTT, M.R., and WATTS, H.A., "Computational solution of linear two-point boundary value problems via orthonormalization", SIAM J.Numer. Anal., 14, 1977, pp. 40 - 70.

SCRATON, R.E., ~~"Estimation of the truncation error in Runge-Kutta and allied processes", Comput. J., 7, 1964, pp. 246-248.~~ "The numerical solution of second-order differential equations not containing the first derivative explicitly" Comput. J., 6, 1964, pp. 368-370.

SHAMPINE, L.F./GORDON, M.K., Computer Solution of Ordinary Differential Equations, W.H.Freeman and Co., San Francisco, 1975.

SHAMPINE, L.F., and WATTS, H.A., "Comparing error estimators for Runge-Kutta methods", Math.Comp., 25, 1971, pp. 445-455.

SHAMPINE, L.F., and WATTS, H.A., "Global error estimation for ordinary differential equations", ACM Trans. Math. Software, 2, 1976, pp. 172-186.

SHAMPINE, L.F., WATTS, H.A., and DAVENPORT, S.M., "Solving non-stiff ordinary differential equations - the state of the art", Rept. SAND-75-0182, Sandia Laboratories, Albuquerque, New Mexico, 1976.

SIEMIENIUCH, J.L., "A study of a method of Fred T.Krogh for the numerical solution of ordinary differential equations", M.Sc. thesis, University of Manchester, 1972.

SLOAN, I.H., "Errors in the Numerov and Runge-Kutta methods", J Comput. Phys., 2, 1968, pp. 414-416.

SMITH, K., HENRY, R.J.W., and BURKE, P.G., "Scattering of electrons by atomic systems with configurations $2p^q$ and $3p^q$", Phys.Rev., 147, 1966, pp. 21-28.

STERN, M.S., "Comparison of numerical solutions of the partial wave Schrödinger differential and integral equations", J.Comput. Phys. 25, 1977, pp. 56-70.

VERLET, L., "Computer 'experiments' on classical fluids. I.Thermodynamical properties of Lennard-Jones molecules", Phys. Rev., 159, 1967, pp. 98-103.

WALSH, J., "Initial and boundary value routines for ordinary differential equations" in Software for Numerical Mathematics, edited by D.J.Evans, Academic Press, London and New York, 1974, pp. 177-192.

```
      C   PROGRAM RADISH                                                    0001
```

LINE NUMBER — TEXT — PAGE 1

```
 1.000  C   PROGRAM RADISH                                                  0001
 2.000  C   BY J.P.COLEMAN AND J.MOHAMED,                                   0002
 3.000  C   DEPARTMENT OF MATHEMATICS, UNIVERSITY OF DURHAM, SOUTH ROAD, DURHAM.  0003
 4.000  C                                                                   0004
 5.000  C   THIS PROGRAM TESTS THE SUBROUTINE DEVOG BY SOLVING THE RADIAL    0005
 6.000  C   SCHRODINGER EQUATION  Y"+(E-V(X)-L(L+1)/X**2)Y=0 FOR SPECIFIED   0006
 7.000  C   PROJECTILE ENERGY E, ANGULAR MOMENTUM L, AND POTENTIAL V(X).     0007
 8.000  C   DEVOG SOLVES THE EQUATION Y"=F(X,Y) WHERE F(X,Y) IS SUPPLIED AS A  0008
 9.000  C   FUNCTION SUBPROGRAM.                                            0009
10.000  C   ALL REAL VARIABLES ARE IN DOUBLE PRECISION FORM.                0010
11.000  C                                                                   0011
12.000  C                                                                   0012
13.000        IMPLICIT REAL*8(A-H,K,O-Z)                                    0013
14.000        COMMON/EKLL1/E,K,PSIG,EPS,L,L1                                0014
15.000        DIMENSION VCOEFF(4),A(5)                                      0015
16.000  C                                                                   0016
17.000  C   DATA CARDS MUST BE AS FOLLOWS:                                  0017
18.000  C   THE FIRST CARD CONTAINS 4 REAL NUMBERS, COEFFICIENTS IN THE     0018
19.000  C   EXPANSION OF V(X), IN FIELDS OF 16 COLUMNS STARTING AT COLUMN 1.  0019
20.000  C   THE SECOND CARD CONTAINS 2 REAL NUMBERS (EPS,C) IN COLUMNS 1 TO 12  0020
21.000  C   AND 13 TO 24.                                                   0021
22.000  C   EACH SUBSEQUENT CARD CONTAINS AN INTEGER (L) IN COLUMNS 1 TO 3,  0022
23.000  C   AND 3 REAL NUMBERS (E,XF,PSIG) IN COLUMNS 4 TO 15, 16 TO 27,    0023
24.000  C   AND 28 TO 39.                                                   0024
25.000  C   IN THE LAST CARD L IS NEGATIVE.                                 0025
26.000  C   CHANNEL 1 IS THE INPUT DEVICE AND CHANNEL 2 THE OUTPUT DEVICE.  0026
27.000  C                                                                   0027
28.000        READ(1,1) (VCOEFF(I),I=1,4)                                   0028
29.000  C   VCOEFF(I) IS THE COEFFICIENT OF X**(I-2) IN THE EXPANSION OF V(X)  0029
30.000  C   ABOUT THE ORIGIN.                                               0030
31.000  C                                                                   0031
32.000        WRITE(2,5)                                                    0032
33.000  C   A CAPTION IS PRINTED                                            0033
34.000        READ(1,2) EPS,C                                               0034
35.000  C   EPS AND C ARE REQUIRED BY THE SUBROUTINE DEVOG.  EPS IS A TOLERANCE  0035
36.000  C   PARAMETER, AND C IS THE FACTOR BY WHICH THE STEPLENGTH IS TO BE  0036
37.000  C   INCREASED IF THE LOCAL TRUNCATION ERROR IS SUFFICIENTLY SMALL.   0037
38.000  C                                                                   0038
39.000        4 READ(1,3) L,E,XF,PSIG                                       0039
40.000        IF(L.LT.0) STOP                                               0040
```

APPENDIX 1

```
41.000 C   A NEGATIVE VALUE OF L INDICATES THE END OF THE DATA.                  0041
42.000       WRITE(2,5) E,L                                                     0042
43.000 C   PRINT THE VALUES OF THE PHYSICAL PARAMETERS E,L                      0043
44.000       WRITE(2,7) EPS,C                                                   0044
45.000 C   PRINT THE VALUES OF THE STEP-CONTROL PARAMETERS EPS,C                0045
46.000       WRITE(2,8) PSIG,XF                                                 0046
47.000 C   PRINT THE VALUES OF THE TERMINATION CONDITIONS PSIG,XF;              0047
48.000 C   THE CALCULATION IS TERMINATED WHEN THE RELATIVE DIFFERENCE BETWEEN   0048
49.000 C   TWO SUCCESSIVE ESTIMATES OF THE PHASE SHIFT IS LESS THAN OR EQUAL TO 0049
50.000 C   PSIG, OR WHEN THE POINT X=XF IS REACHED, WHICHEVER COMES FIRST.      0050
51.000 C                                                                        0051
52.000 C***STARTING SERIES                                                      0052
53.000       A(1)=1.0D0                                                         0053
54.000       A(2)=VCOEFF(1)/(2.0D0*(L+1))                                       0054
55.000       W=VCOEFF(2)-E                                                      0055
56.000       A(3)=(VCOEFF(1)*A(2)+W)/(4*L+6)                                    0056
57.000       A(4)=(VCOEFF(1)*A(3)+W*A(2)+VCOEFF(3))/(6*L+12)                    0057
58.000       A(5)=(VCOEFF(1)*A(4)+W*A(3)+VCOEFF(3)*A(2)+VCOEFF(4))/(8*L+20)     0058
59.000       H=(3.75D-1*EPS/DABS(A(5-L)))**0.25D0                               0059
60.000 C   H IS THE FIRST STEPLENGTH TRIED (SEE DEVOG).                         0060
61.000 C***WARNING: THIS CHOICE OF H MAKES SENSE ONLY IF L DOES NOT EXCEED 4.   0061
62.000       AL4=1.0D0/(L+4)                                                    0062
63.000       AL5=1.0D0/(L+5)                                                    0063
64.000       X0=(0.1D0*EPS/DABS(A(5)))**AL5                                     0064
65.000       X1=(0.1D0*EPS/(H*(L+5)*DABS(A(5))))**AL4                           0065
66.000       IF(X1.LT.X0) X0=X1                                                 0066
67.000 C   THE CHOSEN VALUE OF X0 ENSURES THAT THE FIRST NEGLECTED TERMS       0067
68.000 C   IN THE TAYLOR EXPANSIONS FOR Y0 AND HZ0 DO NOT EXCEED 0.1*EPS.      0068
69.000       XL=X0**L                                                           0069
70.000       IF(L.EQ.0) XL=1.0D0                                                007C
71.000       Y0=(X0*(X0*(X0*A(4)+A(3))+A(2))+A(1))*XL*X0                        0071
72.000       Z0=(X0*(X0*(X0*A(4)*(L+4)+A(3)*(L+3))+A(2)*(L+2))+(L+1)*A(1))*XL   0072
73.000       K=DSQRT(E)                                                         0073
74.000       L1=L*(L+1)                                                         0074
75.000       F0=F(X0,Y0,0)                                                      0075
76.000       WRITE(2,9) X0,Y0,Z0,F0                                             0076
77.000 C   PRINT THE STARTING VALUES X0,Y0,Z0,F0                                0077
78.000 C                                                                        0078
79.000       CALL DEVOG(H,X0,Y0,Z0,F0,EPS,C,XF)                                 0079
80.000       GO TO 4                                                            0080
```

```
81.000  C                                                                0081
82.000       1 FORMAT(4D16.8)                                            0082
83.000       2 FORMAT(2D12.3)                                            0083
84.000       3 FORMAT(I3,3D12.3)                                         0084
85.000       5 FORMAT(1115H1TEST RUN: SOLUTION OF THE RADIAL SCHRODINGER EQUATION    0085
86.000        1 Y''+(E-V(X)-L(L+1)/X**2)Y=0 WITH CALCULATION OF PHASE SHIFT)        0086
87.000       6 FORMAT(1H0,19HPHYSICAL PARAMETERS,7X,6HE    =,D13.6,1H,,2X,5HL    =    0087
88.000        1,I3)                                                      0088
89.000       7 FORMAT(1H ,23HSTEP-CONTROL PARAMETERS,3X,6HEPS  =,D10.3,3X,1H,,2X,    0089
90.000        15HC    =,D10.3)                                          0090
91.000       8 FORMAT(1H ,22HTERMINATION CONDITIONS,4X,6HPSIG =,D11.4,2X,1H,,2X,5    0091
92.000        1HXF  =,D11.4)                                            0092
93.000       9 FORMAT(1H ,15HSTARTING VALUES,11X,6HX0   =,D11.4,2X,1H,,2X,5HY0  =    0093
94.000        1,D11.4,2X,1H,,2X,4HZ0 =,D11.4,2X,1H,,2X,4HF0 =,D11.4)     0094
95.000          END                                                      0095
```

```
      SUBROUTINE DEVOG(H,X0,Y0,Z0,F0,EPS,C,XF)                          C096
C                                                                       0097
C                                                                       0098
C   THIS SUBROUTINE SOLVES THE DIFFERENTIAL EQUATION                    0099
C                     Y"=F(X,Y)                                         0100
C   BY DE VOGELAERE'S METHOD, GIVEN THE INITIAL VALUES Y0=Y(X0),        0101
C   Z0=Y'(X0), AND F0=F(X0,Y0).                                         0102
C                                                                       0103
C   EPS IS AN UPPER BOUND IMPOSED ON THE ESTIMATED TRUNCATION ERROR     0104
C       PER UNIT STEP.  THIS BOUND APPLIES TO THE ABSOLUTE OR RELATIVE  0105
C       ERROR ACCORDING AS THE ABSOLUTE VALUE OF THE SOLUTION AT THE    0106
C       POINT UNDER CONSIDERATION IS LESS OR GREATER THAN 1.            0107
C                                                                       0108
C   H .. IS AN INITIAL STEPLENGTH SUPPLIED BY THE USER.                 0109
C                                                                       0110
C   AT EACH STEP THE LOCAL TRUNCATION ERROR IS ESTIMATED.  IF THIS IS   0111
C   TOO LARGE THE CURRENT STEP IS DISCARDED AND A SMALLER STEPLENGTH IS  0112
C   CHOSEN.  WHEN THE ESTIMATED TRUNCATION ERROR IS SUFFICIENTLY SMALL  0113
C   THE STEPLENGTH IS INCREASED FOR THE NEXT STEP.                      0114
C                                                                       0115
C   C ..THE USER-SUPPLIED FACTOR (E.G. 2) BY WHICH THE STEPLENGTH IS    0116
C       INCREASED WHEN THIS CAN BE DONE WITHOUT EXCEEDING THE BOUND EPS.0117
C                                                                       0118
C   THE STATEMENTS BETWEEN LINES OF ASTERISKS REFER TO THE SCATTERING   0119
C   PROBLEM RATHER THAN TO THE INTEGRATION OF THE DIFFERENTIAL EQUATION. 0120
C                                                                       0121
C                                                                       0122
      IMPLICIT REAL*8(A-H,K,O-Z)                                        0123
C   NINC COUNTS THE NUMBER OF INCREASES IN STEPLENGTH REQUIRED          0124
C   DURING THE COURSE OF INTEGRATION.                                   0125
      NINC=0                                                            0126
C   NDEC COUNTS THE NUMBER OF DECREASES IN STEPLENGTH REQUIRED          0127
C   DURING THE COURSE OF INTEGRATION.                                   0128
      NDEC=0                                                            0129
C   THE INTEGER I COUNTS THE NUMBER OF STEPS CARRIED OUT                0130
      I=0                                                               0131
C                                                                       0132
C                                                                       0133
C   DESCRIPTION AND INITIALIZATION OF VARIABLES REQUIRED FOR            0134
C   BOOK-KEEPING AND ERROR CONTROL.                                     0135
```

```
136.000 C   J1 AND JS  REPRESENT RESPECTIVELY THE NUMBER OF SUCCESSFUL STEPS      0136
137.000 C              CARRIED OUT WITH THE CURRENT H AND THE PREVIOUS           0137
138.000 C              SUCCESSFUL H.                                             0138
139.000       5 J1=0                                                             0139
140.000         JS=0                                                             0140
141.000 C   J2: THE STEPLENGTH IS INCREASED FROM H TO C*H IF FOR 3 SUCCESSIVE   0141
142.000 C       STEPS THE PREDICTED TRUNCATION ERROR WITH A STEPLENGTH OF C*H IS  0142
143.000 C       SUFFICIENTLY SMALL.  J2 IS USED AS A COUNTER FOR THIS PURPOSE.   0143
144.000         J2=0                                                             0144
145.000 C   JD: WHEN A STEPLENGTH DECREASE IS NECESSARY A NEW H IS CHOSEN.      0145
146.000 C       CALCULATIONS MAY SHOW THAT THIS IS INADEQUATE, SO A FURTHER     0146
147.000 C       REDUCTION IS REQUIRED.  JD KEEPS TRACK OF THE STEPLENGTH        0147
148.000 C       DECREASES IN SUCH A CASE.  IT IS INCREASED BY 1 WITH EACH       0148
149.000 C       DECREASE AND IS SET TO ZERO WHEN AN ACCEPTABLE H HAS BEEN FOUND. 0149
150.000         JD=0                                                             0150
151.000 C   C1  IS THE RATIO OF THE CURRENT H TO THE PREVIOUS SUCCESSFUL VALUE.  0151
152.000         C1=1.0D0                                                         0152
153.000         C2=C1*C1                                                         0153
154.000 C   TR1: AT THE ITH STEP TR1 IS THE DIFFERENCE BETWEEN TWO ESTIMATES    0154
155.000 C        OF Y(2I-1).  INITIALLY IT IS GIVEN AN ARBITRARY LARGE VALUE.    0155
156.000         TR1=1.0D2                                                        0156
157.000 C                                                                        0157
158.000 C ***********************************************************************0158
159.000 C   IPS IS INITIALLY ZERO AND IS CHANGED TO 1 BY THE FUNCTION SUBPROGRAM  0159
160.000 C   F(X,Y,IPS) WHEN THE ASYMPTOTIC REGION IS REACHED.                     0160
161.000         IPS=0                                                            0161
162.000 C   THE PARAMETERS JP AND JCONV ARE EXPLAINED IN THE SUBROUTINE PS.      0162
163.000         JP=1                                                             0163
164.000         JCONV=0                                                          0164
165.000 C ***********************************************************************0165
166.000 C                                                                        0166
167.000         HMAX=(XF-X0)/5.0D0                                               0167
168.000 C   IF A STEPLENGTH GREATER THAN HMAX WERE USED THE TRUNCATION ERROR     0168
169.000 C   WOULD NOT BE ESTIMATED BEFORE XF IS REACHED.                         0169
170.000         IF(H.GT.HMAX) H=HMAX                                             0170
171.000 C                                                                        0171
172.000 C***INITIAL STEP                                                         0172
173.000 C   IN THIS SECTION Y1 IS CALCULATED, THE CONTRIBUTION FROM THE ERROR     0173
174.000 C   IN Y1 TO THAT IN Y2 IS ESTIMATED, AND THE STEPLENGTH H IS DECREASED,  0174
175.000 C   IF NECESSARY, UNTIL THIS CONTRIBUTION IS SUFFICIENTLY SMALL.         0175
```

```
176.000  C   A STEPLENGTH ACCEPTABLE FOR Y1 MAY BE FOUND INADEQUATE WHEN          0176
177.000  C   THE ERROR IN Y2 IS ESTIMATED; IN THAT CASE THIS SECTION WILL BE     0177
178.000  C   REPEATED. THE COUNTER I IS ZERO WHEN THIS SECTION IS FIRST USED     0178
179.000  C   AND I=1 AT ANY LATER ENTRY.                                         0179
180.000        Z=H*Z0                                                           0180
181.000        H2=H*H                                                           0181
182.000        FEVEN=H2*F0/3.0D0                                                0182
183.000        F2I=FEVEN                                                        0183
184.000        Y=Y0+Z+1.5D0*FEVEN                                               0184
185.000        X=X0+H                                                           0185
186.000        FODD=4.0D0*H2*F(X,Y,IPS)/3.0D0                                   0186
187.000        Y=Y+(FODD-4.0D0*FEVEN)/8.0D0                                     0187
188.000        FODD=4.0D0*H2*F(X,Y,IPS)/3.0D0                                   0188
189.000        X=X+H                                                            0189
190.000        Z1=Z+FEVEN                                                       0190
191.000        Y=Y0+2.0D0*Z1+FODD                                               0191
192.000        FEVEN=H2*F(X,Y,IPS)/3.0D0                                        0192
193.000        Z=Z1+FODD+FEVEN                                                  0193
194.000        IF(I.GT.0) GO TO 4                                               0194
195.000  C   THIS TEST AVOIDS UNNECESSARY COMPUTATION IF I=1 I.E. WHEN H IS     0195
196.000  C   LESS THAN A VALUE ALREADY FOUND ACCEPTABLE FOR Y1.                  0196
197.000        YODD1=Y-Z+(1.4D1*FEVEN+3.0D0*FODD-2.0D0*F2I)*6.25D-2             0197
198.000        X0H=X0+H                                                         0198
199.000        FODD1=4.0D0*H2*F(X0H,YODD1,IPS)/3.0D0                            0199
200.000        Y1ERR=0.75D0*(FODD1-FODD)/H                                      0200
201.000        DY1ERR=DABS(Y1ERR)                                               0201
202.000        DY=DABS(Y)                                                       0202
203.000        TOL=EPS*DMAX1(1.0D0,DY)                                          0203
204.000        IF(DY1ERR.LT.TOL) GO TO 4                                        0204
205.000        C1=(0.5D0*EPS/DY1ERR)**0.2D0                                     0205
206.000        H=C1*H                                                           0206
207.000        GO TO 5                                                          0207
208.000      4 J1=J1+1                                                          0208
209.000        F2I=FEVEN                                                        0209
210.000        ERRFAC=4.0D0/(4.5D1*H)                                           0210
211.000        X=X+H                                                            0211
212.000  C   ERRFAC IS A FACTOR WHICH APPEARS IN THE TRUNCATION-ERROR ESTIMATE. 0212
213.000  C                                                                      0213
214.000        I=1                                                              0214
215.000  C                                                                      0215
```

```
216.000 C                                                              0216
217.000 C***GENERAL DE VOGELAERE LOOP                                  0217
218.000       3     Z=Z+FEVEN                                          0218
219.000             Y=Y+Z                                              0219
220.000             YODD=Y+FEVEN-0.125D0*FODD                          0220
221.000             FODD=4.0D0*H2*F(X,YODD,IPS)/3.0D0                  0221
222.000             X=X+H                                              0222
223.000             Z=Z+FODD                                           0223
224.000             Y=Y+Z                                              0224
225.000             FEVEN=H2*F(X,Y,IPS)/3.0D0                          0225
226.000             Z=Z+FEVEN                                          0226
227.000 C    THIS COMPLETES THE EVALUATION OF Y(2I+1) AND Y(2I+2)       0227
228.000 C                                                              0228
229.000 C                                                              0229
230.000 C    TRUNCATION ERROR ESTIMATE.                                0230
231.000 C    FIRST A MORE ACCURATE ESTIMATE IS OBTAINED FOR Y(2I+1).  TR2 IS THE  0231
232.000 C    DIFFERENCE BETWEEN THIS AND THE EARLIER ESTIMATE OF Y(2I+1).  TR1 IS  0232
233.000 C    THE DIFFERENCE BETWEEN THE TWO ESTIMATES OF Y(2I-1) FROM THE  0233
234.000 C    PREVIOUS STEP.                                            0234
235.000             YODD1=Y-Z+(1.4D1*FEVEN+3.0D0*FODD-2.0D0*F2I)*6.25D-2  0235
236.000             TR2=YODD1-YODD                                     0236
237.000 C    TR2 HAS TO BE MODIFIED ON THE 2ND STEP WITH A NEW H, AND ON THE 1ST  0237
238.000 C    STEP WITH A NEW H IF ONLY ONE STEP WAS DONE WITH THE PREVIOUS H  0238
239.000             TR21=TR2                                           0239
240.000             IF(I.LE.2) GO TO 32                                0240
241.000             IF(J1.EQ.1.OR.(J1.EQ.0.AND.JS.EQ.1)) GO TO 30      0241
242.000                GO TO 32                                        0242
243.000 C           THEN MODIFY TR2                                    0243
244.000          30    TR2=BETA*TR21                                   0244
245.000                IF(J1.EQ.0) TR2=BETA1*TR21                      0245
246.000             CONTINUE                                           0246
247.000 C    TRERR IS THE ESTIMATED TRUNCATION ERROR PER UNIT STEP     0247
248.000          32    TRERR=(TR2-TR1)*ERRFAC                          0248
249.000             DTRERR=DABS(TRERR)                                 0249
250.000 C    IF TRUNCATION ERROR IS TOO LARGE GO TO 14.  THIS DOES NOT APPLY TO  0250
251.000 C    THE FIRST STEP SINCE TWO STEPS ARE REQUIRED FOR AN ERROR ESTIMATE.  0251
252.000 C    THE PARAMETER TOL PROVIDES AN ABSOLUTE OR RELATIVE ERROR CRITERION  0252
253.000 C    ACCORDING AS THE ABSOLUTE VALUE OF Y IS LESS OR GREATER THAN ONE.  0253
254.000             DY=DABS(Y)                                         0254
255.000             TOL=EPS*DMAX1(1.0D0,DY)                            0255
```

```
256.000            IF(DTRERR.GT.TOL.AND.I.GT.1) GO TO 14                    0256
257.000   C                                                                 0257
258.000   C                                                                 0258
259.000   C   CURRENT STEP ACCEPTED.  DATA STORED FROM PREVIOUS STEP UPDATED. 0259
260.000            JD=0                                                     0260
261.000            J1=J1+1                                                  0261
262.000            TR1=TR21                                                 0262
263.000            F21=FEVEN                                                0263
264.000            Z1=Z                                                     0264
265.000            FEVEN1=FEVEN                                             0265
266.000            FODD1=FODD                                               0266
267.000            Y1=Y                                                     0267
268.000            H1=H                                                     0268
269.000   C   UPDATE ERRFAC IF NECESSARY                                    0269
270.000            IF(J1.GT.2) GO TO 31                                     0270
271.000            ERRFAC=4.0D0/(4.5D1*H)                                   0271
272.000            IF(J1.EQ.1) ERRFAC=0.8D0*(C2/H)/(3.0D0*C2+5.0D0*C1+1.0D0) 0272
273.000      31    I=I+1                                                    0273
274.000   C                                                                 0274
275.000   C ***************************************************************** 0275
276.000   C   THE PHASE SHIFT IS CALCULATED IF IPS=1, WHICH MEANS THAT X IS  0276
277.000   C   SUFFICIENTLY LARGE                                            0277
278.000         IF (IPS.EQ.1) CALL PS(X,Y,JP,JCONV)                        0278
279.000   C   THE CALCULATION IS TERMINATED IF THE PHASE SHIFT HAS CONVERGED 0279
280.000         IF (JCONV.EQ.1) GO TO 1                                    0280
281.000   C ***************************************************************** 0281
282.000   C                                                                 0282
283.000   C                                                                 0283
284.000   C   PREDICT TRUNCATION ERROR FOR THE NEXT STEP WITH A STEPLENGTH OF C*H. 0284
285.000   C   IF THIS IS SUFFICIENTLY SMALL GO TO 13 TO INCREASE THE STEPLENGTH. 0285
286.000      10    DTRERR=DTRERR*C**4                                       0286
287.000            IF(DTRERR.LT.(0.5D0*TOL)) GO TO 13                       0287
288.000            J2=0                                                     0288
289.000   C                                                                 0289
290.000   C   STEPLENGTH UNCHANGED FOR NEXT STEP. UPDATE X AND RETURN TO BEGINNING 0290
291.000   C   OF LOOP BUT EXIT IF X EXCEEDS XF.                             0291
292.000      23    X=X+H                                                    0292
293.000            IF(X.GT.XF) GO TO 2                                      0293
294.000            GO TO 3                                                  0294
295.000   C***END OF DE VOGELAERE LOOP                                      0295
```

```
296.000  C                                                                    0296
297.000  C                                                                    0297
298.000  C   PRINT THE FINAL VALUES OF I,NINC,NDEC AND RETURN TO CALLING PROGRAM.  0298
299.000       1    WRITE(2,6)I,NINC,NDEC                                     0299
300.000            RETURN                                                    0300
301.000       2    WRITE(2,12)I,NINC,NDEC                                    0301
302.000            RETURN                                                    0302
303.000  C                                                                    0303
304.000  C                                                                    0304
305.000  C   THE CODING RELEVANT TO THE CHANGE OF STEPLENGTH IS CONTAINED     0305
306.000  C   BETWEEN LINES OF DASHES.                                         0306
307.000  C                                                                    0307
308.000  C ---------------------------------------------------------------    0308
309.000  C                                                                    0309
310.000  C                                                                    0310
311.000  C***STEPLENGTH DECREASE                                              0311
312.000  C   IF THE LAST STEP WAS UNSUCCESSFUL JUMP OVER THE UPDATING SECTION  0312
313.000     14 JD=JD+1                                                       0313
314.000        IF(J1.EQ.0) GO TO 20                                          0314
315.000        JS=J1                                                         0315
316.000        J1=0                                                          0316
317.000        TR3=TR1                                                       0317
318.000        F211=F21                                                      0318
319.000        C11=C1                                                        0319
320.000        C21=C2                                                        0320
321.000     20 C1=(0.5D0*TOL/DTRERR)**0.25D0                                 0321
322.000  C   SINCE THE LAST STEP IS REJECTED X IS REDUCED BY 2H BEFORE       0322
323.000  C   H IS GIVEN ITS NEW VALUE                                        0323
324.000        X=X-2.0D0*H                                                   0324
325.000        H=C1*H                                                        0325
326.000        NDEC=NDEC+1                                                   0326
327.000  C   IF I=2 ENTRY TO THIS SECTION MEANS THAT H WAS TOO LARGE IN THE LAST  0327
328.000  C   TWO STEPS SO IT IS NECESSARY TO RESTART FROM XO WITH THE NEW H   0328
329.000        IF(I.EQ.2) GO TO 5                                            0329
330.000        Z=Z1                                                          0330
331.000        F21=F211                                                      0331
332.000        FEVEN=FEVEN1                                                  0332
333.000        FODD=FODD1                                                    0333
334.000        Y=Y1                                                          0334
335.000  C   C1 NOW BECOMES THE RATIO OF THE NEW H TO THE LAST SUCCESSFUL H.  0335
```

```
336.000         C1=H/H1                                                  0336
337.000         C2=C1*C1                                                 0337
338.000   C    NOW ENTER RESTART SECTION                                  0338
339.000         GO TO 16                                                 0339
340.000   C                                                              0340
341.000   C                                                              0341
342.000   C***STEPLENGTH INCREASE                                        0342
343.000   C    NO ACTION TAKEN UNLESS J2=3.                              0343
344.000      13 J2=J2+1                                                  0344
345.000         IF(J2.LT.3) GO TO 23                                     0345
346.000         J2=0                                                     0346
347.000         JS=J1                                                    0347
348.000         J1=0                                                     0348
349.000         TR3=TR21                                                 0349
350.000         F2I1=F2I                                                 0350
351.000         C11=C1                                                   0351
352.000         C21=C2                                                   0352
353.000         NINC=NINC+1                                              0353
354.000         C1=C                                                     0354
355.000         C2=C1*C1                                                 0355
356.000         H=C1*H                                                   0356
357.000   C                                                              0357
358.000   C                                                              0358
359.000   C***RESTART SECTION. THIS PROVIDES THE DATA NECESSARY FOR REENTRY TO   0359
360.000   C    THE DEVOGELAERE LOOP WITH INCREASED OR DECREASED H.       0360
361.000      16 H2=H*H                                                   0361
362.000         Z=C1*Z                                                   0362
363.000         F2I=C2*F21                                               0363
364.000         FEVEN=C2*FEVEN                                           0364
365.000         FODD=C2*FODD                                             0365
366.000         FODD=4.0D0*(1.0D0-C1)*FEVEN+C1*FODD                      0366
367.000         ALPHA=(2.0D0+C1)*C1*C2/3.0D0                             0367
368.000         TR1=ALPHA*TR3                                            0368
369.000   C    SOME DIFFERENCES OCCUR IF THE PREVIOUS STEPLENGTH WAS USED ONLY ONCE   0369
370.000         IF(JS.EQ.1) GO TO 21                                     0370
371.000            GO TO 22                                              0371
372.000   C     THEN CALCULATE APPROPRIATE ERROR TERM WHEN JS=1           0372
373.000      21    ERRFAC=4.8D0*C2*C21/((C21*(-C2+7.0D0*C1+1.2D1)+2.0D0*C11*(-C2+6   0373
374.000        1   .0D0*C1+1.0D1)+2.0D0*(C1+2.0D0))*H)                    0374
375.000            IF(JD.EQ.1) BETA1=BETA                                0375
```

818

```
376.000            GO TO 24                                                    0376
377.000    C     ELSE                                                         0377
378.000       22    ERRFAC=1.6D0*(C2/H)/(1.2D1+7.0D0*C1-C2)                   0378
379.000           CONTINUE                                                     0379
380.000       24  BETA=ALPHA/(C2*C2)                                          0380
381.000           X=X+H                                                        0381
382.000           IF(X.GT.XF) GO TO 2                                          0382
383.000           GO TO 3                                                      0383
384.000    C                                                                   0384
385.000    C ------------------------------------------------------------------0385
386.000    C                                                                   0386
387.000        6 FORMAT(1H0,46HPHASE SHIFT HAS CONVERGED TO REQUIRED ACCURACY/39HNU 0387
388.000         1MBER OF INTEGRATION STEPS CARRIED OUT,I6/75HNUMBER OF INCREASES IN 0388
389.000         2 STEPLENGTH REQUIRED DURING THE COURSE OF INTEGRATION,I3/75HNUMBER 0389
390.000         3 OF DECREASES IN STEPLENGTH REQUIRED DURING THE COURSE OF INTEGRAT 0390
391.000         4ION,I3,///)                                                    0391
392.000       12 FORMAT(1H0,52HTHE END OF THE RANGE OF INTEGRATION HAS BEEN REACHED 0392
393.000         1/39HNUMBER OF INTEGRATION STEPS CARRIED OUT,I6/75HNUMBER OF INCREA 0393
394.000         2SES IN STEPLENGTH REQUIRED DURING THE COURSE IF INTEGRATION,I3/75H 0394
395.000         3NUMBER OF DECREASES IN STEPLENGTH REQUIRED DURING THE COURSE OF IN 0395
396.000         4TEGRATION,I3,///)                                              0396
397.000          END                                                            0397
```

```
398.000       SUBROUTINE PS(X,Y,JP,JCONV)                                    0398
399.000  C                                                                   0399
400.000  C                                                                   0400
401.000  C    THIS SUBROUTINE CALCULATES THE PHASE SHIFT FROM THE SOLUTION OF 0401
402.000  C    THE SCHRODINGER EQUATION AT TWO POINTS XM AND X AT LEAST ONE    0402
403.000  C    ATOMIC UNIT APART.                                             0403
404.000  C    Y IS THE VALUE OF THE WAVE FUNCTION AT X                       0404
405.000  C    JP=1 ON FIRST ENTRY TO THIS SUBROUTINE AND =2 ON LATER ENTRIES  0405
406.000  C    JCONV IS SET EQUAL TO ZERO IN THE CALLING PROGRAM.  IT IS CHANGED 0406
407.000  C    TO ONE WHEN THE RELATIVE DIFFERENCE BETWEEN TWO SUCCESSIVE      0407
408.000  C    ESTIMATES OF THE PHASE SHIFT IS <=PSIG                          0408
409.000  C                                                                   0409
410.000  C    COMMON BLOCKS:                                                 0410
411.000  C      EKLL1 TRANSFERS ENERGY, WAVE NUMBER, ANGULAR MOMENTUM AND THE 0411
412.000  C            TOLERANCE PARAMETERS PSIG AND EPS FROM THE MAIN PROGRAM 0412
413.000  C      PHASE WHICH IS USED ONLY IN THIS SUBROUTINE STORES DATA FROM   0413
414.000  C            THE PRECEEDING ENTRY TO THE SUBROUTINE.                 0414
415.000  C                                                                   0415
416.000  C                                                                   0416
417.000       IMPLICIT REAL*8(A-H,K,O-Z)                                     0417
418.000       COMMON/EKLL1/E,K,PSIG,EPS,L,L1/PHASE/XM,YM,SM,CM,PSM           0418
419.000       IF (JP.EQ.1) GO TO 1                                           0419
420.000  C    THE FUNCTION OF THE SUBROUTINE ON THE FIRST ENTRY IS TO STORE DATA 0420
421.000  C    WHICH WILL BE USED LATER IN A PHASE-SHIFT CALCULATION.          0421
422.000       XINC=X-XM                                                      0422
423.000       IF(XINC.LT.1.0) RETURN                                         0423
424.000       KX=K*X                                                         0424
425.000       S=REG(KX,L)                                                    0425
426.000       C=AIREG(KX,L)                                                  0426
427.000       ANUM=Y*SM-YM*S                                                 0427
428.000       ADENOM=YM*C-Y*CM                                               0428
429.000       A=ANUM/ADENOM                                                  0429
430.000       PSHIFT=DATAN(A)                                                0430
431.000       WRITE(2,5) X,PSHIFT                                            0431
432.000       P=DABS((PSHIFT-PSM)/PSHIFT)                                    0432
433.000       IF (P.LT.PSIG) JCONV=1                                         0433
434.000  C                                                                   0434
435.000  C    STORAGE OF DATA FOR NEXT PHASE-SHIFT CALCULATION               0435
436.000     2 PSM=PSHIFT                                                     0436
437.000       XM=X                                                           0437
```

```
438.000          YM=Y                                                    0438
439.000          SM=S                                                    0439
440.000          CM=C                                                    0440
441.000    C                                                             0441
442.000        4 FORMAT(1HO,5X,1HX,11X,11HPHASE SHIFT)                   0442
443.000        5 FORMAT(1H ,D11.4,5X,D13.6)                              0443
444.000          RETURN                                                  0444
445.000    C                                                             0445
446.000    C     THIS SECTION IS USED ONLY ON THE FIRST ENTRY             0446
447.000        1 KX=K*X                                                  0447
448.000          S=REG(KX,L)                                             0448
449.000          C=AIREG(KX,L)                                           0449
450.000          PSHIFT=0.0D0                                            045C
451.000          JP=2                                                    0451
452.000          WRITE(2,4)                                              0452
453.000          GO TO 2                                                 0453
454.000          END                                                     0454
```

```
      REAL FUNCTION F*8(X,Y,IPS)                                        0455
```

LINE NUMBER TEXT PAGE 1

```
455.000       REAL FUNCTION F*8(X,Y,IPS)                                0455
456.000 C                                                               0456
457.000 C     THIS SUBPROGRAM RETURNS THE VALUE OF Y"(X).               0457
458.000 C     IPS IS SET EQUAL TO ZERO IN THE CALLING PROGRAM, AND IS CHANGED TO  0458
459.000 C         1 WHEN THE ASYMPTOTIC REGION HAS BEEN REACHED I.E. WHEN V(X)  0459
460.000 C         CAN BE NEGLECTED IN COMPARISON WITH L(L+1)/X**2-E.     0460
461.000 C                                                               0461
462.000       IMPLICIT REAL*8(A-H,K,O-Z)                                0462
463.000       COMMON/EKLL1/E,K,PSIG,EPS,L,L1                            0463
464.000 C     EPS IS THE TOLERANCE PARAMETER USED IN DEVOG               0464
465.000 C     PSIG IS NOT USED IN THIS SUBPROGRAM                       0465
466.000       V=-2.0D0*(1.0D0+1.0D0/X)*DEXP(-2.0D0*X)                   0466
467.000 C     HERE V IS THE POTENTIAL FOR AN ELECTRON IN THE STATIC FIELD  0467
468.000 C     OF THE HYDROGEN ATOM                                      0468
469.000       A=L1/(X*X)-E                                              0469
470.000       IF(A.LT.0.0D0.AND.DABS(V).LT.EPS*(-A)) IPS=1              047C
471.000       F=(A+V)*Y                                                 0471
472.000       RETURN                                                    0472
473.000       END                                                       0473
```

```
474.000          REAL FUNCTION REG*8(X,L)                                        0474
475.000    C     SPHERICAL BESSEL FUNCTION OF THE FIRST KIND TIMES X.            0475
476.000    C     REG(X,L)=X*JL(X) IN THE NOTATION OF ABRAMOWITZ  AND             0476
477.000    C     STEGUN HANDBOOK OF MATHEMATICAL FUNCTIONS P437.                 0477
478.000    C     IF L DOES NOT EXCEED X FORWARD RECURRENCE IS USED, OTHERWISE    0478
479.000    C     BACKWARD RECURRENCE IS USED WITH STARTING CONDITIONS FOUND AS   0479
480.000    C     SUGGESTED BY CORBATO AND URETSKY J. ASSOC. COMP. MACH. VOL 6,   0480
481.000    C                                          PP 366-375 (1959).       0481
482.000          IMPLICIT REAL*8(A-H,O-Z)                                        0482
483.000          DIMENSION PF(40)                                                0483
484.000          IF (L.GT.X) GO TO 3                                             0484
485.000    C                                                                     0485
486.000    C     FORWARD RECURRENCE                                              0486
487.000          A=DSIN(X)                                                       0487
488.000          IF (L.EQ.0) GO TO 2                                             0488
489.000          B=DCOS(X)                                                       0489
490.000          FAC=1.0D0/X                                                     0490
491.000          X2=FAC+FAC                                                      0491
492.000          DO 1 J=1,L                                                      0492
493.000             A1=A                                                         0493
494.000             A=FAC*A-B                                                    0494
495.000             B=A1                                                         0495
496.000             FAC=FAC+X2                                                   0496
497.000        1 CONTINUE                                                        0497
498.000        2 REG=A                                                           0498
499.000          RETURN                                                          0499
500.000    C                                                                     0500
501.000    C  ************************************                               0501
502.000    C                                                                     0502
503.000    C     A RELATIVE ERROR LESS THAN 1.0D-7 IS DEMANDED WHEN BACKWARD     0503
504.000    C     RECURRENCE IS USED.  IF THE DIMENSION OF THE ARRAY PF IS        0504
505.000    C     INSUFFICIENT TO ENSURE THIS ACCURACY A WARNING IS PRINTED.      0505
506.000        3 IF (L.GE.40) GO TO 6                                            0506
507.000          ERLG=23.25D0                                                    0507
508.000    C     ERLG IS THE ABSOLUTE VALUE OF THE LOG TO THE BASE 2 OF DELTA    0508
509.000    C     WHERE DELTA IS THE MAXIMUM PERMITTED RELATIVE ERROR IN REG.     0509
510.000    C     HERE DELTA=1.0D-7.                                              0510
511.000    C     NU1 IS DEFINED BY EQN(31) OF CORBATO AND URETSKY.               0511
512.000          U=2.0D0*X/DFLOAT(2*L+1)                                         0512
513.000          NU1=L+IDINT(ERLG*(0.1D0+0.175D0*U*(2.0D0-U*U)/(1.0D0-U*U)))     0513
```

```
                REAL FUNCTION REG*8(X,L)                                        0474
514.000         NP=IDINT(X-0.5D0+DSQRT(ERLG*0.35D0*X))                          0514
515.000         IF (NP.LT.L) GO TO 4                                            0515
516.000         U=2.0D0*X/DFLOAT(2*NP+1)                                        0516
517.000         NUP=NP+IDINT(ERLG*(0.1D0+0.175D0*U*(2.0D0-U*U)/(1.0D0-U*U)))    0517
518.000 C       BACKWARD RECURRENCE BEGINS WITH THE SMALLER OF NU1 AND NUP.      0518
519.000         IF (NUP.LT.NU1) NU1=NUP                                         0519
520.000 C       REDUCED ACCURACY WHEN NU1 TOO LARGE FOR DECLARED ARRAY          0520
521.000         IF (NU1.GE.40) GO TO 7                                          0521
522.000       4 PF(NU1+1)=1.0D-10                                               0522
523.000         FAC=DFLOAT(NU1+NU1+1)/X                                         0523
524.000         PF(NU1)=FAC*PF(NU1+1)                                           0524
525.000         X2=2.0D0/X                                                      0525
526.000         J1=NU1-1                                                        0526
527.000         DO 5 J=1,J1                                                     0527
528.000            FAC=FAC-X2                                                   0528
529.000            PF(NU1-J)=FAC*PF(NU1+1-J)-PF(NU1+2-J)                        0529
530.000       5 CONTINUE                                                        0530
531.000         CRN=(PF(1)/X-PF(2))*DCOS(X)+DSIN(X)*PF(1)                       0531
532.000         REG=PF(L+1)/CRN                                                 0532
533.000         RETURN                                                          0533
534.000 C                                                                       0534
535.000       6 WRITE(2,100) L                                                  0535
536.000     100 FORMAT(1H ,3HL= ,I3,52H TOO LARGE FOR ARRAY DECLARED.  REG REPLACE  0536
537.000        1D BY ZERO)                                                      0537
538.000         REG=0.0D0                                                       0538
539.000         RETURN                                                          0539
540.000 C                                                                       0540
541.000       7 WRITE(2,101) L,NU1                                              0541
542.000     101 FORMAT(1H ,31HFOR BACKWARD RECURRENCE WITH L=,I3,32H THE CALCULATE  0542
543.000        1D VALUE OF NU1 IS ,I3,24H SINCE THIS IS TOO LARGE/45H FOR THE DECL  0543
544.000        2ARED ARRAY NU1 IS REPLACED BY 39)                               0544
545.000         NU1=39                                                          0545
546.000         GO TO 4                                                         0546
547.000         END                                                             0547
```

```
548.000          REAL FUNCTION AIREG*8(X,L)                                    0548
549.000    C     SPHERICAL BESSEL FUNCTION OF THE SECOND KIND TIMES -X.        0549
550.000    C     AIREG(X,L)=-X*YL(X) IN THE NOTATION OF ABRAMOWITZ AND         0550
551.000    C     STEGUN HANDBOOK OF MATHEMATICAL FUNCTIONS P437.               0551
552.000    C     CALCULATION IS BY FORWARD RECURRENCE.                         0552
553.000    C     NO OVERFLOW PROTECTION IS INCLUDED SINCE THIS ROUTINE         0553
554.000    C     WILL NOT BE CALLED FOR VERY SMALL VALUES OF X                 0554
555.000    C                                                                   0555
556.000          IMPLICIT REAL*8(A-H,O-Z)                                      0556
557.000          A=DCOS(X)                                                     0557
558.000          IF (L.EQ.0) GO TO 2                                           0558
559.000          FAC=1.0D0/X                                                   0559
560.000          B=-DSIN(X)                                                    056C
561.000          X2=FAC+FAC                                                    0561
562.000          DO 1 J=1,L                                                    0562
563.000             A1=A                                                       0563
564.000             A=FAC*A-B                                                  0564
565.000             FAC=FAC+X2                                                 0565
566.000             B=A1                                                       0566
567.000        1 CONTINUE                                                      0567
568.000        2 AIREG=A                                                       056E
569.000          RETURN                                                        0569
570.000          END                                                           057C
```

TABLE 32

E \ L	0	1	2
0.16	1.0575 1.0575 1.0577 ------ 1.0575*	0.0146 0.0146 0.0145 ------ 0.0147*	0.0005 0.0005 * * * ------ 0.0005*
0.25	1.0447 1.0447 1.0445 ------ 1.0448*	0.0260 0.0260 0.0258 ------ 0.0260*	0.0014 0.0014 0.0012 ------ 0.0014*
0.5	0.9908 0.9908 0.9912 ------ 0.9909*	0.0584 0.0584 0.0583 ------ 0.0584*	0.0055 0.0056 0.0054 ------ 0.0056*
0.8	0.9356 0.9356 0.9351 ------ 0.9356*	0.0924 0.0924 0.0919 ------ 0.0924*	
1.0	0.90550 0.90552 0.90562 ------- 0.905677**	0.111468 0.111474 0.111103 -------- 0.111510**	
4.0	0.695 0.695 0.689 ----- 0.694+		
9.0	0.572 0.573 0.570 ----- 0.569+		
16.0	0.492 0.492 0.486 ----- 0.490+		
25.0	0.434 0.434 0.429 ----- 0.432+		

Corresponding to each value of E and L there are 4 tabulated values of the phase shift:

the first corresponds to using EPS = 10^{-6}, PSIG = 10^{-4}

the second corresponds to using EPS = 10^{-8}, PSIG = 10^{-5}

the third corresponds to using EPS = 10^{-4}, PSIG = 10^{-3}

the fourth corresponds to the published result.

* refers to the published result of Burke and Smith (1962).

** refers to the published result of Chandrasekhar and Breen (1946).

\+ refers to the published result of McDougall (1932).

*** phase shift failed to converge to required accuracy over specified range of integration.

```
      C   THIS PROGRAM TESTS THE SUBROUTINE RAPAL BY SOLVING THE RADIAL
```



```
  1.000 C   THIS PROGRAM TESTS THE SUBROUTINE RAPAL BY SOLVING THE RADIAL
  2.000 C   SCHRODINGER EQUATION Y''+(E-L(L+1)/X**2-V(X))Y=0 FOR SPECIFIED
  3.000 C   PROJECTILE ENERGY E, ANGULAR MOMENTUM L, AND POTENTIAL V(X).
  4.000 C   ALL REAL VARIABLES ARE IN DOUBLE PRECISION FORM.
  5.000 C
  6.000 C
  7.000       IMPLICIT REAL*8(A-H,O-Z)
  8.000       COMMON/EKLL1/E,AK,PSIG,EPS,C,L,L1
  9.000       COMMON VCOEFF(8),A(9),F(8000),XX(8000),D(6,7),JY
 10.000 C
 11.000 C   DATA CARDS MUST BE AS FOLLOWS:
 12.000 C   THE FIRST TWO CARDS CONTAIN 8 REAL NUMBERS, COEFFICIENTS IN THE
 13.000 C   EXPANSION OF V(X), IN FIELDS OF 16 COLUMNS STARTING AT COLUMN 1.
 14.000 C   THE THIRD CARD CONTAINS 1 REAL NUMBER (EPS) IN COLUMNS 1 TO 12.
 15.000 C   EACH SUBSEQUENT CARD CONTAINS AN INTEGER (L) IN COLUMNS 1 TO 3,
 16.000 C   AND 4 REAL NUMBERS (E,XF,PSIG,H) IN COLUMNS 4 TO 15, 16 TO 27,
 17.000 C   28 TO 39 AND 40 TO 51.
 18.000 C   IN THE LAST CARD L IS NEGATIVE.
 19.000 C   CHANNEL 1 IS THE INPUT DEVICE AND CHANNEL 2 THE OUTPUT DEVICE.
 20.000 C
 21.000       READ(1,1) (VCOEFF(I),I=1,8)
 22.000 C   VCOEFF(I) IS THE COEFFICIENT OF X**(I-2) IN THE EXPANSION OF V(X)
 23.000 C   ABOUT THE ORIGIN.
 24.000 C
 25.000       WRITE(2,5)
 26.000 C   A CAPTION IS PRINTED
 27.000       READ(1,2) EPS
 28.000       C=2.0D0
 29.000 C   EPS AND C ARE REQUIRED BY THE SUBROUTINE RAPAL. EPS IS A TOLERANCE
 30.000 C   PARAMETER, AND C IS THE FACTOR BY WHICH THE STEPLENGTH IS TO BE
 31.000 C   INCREASED IF THE LOCAL TRUNCATION ERROR IS SUFFICIENTLY SMALL.
 32.000 C
 33.000     4 READ(1,3) L,E,XF,PSIG,H
 34.000       IF(L.LT.0) STOP
 35.000 C   A NEGATIVE VALUE OF L INDICATES THE END OF THE DATA.
 36.000       WRITE(2,6) E,L
 37.000 C   PRINT THE VALUES OF THE PHYSICAL PARAMETERS E,L
 38.000       WRITE(2,7) EPS,C
 39.000 C   PRINT THE VALUES OF THE STEP-CONTROL PARAMETERS EPS,C
 40.000       WRITE(2,8) PSIG,XF
```

APPENDIX 2

```
C   THIS PROGRAM TESTS THE SUBROUTINE RAPAL BY SOLVING THE RADIAL
C   PRINT THE VALUES OF THE TERMINATION CONDITIONS PSIG,XF;
C   THE CALCULATION IS TERMINATED WHEN THE RELATIVE DIFFERENCE BETWEEN
C   TWO SUCCESSIVE ESTIMATES OF THE PHASE SHIFT IS LESS THAN OR EQUAL TO
C   PSIG, OR WHEN THE POINT X=XF IS REACHED, WHICHEVER COMES FIRST.
C
C***STARTING SERIES
      A(1)=1.0D0
      A(2)=VCOEFF(1)/(2.0D0*(L+1))
      W=VCOEFF(2)-E
      A(3)=(VCOEFF(1)*A(2)+W)/(4*L+6)
      A(4)=(VCOEFF(1)*A(3)+W*A(2)+VCOEFF(3))/(6*L+12)
      A(5)=(VCOEFF(1)*A(4)+W*A(3)+VCOEFF(3)*A(2)+VCOEFF(4))/(8*L+20)
      JH=0
      AL5=1.0D0/(L+5)
      X0=(0.1D0*EPS*EPS/DABS(A(5)))**AL5
C   THE CHOSEN VALUE OF X0 ENSURES THAT THE FIRST NEGLECTED TERM
C   IN THE TAYLOR EXPANSION FOR Y0 DOES NOT EXCEED 0.1*EPS*EPS.
      XL=X0**L
      IF(L.EQ.0) XL=1.0D0
      Y0=(X0*(X0*(X0*A(4)+A(3))+A(2))+A(1))*XL*X0
   10 X=X0+H
C   H IS THE FIRST STEPLENGTH TRIED.
      H12=H*H/1.2D1
      XL=X**L
      Y1=(X*(X*(X*(X*A(5)+A(4))+A(3))+A(2))+A(1))*XL*X
      X2=X+H
      L1=L*(L+1)
      VN=L1/(X*X)-E+POT(X,IPS)
      VD=L1/(X2*X2)-E+POT(X2,IPS)
      Y1N=2.0D0*(1.0D0+5.0D0*H12*VN)
      Y1D=1.0D0-H12*VD
      IF(JH.EQ.1) GO TO 14
C   IF JH=0 CALCULATE THE COEFFICIENTS A(6),---,A(9); WHEN JH=1
C   AVOID UNNECESSARY RE-COMPUTATION OF THESE COEFFICIENTS.
      DO 12 I=6,9
      IM1=I-1
      SUM=0.0D0
      DO 11 J=3,IM1
      SUM=SUM+VCOEFF(J)*A(I-J)
   11 CONTINUE
```

```
      A(I)=(VCOEFF(1)*A(I-1)+W*A(I-2)+SUM)/((I-1)*(2*L+I))
   12 CONTINUE
C    Y1 IS CALCULATED AND THE CONTRIBUTION FROM THE ERROR IN Y1 TO THAT
C    IN Y2 IS ESTIMATED, AND THE STEPLENGTH H IS DECREASED, IF NECESSARY,
C    UNTIL THIS CONTRIBUTION IS SUFFICIENTLY SMALL.
   14 DO 15 I=6,9
      TERM=A(I)*X**(L+I)
      TERM1=TERM*Y1N/Y1D
      DTERM1=DABS(TERM1)
      IF(DTERM1.LT.H*EPS) GO TO 17
      Y1=Y1+TERM
      IF(I.EQ.9) GO TO 16
   15 CONTINUE
   16 C1=(0.5D0*H*EPS/DTERM1)**0.2D0
      JH=1
      H=C1*H
      GO TO 10
   17 JY=I-1
      AK=DSQRT(E)
      WRITE(2,9) X0,Y0,Y1
C    PRINT THE STARTING VALUES X0,Y0,Y1
C
      CALL RAPAL(H,X0,X,Y0,Y1,XF)
      GO TO 4
C
    1 FORMAT(5D16.8)
    2 FORMAT(D12.3)
    3 FORMAT(I3,4D12.3)
    5 FORMAT(115H1TEST RUN: SOLUTION OF THE RADIAL SCHRODINGER EQUATION
     1 Y''+(E-L(L+1)/X**2-V(X))Y=0 WITH CALCULATION OF PHASE SHIFT/42H A
     2RISING FROM THE SCATTERING PROBLEM (E>0))
    6 FORMAT(1HO,19HPHYSICAL PARAMETERS,7X,6HE    =,D13.6,1H,,2X,5HL    =
     1,I3)
    7 FORMAT(1H ,23HSTEP-CONTROL PARAMETERS,3X,6HEPS  =,D10.3,3X,1H,,2X,
     15HC    =,D10.3)
    8 FORMAT(1H ,22HTERMINATION CONDITIONS,4X,6HPSIG =,D11.4,2X,1H,,2X,5
     1HXF  =,D11.4)
    9 FORMAT(1H ,15HSTARTING VALUES,11X,6HX0  =,D11.4,2X,1H,,2X,5HY0  =
     1,D11.4,2X,1H,,2X,4HY1 =,D11.4)
      END
```

```
121.000         SUBROUTINE RAPAL(H,XO,X,YO,Y1,XF)
122.000   C
123.000   C
124.000   C   THIS SUBROUTINE SOLVES THE DIFFERENTIAL EQUATION
125.000   C                        Y''=F(X,Y)
126.000   C   BY THE RAPTIS AND ALLISON METHOD WHICH EXPLOITS THE A PRIORI
127.000   C   KNOWLEDGE OF THE ASYMPTOTIC FORM OF THE SOLUTION OF THE SCHRODINGER
128.000   C   EQUATION.
129.000   C   THIS METHOD IS A VARIANT OF NUMEROV'S METHOD WHERE NOW THE
130.000   C   COEFFICIENTS OF THE FORMULA DEPEND ON THE LENGTH OF THE INTERVAL
131.000   C   OVER WHICH INTEGRATION IS PERFORMED.
132.000   C   H .. IS AN INITIAL STEPLENGTH SUPPLIED BY THE USER. AT EACH STEP
133.000   C        THE LOCAL TRUNCATION ERROR IS ESTIMATED. IF THIS IS TOO LARGE
134.000   C        THE CURRENT STEP IS DISCARDED AND A SMALLER STEPLENGTH IS
135.000   C        CHOSEN. WHEN THE ESTIMATED TRUNCATION ERROR IS SUFFICIENTLY
136.000   C        SMALL THE STEPLENGTH IS INCREASED FOR THE NEXT STEP. IF THE
137.000   C        FIRST ESTIMATE OF THE TRUNCATION ERROR EXCEEDS EPS THE PROGRAM
138.000   C        PREDICTS A SUITABLE STEPLENGTH AND STARTS AGAIN AT XO AT THE
139.000   C        BEGINNING OF THE RANGE OF INTEGRATION.
140.000   C   C .. THE STEPLENGTH IS DOUBLED WHEN THIS CAN BE DONE WITHOUT
141.000   C        EXCEEDING THE BOUND EPS.
142.000   C
143.000   C   THE STATEMENTS BETWEEN LINES OF ASTERISKS REFER TO THE SCATTERING
144.000   C   PROBLEM RATHER THAN TO THE INTEGRATION OF THE DIFFERENTIAL EQUATION.
145.000   C
146.000   C
147.000         IMPLICIT REAL*8(A-H,O-Z)
148.000         COMMON/EKLL1/E,AK,PSIG,EPS,C,L,L1
149.000         COMMON VCOEFF(8),A(9),F(8000),XX(8000),D(6,7),JY
150.000   C   NINC COUNTS THE NUMBER OF INCREASES IN STEPLENGTH REQUIRED
151.000   C   DURING THE COURSE OF INTEGRATION.
152.000         NINC=0
153.000   C   NDEC COUNTS THE NUMBER OF DECREASES IN STEPLENGTH REQUIRED
154.000   C   DURING THE COURSE OF INTEGRATION.
155.000         NDEC=0
156.000   C
157.000         1 GO TO 4
158.000   C   IF THE INITIAL VALUE OF H PROVES TO BE TOO LARGE Y1 MUST BE
159.000   C   RECALCULATED WITH THE NEW H.
160.000         2 X=XO+H
```

```
      XL=X**L
      Y1=(X*(X*(X*(X*A(5)+A(4))+A(3))+A(2))+A(1))*XL*X
      DO 3 J=0,JY
      TERM=A(I)*X**(L+1)
      Y1=Y1+TERM
    3 CONTINUE
      TERM=A(JY+1)*X**(L+JY+1)
      DTERM=DABS(TERM)
      IF(DTERM.GT.0.5D0*H*EPS) GO TO 22
      CONTINUE
C
C   DESCRIPTION AND INITIALIZATION OF VARIABLES REQUIRED FOR
C   BOOK-KEEPING AND ERROR CONTROL.
C   J IS SET TO 1 ON INITIAL ENTRY TO SUBROUTINE DIVDIF.
    4 J=0
C   J2: THE STEPLENGTH IS INCREASED FROM H TO C*H IF FOR 3 SUCCESSIVE
C       STEPS THE PREDICTED TRUNCATION ERROR WITH A STEPLENGTH OF C*H IS
C       SUFFICIENTLY SMALL. J2 IS USED AS A COUNTER FOR THIS PURPOSE.
      J2=0
C   JD: WHEN A STEPLENGTH DECREASE IS NECESSARY A NEW H IS CHOSEN.
C       CALCULATIONS MAY SHOW THAT THIS IS INADEQUATE, SO A FURTHER
C       REDUCTION IS REQUIRED. JD KEEPS TRACK OF THE STEPLENGTH
C       DECREASES IN SUCH A CASE. IT IS INCREASED BY 1 WITH EACH
C       DECREASE AND IS SET TO ZERO WHEN AN ACCEPTABLE H HAS BEEN FOUND.
      JD=0
C   TO DISTINGUISH BETWEEN CALLS MADE TO SUBROUTINE DIVDIF FOR THE
C   CALCULATION OF THE DIVIDED DIFFERENCES REQUIRED IN THE ESTIMATE
C   OF THE TRUNCATION ERROR PER UNIT STEP AND FOR EXTRACTING THE VALUES
C   NEEDED FOR THE INTERPOLATION PROCESS, DD IS SET TO ZERO IMMEDIATELY
C   AFTER THE VALUES FOR THE INTERPOLATION HAVE BEEN FOUND.
C   INITIALLY DD IS SET TO 100. (THIS IS ARBITRARY; TAKE DD TO BE ANY
C   NON-ZERO VALUE)
      DD=1.0D2
C   JB IS INITIALLY ZERO; JB IS SET TO 1 WHEN THE CLASSICAL REGION
C   IS REACHED AND AFTER THE CALCULATION OF THE NEW COEFFICIENT B01
C   WHICH REPLACES B0 IN THE NEXT STEP. THE RAPTIS AND ALLISON METHOD
C   IS USED IN THE NEXT STEP; IN THE UPDATING SECTION, BEFORE ENTERING
C   THIS STEP, JB IS SET TO 2 AND THIS INDICATES TO SUBROUTINE DIVDIF
C   THAT THE 4TH DIVIDED DIFFERENCE TABLE OF (E*Y+Y'') MUST BE SET UP.
C   THEREAFTER JB IS SET TO 3.
```

```
201.000           JB=0
202.000     C
203.000     C     ***************************************************************
204.000     C     IPS IS INITIALLY ZERO AND IS CHANGED TO 1 BY THE FUNCTION SUBPROGRAM
205.000     C     POT(X,IPS) WHEN THE CLASSICAL REGION IS REACHED.
206.000           IPS=0
207.000     C     THE PARAMETERS JP AND JCONV ARE EXPLAINED IN THE SUBROUTINE PS.
208.000           JP=1
209.000           JCONV=0
210.000     C     ***************************************************************
211.000     C
212.000     C     WE USE THE NUMEROV METHOD WITH CONSTANT COEFFICIENTS OUT TO THE
213.000     C     CLASSICAL TURNING POINT.
214.000     C
215.000     C***INITIAL STEP
216.000           B0=1.000/1.2D1
217.000           H2=H*H
218.000           F(1)=YO
219.000           V=L1/(XO*XO)-E+POT(XO,IPS)
220.000           YPREK=(1.0D0-B0*H2*V)*F(1)
221.000           F(2)=Y1
222.000           V=L1/(X*X)-E+POT(X,IPS)
223.000           H2VF=H2*V*F(2)
224.000           YK=(1.000-B0*H2*V)*F(2)
225.000           XX(1)=XO
226.000           XX(2)=X
227.000     C     F(K) IS THE SOLUTION AT XX(K)
228.000     C
229.000           K=2
230.000     C
231.000     C
232.000     C***KAPTIS AND ALLISON LOOP
233.000        15 Y=2.0D0*YK-YPREK+H2VF
234.000           X=X+H
235.000     C     EXIT FROM LOOP IF X EXCEEDS XF
236.000           IF(X.GT.XF) GO TO 2
237.000           V=L1/(X*X)-E+POT(X,IPS)
238.000           XX(K+1)=X
239.000           F(K+1)=Y/(1.0D0-B0*H2*V)
240.000           IF(IPS.NE.1.OR.JB.GT.0.OR.K+1.LT.7) GO TO 25
```

```
241.000  C
242.000  C   BEYOND THE CLASSICAL TURNING POINT WE LET THE COEFFICIENTS OF THE
243.000  C   NUMEROV METHOD VARY WITH THE INTERVAL H. (SEE RAPTIS AND ALLISON:
244.000  C   ' EXPONENTIAL-FITTING METHODS FOR THE NUMERICAL SOLUTION OF THE
245.000  C   SCHRODINGER EQUATION'.)
246.000        AKH=AK*H
247.000        AKH2=AKH/2.0D0
248.000        H2E=H2*E
249.000        B01=(H2E-4.0D0*DSIN(AKH2)*DSIN(AKH2))/(4.0D0*H2E*DSIN(AKH2)*DSIN(A
250.000       1KH2))
251.000        JB=1
252.000  C
253.000  C
254.000     25 CONTINUE
255.000  C   AN ESTIMATE OF THE TRUNCATION ERROR PER UNIT STEP IS CALCULATED
256.000  C   FOR K+1.GE.7, USING 6TH DIVIDED DIFFERENCES OF Y IN THE
257.000  C   NUMEROV METHOD, AND THEREAFTER USING 4TH DIVIDED DIFFERENCES
258.000  C   OF (E*Y+Y'') IN THE RAPTIS AND ALLISON METHOD.
259.000        IF(K+1.EQ.7) J=1
260.000        IF(K+1.GE.7) GO TO 9
261.000        GO TO 8
262.000     9  CALL DIVDIF(DD,K,J,JD,IPS,JB)
263.000        TRERR=-H**5*DD/2.4D2
264.000        DTRERR=DABS(TRERR)
265.000  C   IF TRUNCATION ERROR IS TOO LARGE GO TO 12.
266.000  C   THE PARAMETER TOL PROVIDES AN ABSOLUTE OR RELATIVE ERROR CRITERION
267.000  C   ACCORDING AS THE ABSOLUTE VALUE OF F(K+1) IS LESS OR GREATER
268.000  C   THAN ONE.
269.000        DF=DABS(F(K+1))
270.000        TOL=EPS*DMAX1(1.0D0,DF)
271.000        IF(DTRERR.GT.TOL) GO TO 12
272.000  C
273.000  C
274.000  C   CURRENT STEP ACCEPTED
275.000        JD=0
276.000        J=2
277.000  C
278.000  C   ***************************************************************
279.000  C   THE PHASE SHIFT IS CALCULATED IF IPS=1, WHICH MEANS THAT X IS
280.000  C   SUFFICIENTLY LARGE.
```

```
281.000           IF(IPS.EQ.1) CALL PS(K,JP,JCONV)
282.000     C   THE CALCULATION IS TERMINATED IF THE PHASE SHIFT HAS CONVERGED.
283.000           IF(JCONV.EQ.1) GO TO 30
284.000     C   ************************************************************
285.000     C
286.000     C
287.000     C   PREDICT TRUNCATION ERROR FOR THE NEXT STEP WITH A STEPLENGTH OF C*H.
288.000     C   IF THIS IS SUFFICIENTLY SMALL, GO TO 13 TO INCREASE THE STEPLENGTH.
289.000           DTRERR=DTRERR*C**5
290.000           IF(DTRERR.LT.(0.5D0*TOL)) GO TO 13
291.000           J2=0
292.000     C
293.000     C   STEPLENGTH UNCHANGED FOR NEXT STEP. UPDATE VALUES FOR NEXT STEP
294.000     C   AND RETURN TO BEGINNING OF LOOP.
295.000         8 K=K+1
296.000           H2VF=H2*V*F(K)
297.000           IF(JB.NE.1) GO TO 14
298.000           JB=2
299.000           J2=0
300.000           GO TO 20
301.000        14 B01=B0
302.000        20 YPREK=((B0-B01)*F(K-1)+B01*YK)/B0
303.000           YK=((B0-B01)*F(K)+B01*Y)/B0
304.000           B0=B01
305.000           GO TO 15
306.000     C***END OF RAPTIS AND ALLISON LOOP
307.000     C
308.000     C
309.000     C    PRINT THE FINAL VALUES OF K,NINC,NDEC AND RETURN TO CALLING PROGRAM.
310.000         6 WRITE(2,18) K,NINC,NDEC
311.000           RETURN
312.000        22 WRITE(2,24)
313.000           RETURN
314.000        30 WRITE(2,32) K,NINC,NDEC
315.000           RETURN
316.000     C
317.000     C
318.000     C    THE STATEMENTS BETWEEN LINES OF DASHES REFER TO THE CODING RELEVANT
319.000     C    TO THE CHANGE OF STEPLENGTH.
320.000     C
```

```
321.000   C ------------------------------------------------------------
322.000   C
323.000   C***STEPLENGTH DECREASE
324.000      12 JD=JD+1
325.000         C1=(0.5D0*TOL/DTRERR)**0.2D0
326.000         H=C1*H
327.000         NDEC=NDEC+1
328.000   C    IF J=0 THIS MEANS THAT H WAS TOO LARGE IN THE INITIAL STEPS SO IT IS
329.000   C    NECESSARY TO RESTART FROM X0 WITH THE NEW H.
330.000         IF(J.EQ.0) GO TO 2
331.000         IF(JB.GT.1) CALL DIVDIF(DD,K,J,JD,IPS,JB)
332.000   C    IF THE TRUNCATION ERROR PER UNIT STEP AT XX(K+1) IS GREATER
333.000   C    THAN THE TOLERANCE, THE STEPLENGTH MUST BE REDUCED TO C*H;
334.000   C    F(K+1) IS RECALCULATED FROM F(K) AND FPREK1 WITH THE NEW
335.000   C    STEPLENGTH. FPREK1 IS FOUND BY CONSTRUCTING THE DIVIDED
336.000   C    DIFFERENCE FORM OF THE 5TH DEGREE POLYNOMIAL INTERPOLATING
337.000   C    AT THE POINTS XX(K),XX(K-1),XX(K-2),XX(K-3),XX(K-4),XX(K-5).
338.000         X1=XX(K)-H
339.000         FPREK1=D(6,1)
340.000         DO 17 I=1,5
341.000         M=6-I
342.000         FPREK1=D(M,1)+(X1-XX(K-I))*FPREK1
343.000      17 CONTINUE
344.000   C
345.000         H2=H*H
346.000         IF(IPS.EQ.1) GO TO 28
347.000         GO TO 29
348.000      28 AKH=AK*H
349.000         AKH2=AKH/2.0D0
350.000         H2E=H2*E
351.000   C    UPDATE YPREK AND YK
352.000         B0=(H2E-4.0D0*DSIN(AKH2)*DSIN(AKH2))/(4.0D0*H2E*DSIN(AKH2)*DSIN(AK
353.000        1H2))
354.000      29 V=L1/(X1*X1)-E+POT(X1,IPS)
355.000         YPREK=FPREK1*(1.0D0-B0*H2*V)
356.000         X=X1+H
357.000         V=L1/(X*X)-E+POT(X,IPS)
358.000         YK=F(K)*(1.0D0-B0*H2*V)
359.000         H2VF=H2*V*F(K)
360.000         GO TO 15
```

```
361.000 C
362.000 C
363.000 C***STEPLENGTH INCREASE
364.000    13 J2=J2+1
365.000       IF(J2.LT.3) GO TO 8
366.000       J2=0
367.000       H=C*H
368.000       C2=C*C
369.000       NINC=NINC+1
370.000       H2=H*H
371.000       IF(IPS.EQ.1) GO TO 26
372.000       B01=B0
373.000       GO TO 27
374.000    26 AKH=AK*H
375.000       AKH2=AKH/2.0D0
376.000       H2E=H2*E
377.000       B01=(H2E-4.0D0*DSIN(AKH2)*DSIN(AKH2))/(4.0D0*H2E*DSIN(AKH2)*DSIN(A
378.000      1KH2))
379.000    27 YPREK=(C2*B01*YPREK+(B0-C2*B01)*F(K-1))/B0
380.000       YK=(C2*B01*Y+(B0-C2*B01)*F(K+1))/B0
381.000       B0=B01
382.000       K=K+1
383.000       H2VF=H2*V*F(K)
384.000       GO TO 15
385.000 C
386.000 C ------------------------------------------------------------------
387.000 C
388.000    18 FORMAT(1H0,52HTHE END OF THE RANGE OF INTEGRATION HAS BEEN REACHED
389.000      1/39HNUMBER OF INTEGRATION STEPS CARRIED OUT,I6/75HNUMBER OF INCREA
390.000      2SES IN STEPLENGTH REQUIRED DURING THE COURSE IF INTEGRATION,I3/75H
391.000      3NUMBER OF DECREASES IN STEPLENGTH REQUIRED DURING THE COURSE OF IN
392.000      4TEGRATION,I3,///)
393.000    24 FORMAT(1H0,113HTHE REQUIRED ACCURACY IN Y1 HAS NOT BEEN ACHIEVED;
394.000      1FURTHER COEFFICIENTS IN THE EXPANSION OF V(X) ABOUT THE ORIGIN/13H
395.000      2 ARE REQUIRED/1H )
396.000    32 FORMAT(1H0,46HPHASE SHIFT HAS CONVERGED TO REQUIRED ACCURACY/39HNU
397.000      1MBER OF INTEGRATION STEPS CARRIED OUT,I6/75HNUMBER OF INCREASES IN
398.000      2 STEPLENGTH REQUIRED DURING THE COURSE OF INTEGRATION,I3/75HNUMBER
399.000      3 OF DECREASES IN STEPLENGTH REQUIRED DURING THE COURSE OF INTEGRAT
400.000      4ION,I3,///)
```

```
401.000         END
```

```
      SUBROUTINE DIVDIF(DD,K,J,JD,IPS,JB)
C
C
C   THIS SUBROUTINE SETS UP THE TABLE OF 6TH DIVIDED DIFFERENCES
C   OF THE SOLUTION Y AT THE POINTS XX(K+1),XX(K),XX(K-1),XX(K-2),
C   XX(K-3),XX(K-4),XX(K-5) AS LONG AS THE METHOD OF NUMEROV IS USED;
C   THE RETURNED VALUE OF DD IS THE ESTIMATE OF THE 6TH DERIVATIVE
C   OF THE SOLUTION. WHEN IPS=1 AND JB>1 THIS SIGNALS THE USE OF THE
C   RAPTIS AND ALLISON METHOD AND A TABLE OF 4TH DIVIDED DIFFERENCES
C   OF (E*Y+Y'') IS SET UP AT THE POINTS XX(K+1),XX(K),XX(K-1),XX(K-2),
C   XX(K-3); THE RETURNED VALUE OF DD IS THE ESTIMATE OF THE 4TH
C   DERIVATIVE OF (E*Y+Y'').
C
        IMPLICIT REAL*8(A-H,O-Z)
        COMMON/EKLL1/E,AK,PSIG,EPS,C,L,L1
        DIMENSION X(7),D1(4,4)
        COMMON VCOEFF(8),A(9),F(8000),XX(8000),D(6,7),JY
        IF(DD.EQ.0.0D0) GO TO 30
        IF(JB.GT.1.AND.JD.GT.0) GO TO 45
        GO TO 50
C
C
C   THIS SECTION IS USED WHEN A DECREASE IN STEPLENGTH IS REQUIRED
C   DURING THE USE OF THE RAPTIS AND ALLISON METHOD.
C
   45 IF(JD.GE.2) GO TO 51
C
C   ENTRIES IN THE 4TH DIVIDED DIFFERENCE TABLE FOR (E*Y+Y'') ARE
C   STORED IN THE ARRAY D1 FOR FUTURE USE.
        DO 46 N=1,4
        I=5-N
        DO 46 M=1,N
        D1(I,M)=D(I,M)
   46 CONTINUE
C
C   A 5TH DIVIDED DIFFERENCE TABLE OF THE SOLUTION Y IS CONSTRUCTED.
        DO 47 I=1,6
        D(1,I)=F(K+I-6)
        X(I)=XX(K+I-6)
   47 CONTINUE
```

```
442.000          DO 48 N=2,6
443.000          IMAX=7-N
444.000          DO 48 I=1,IMAX
445.000          D(N,I)=(D(N-1,I)-D(N-1,I+1))/(X(I)-X(I+N-1))
446.000       48 CONTINUE
447.000    C
448.000    C     THE VALUES D(1,1),D(2,1),...,D(6,1) ARE NEEDED FOR THE
449.000    C     INTERPOLATION PROCESS IN RAPAL AND ARE STORED IN D(6,5),D(6,5),...,
450.000    C     D(6,1) RESPECTIVELY. DD IS SET TO ZERO.
451.000          DO 49 I=2,6
452.000          D(6,I)=D(7-I,1)
453.000       49 CONTINUE
454.000          DD=0.0D0
455.000          RETURN
456.000    C
457.000    C     IF SUCCESSIVE DECREASES IN STEPLENGTH ARE PERFORMED IN THE RAPTIS
458.000    C     AND ALLISON METHOD THE VALUES OF D(1,1),D(2,1),...,D(6,1) NEEDED
459.000    C     FOR THE INTERPOLATION PROCESS IN RAPAL ARE ACCESSED.
460.000       51 DO 52 I=1,5
461.000          D(I,1)=D(6,7-I)
462.000       52 CONTINUE
463.000          DD=0.0D0
464.000          RETURN
465.000    C
466.000    C
467.000       50 IF(J.NE.1) GO TO 1
468.000    C
469.000    C     THE FIRST ENTRY TO DIVDIF SETS UP A 6TH DIVIDED DIFFERENCE TABLE
470.000    C     OF THE SOLUTION AT THE FIRST SEVEN MESH POINTS.
471.000          DO 2 I=1,7
472.000       2  D(1,I)=F(I)
473.000          DO 3 N=2,6
474.000          IMAX=8-N
475.000          DO 3 I=1,IMAX
476.000          D(N,I)=(D(N-1,I)-D(N-1,I+1))/(XX(I)-XX(I+N-1))
477.000       3  CONTINUE
478.000          DD=7.2D2*(D(6,1)-D(6,2))/(XX(1)-XX(7))
479.000          J=0
480.000          RETURN
481.000    C
```

```
      1 IF(IPS.EQ.1.AND.JB.GT.1) GO TO 30
C
C   SUBSEQUENT ENTRIES TO DIVDIF CALCULATE THE 6TH DIVIDED DIFFERENCE
C   TABLE OF THE SOLUTION AS LONG AS THE METHOD OF NUMEROV IS USED.
        DO 5 I=1,7
        D(1,I)=F(K+I-6)
        X(I)=XX(K+I-6)
      5 CONTINUE
C
        IF(JD.GT.0) GO TO 8
C
        DO 6 N=2,6
        IMAX=7-N
        DO 6 I=1,IMAX
        D(N,I)=D(N,I+1)
      6 CONTINUE
C   IF THE STEPLENGTH HAS BEEN DECREASED IT IS NECESSARY TO RECALCULATE
C   SOME ENTRIES IN THE 6TH DIVIDED DIFFERENCE TABLE.
      8 DO 7 N=2,6
        I=8-N
        D(N,I)=(D(N-1,I)-D(N-1,I+1))/(X(I)-X(7))
      7 CONTINUE
        DD=7.2D2*(D(6,1)-D(6,2))/(X(1)-X(7))
        RETURN
C
C
C   THIS SECTION CALCULATES ENTRIES IN THE 4TH DIVIDED DIFFERENCE TABLE
C   FOR (E*Y+Y'').
     30 DO 35 I=1,5
        X(I)=XX(K+I-4)
        D(1,I)=(L1/(X(I)*X(I))+POT(X(I),IPS))*F(K+I-4)
     35 CONTINUE
C
        IF(JB.EQ.2) GO TO 40
        IF(JD.GT.0) GO TO 55
C
        DO 36 N=2,4
        IMAX=5-N
        DO 36 I=1,IMAX
        D(N,I)=D(N,I+1)
```

```
522.000        36 CONTINUE
523.000           GO TO 57
524.000     C
525.000     C    THE PREVIOUS ENTRY TO DIVDIF OBTAINED VALUES FOR INTERPOLATION
526.000     C    IN RAPAL AND ENTRIES IN THE 4TH DIVIDED DIFFERENCE TABLE FOR
527.000     C    (E*Y+Y'') MUST BE RECALLED FROM THE ARRAY D1.
528.000        55 DO 56 N=1,4
529.000           I=5-N
530.000           DO 56 M=1,N
531.000           D(I,M)=D1(I,M)
532.000        56 CONTINUE
533.000        57 DO 37 N=2,4
534.000           I=6-N
535.000           D(N,I)=(D(N-1,I)-D(N-1,I+1))/(X(I)-X(5))
536.000        37 CONTINUE
537.000           DD=2.4D1*(D(4,1)-D(4,2))/(X(1)-X(5))
538.000           RETURN
539.000     C
540.000     C    CALCULATE 4TH DIVIDED DIFFERENCE TABLE ON FIRST ENTRY TO DIVDIF
541.000     C    WITH RAPTIS AND ALLISON METHOD.
542.000        40 DO 33 N=2,4
543.000           IMAX=6-N
544.000           DO 33 I=1,IMAX
545.000           D(N,I)=(D(N-1,I)-D(N-1,I+1))/(X(I)-X(I+N-1))
546.000        33 CONTINUE
547.000           JB=3
548.000           DD=2.4D1*(D(4,1)-D(4,2))/(X(1)-X(5))
549.000           RETURN
550.000           END
```

```
      SUBROUTINE PS(K,JP,JCONV)
C
C
C     THIS SUBROUTINE CALCULATES THE PHASE SHIFT FROM THE SOLUTION OF
C     THE SCHRODINGER EQUATION AT TWO POINTS XM AND X AT LEAST ONE
C     ATOMIC UNIT APART.
C     Y IS THE VALUE OF THE WAVE FUNCTION AT X
C     JP=1 ON FIRST ENTRY TO THIS SUBROUTINE AND =2 ON LATER ENTRIES
C     JCONV IS SET EQUAL TO ZERO IN THE CALLING PROGRAM. IT IS CHANGED
C     TO ONE WHEN THE RELATIVE DIFFERENCE BETWEEN TWO SUCCESSIVE
C     ESTIMATES OF THE PHASE SHIFT IS <=PSIG
C
C     COMMON BLOCKS:
C       EKLL1 TRANSFERS ENERGY, WAVE NUMBER, ANGULAR MOMENTUM, THE
C             STEPLENGTH PARAMETER C AND THE TOLERANCE PARAMETERS
C             PSIG AND EPS FROM THE MAIN PROGRAM.
C       PHASE WHICH IS USED ONLY IN THIS SUBROUTINE STORES DATA FROM
C             THE PRECEEDING ENTRY TO THE SUBROUTINE.
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/EKLL1/E,AK,PSIG,EPS,C,L,L1/PHASE/XM,YM,SM,CM,PSM
      COMMON VCOEFF(8),A(9),F(8000),XX(8000),D(6,7),JY
      X=XX(K+1)
      Y=F(K+1)
      IF (JP.EQ.1) GO TO 1
C     THE FUNCTION OF THE SUBROUTINE ON THE FIRST ENTRY IS TO STORE DATA
C     WHICH WILL BE USED LATER IN A PHASE-SHIFT CALCULATION.
      XINC=X-XM
      IF(XINC.LT.1.0) RETURN
      AKX=AK*X
      S1=REG(AKX,L)
      C1=AIREG(AKX,L)
      ANUM=Y*SM-YM*S1
      ADENOM=YM*C1-Y*CM
      A1=ANUM/ADENOM
      PSHIFT=DATAN(A1)
      WRITE(2,5) X,PSHIFT
      P=DABS((PSHIFT-PSM)/PSHIFT)
      IF (P.LT.PSIG) JCONV=1
```

```
      C
      C     STORAGE OF DATA FOR NEXT PHASE-SHIFT CALCULATION
    2 PSM=PSHIFT
      XM=X
      YM=Y
      SM=S1
      CM=C1
      C
    4 FORMAT(1HO,5X,1HX,11X,11HPHASE SHIFT)
    5 FORMAT(1H ,D11.4,5X,D13.6)
      RETURN
      C
      C     THIS SECTION IS USED ONLY ON THE FIRST ENTRY
    1 AKX=AK*X
      S1=REG(AKX,L)
      C1=AIREG(AKX,L)
      PSHIFT=0.0D0
      JP=2
      WRITE(2,4)
      GO TO 2
      END
```

REAL FUNCTION PUT*8(X,IPS)

LINE NUMBER TEXT PAGE 1

```
      REAL FUNCTION PUT*8(X,IPS)
C
C   THIS SUBPROGRAM RETURNS THE VALUE OF THE POTENTIAL V(X).
C   IPS IS SET EQUAL TO ZERO IN THE CALLING PROGRAM AND IS CHANGED TO
C       1 WHEN THE CLASSICAL REGION HAS BEEN REACHED.
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/ZERLI/L,AK,PSIG,EPS,C,L,L1
C   EPS IS THE TOLERANCE PARAMETER USED IN KAPAL.
C   PSIG IS NOT USED IN THIS SUBPROGRAM.
      V=-2.0D0*(1.0D0+1.0D0/X)*DEXP(-2.0D0*X)
C   HERE V IS THE POTENTIAL FOR AN ELECTRON IN THE STATIC FIELD
C   OF THE HYDROGEN ATOM.
      A=L1/(X*X)-L
      IF(A.LT.0.0D0.AND.DABS(V).LT.(-A)) IPS=1
      PUT=V
      RETURN
      END
```

REAL FUNCTION REG*8(X,L)

LINE NUMBER		TEXT	PAGE 1
631.000		REAL FUNCTION REG*8(X,L)	
632.000	C	SPHERICAL BESSEL FUNCTION OF THE FIRST KIND TIMES X.	
633.000	C	REG(X,L)=X*JL(X) IN THE NOTATION OF ABRAMOWITZ AND	
634.000	C	STEGUN HANDBOOK OF MATHEMATICAL FUNCTIONS P437.	
635.000	C	IF L DOES NOT EXCEED X FORWARD RECURRENCE IS USED, OTHERWISE	
636.000	C	BACKWARD RECURRENCE IS USED WITH STARTING CONDITIONS FOUND AS	
637.000	C	SUGGESTED BY CORBATO AND URETSKY J. ASSOC. COMP. MACH. VOL 6,	
638.000	C	PP 366-375 (1959).	
639.000		IMPLICIT REAL*8(A-H,O-Z)	
640.000		DIMENSION PF(40)	
641.000		IF (L.GT.X) GO TO 3	
642.000	C		
643.000	C	FORWARD RECURRENCE	
644.000		A=DSIN(X)	
645.000		IF (L.EQ.0) GO TO 2	
646.000		B=DCOS(X)	
647.000		FAC=1.0D0/X	
648.000		X2=FAC+FAC	
649.000		DO 1 J=1,L	
650.000		A1=A	
651.000		A=FAC*A-B	
652.000		B=A1	
653.000		FAC=FAC+X2	
654.000		1 CONTINUE	
655.000		2 REG=A	
656.000		RETURN	
657.000	C		
658.000	C	**********	
659.000	C		
660.000	C	A RELATIVE ERROR LESS THAN 1.0D-7 IS DEMANDED WHEN BACKWARD	
661.000	C	RECURRENCE IS USED. IF THE DIMENSION OF THE ARRAY PF IS	
662.000	C	INSUFFICIENT TO ENSURE THIS ACCURACY A WARNING IS PRINTED.	
663.000		3 IF (L.GE.40) GO TO 6	
664.000		ERLG=23.25D0	
665.000	C	ERLG IS THE ABSOLUTE VALUE OF THE LOG TO THE BASE 2 OF DELTA	
666.000	C	WHERE DELTA IS THE MAXIMUM PERMITTED RELATIVE ERROR IN REG.	
667.000	C	HERE DELTA=1.0D-7.	
668.000	C	NU1 IS DEFINED BY EQN(31) OF CORBATO AND URETSKY.	
669.000		U=2.0D0*X/DFLOAT(2*L+1)	
670.000		NU1=L+IDINT(ERLG*(0.1D0+0.175D0*U*(2.0D0-U*U)/(1.0D0-U*U)))	

```
671.000          NP=IDINT(X-0.5D0+DSQRT(ERLG*0.35D0*X))
672.000          IF (NP.LT.L) GO TO 4
673.000          U=2.0D0*X/DFLOAT(2*NP+1)
674.000          NUP=NP+IDINT(ERLG*(0.1D0+0.175D0*U*(2.0D0-U*U)/(1.0D0-U*U)))
675.000    C     BACKWARD RECURRENCE BEGINS WITH THE SMALLER OF NU1 AND NUP.
676.000          IF (NUP.LT.NU1) NU1=NUP
677.000    C     REDUCED ACCURACY WHEN NU1 TOO LARGE FOR DECLARED ARRAY
678.000          IF (NU1.GE.40) GO TO 7
679.000        4 PF(NU1+1)=1.0D-10
680.000          FAC=DFLOAT(NU1+NU1+1)/X
681.000          PF(NU1)=FAC*PF(NU1+1)
682.000          X2=2.0D0/X
683.000          J1=NU1-1
684.000          DO 5 J=1,J1
685.000             FAC=FAC-X2
686.000             PF(NU1-J)=FAC*PF(NU1+1-J)+PF(NU1+2-J)
687.000        5 CONTINUE
688.000          CRN=(PF(1)/X-PF(2))*DCOS(X)+DSIN(X)*PF(1)
689.000          REG=PF(L+1)/CRN
690.000          RETURN
691.000    C
692.000        6 WRITE(2,100) L
693.000      100 FORMAT(1H ,3HL= ,I3,52H TOO LARGE FOR ARRAY DECLARED.  REG REPLACE
694.000         1D BY ZERO)
695.000          REG=0.0D0
696.000          RETURN
697.000    C
698.000        7 WRITE(2,101) L,NU1
699.000      101 FORMAT(1H ,31HFOR BACKWARD RECURRENCE WITH L=,I3,32H THE CALCULATE
700.000         1D VALUE OF NU1 IS ,I3,24H SINCE THIS IS TOO LARGE/45H FOR THE DECL
701.000         2ARED ARRAY NU1 IS REPLACED BY 39)
702.000          NU1=39
703.000          GO TO 4
704.000          END
```

REAL FUNCTION AIREG*8(X,L)

LINE NUMBER		TEXT	PAGE 1
705.000		REAL FUNCTION AIREG*8(X,L)	
706.000	C	SPHERICAL BESSEL FUNCTION OF THE SECOND KIND TIMES -X.	
707.000	C	AIREG(X,L)=-X*YL(X) IN THE NOTATION OF ABRAMOWITZ AND	
708.000	C	STEGUN HANDBOOK OF MATHEMATICAL FUNCTIONS P437.	
709.000	C	CALCULATION IS BY FORWARD RECURRENCE.	
710.000	C	NO OVERFLOW PROTECTION IS INCLUDED SINCE THIS ROUTINE	
711.000	C	WILL NOT BE CALLED FOR VERY SMALL VALUES OF X	
712.000	C		
713.000		IMPLICIT REAL*8(A-H,O-Z)	
714.000		A=DCOS(X)	
715.000		IF (L.EQ.0) GO TO 2	
716.000		FAC=1.0D0/X	
717.000		B=-DSIN(X)	
718.000		X2=FAC+FAC	
719.000		DO 1 J=1,L	
720.000		A1=A	
721.000		A=FAC*A-B	
722.000		FAC=FAC+X2	
723.000		B=A1	
724.000	1	CONTINUE	
725.000	2	AIREG=A	
726.000		RETURN	
727.000		END	

```
      SUBROUTINE DEVOG(H,K2,K3,XF)
C
C
C   INTEGRATES NP SECOND ORDER COUPLED DIFFERENTIAL EQUATIONS
C   USING DE VOGELAERE'S METHOD SUBJECT TO A SET OF K3 INITIAL
C   BOUNDARY CONDITIONS.
C
C   H IS THE STEPLENGTH SUPPLIED BY THE USER.
C
C   AT EACH STEP THE LOCAL TRUNCATION ERROR IN EACH COMPONENT
C   OF THE SOLUTION Y(I,J),I=1,---,NP,J=1,---,K3 IS ESTIMATED.
C   IF THIS ESTIMATE FOR ANY COMPONENT OF Y IS TOO LARGE THE CURRENT
C   STEP IS DISCARDED AND A SMALLER STEPLENGTH IS CHOSEN. WHEN THE
C   ESTIMATED TRUNCATION ERROR IS SUFFICIENTLY SMALL IN EACH
C   COMPONENT OF Y THE STEPLENGTH IS INCREASED FOR THE NEXT STEP.
C
C   K2 IS THE INDEX NUMBER OF THE BOUND-STATE ORBITAL WHICH IS TO
C   BE USED TO GENERATE THE K2TH INHOMOGENEOUS SOLUTION.
C
C   K3 IS THE TOTAL NUMBER OF LINEARLY INDEPENDENT SOLUTIONS
C   GENERATED OVER THE RANGE OF INTEGRATION; IN THE NOTATION OF
C   CHANDRA (1973) K3=NTOT1=NP+NBND IN THE OUTWARD INTEGRATION AND
C   K3=NTOT2=NP+NA+NBND IN THE INWARD INTEGRATION. ON EXIT FROM DEVOG
C   K3 CONTAINS THE NUMBER OF STEPS CARRIED OUT DURING THE COURSE OF
C   THE INTEGRATION.
C
C   XF IS THE ENDPOINT OF THE RANGE OF INTEGRATION; THE SOLUTIONS OF
C   THE OUTWARD AND INWARD INTEGRATIONS ARE MATCHED AT XF.
C
C
C   COMMON BLOCKS:
C
C    CE      :TRANSFERS THE USER SUPPLIED FACTOR C (E.G.2) AND THE
C             TOLERANCE PARAMETER EPS FROM THE MAIN PROGRAM.
C             THE STEPLENGTH IS INCREASED BY THE FACTOR C WHEN THIS
C             CAN BE DONE WITHOUT EXCEEDING THE BOUND EPS WHICH
C             IS AN UPPER BOUND ON THE ESTIMATED TRUNCATION ERROR PER
C             UNIT STEP IN EACH COMPONENT OF THE SOLUTION Y(I,J).
C    BLCK 1 :ON ENTRY TO DEVOG THE INITIAL VALUE OF THE STEPLENGTH
C             IS H0; ON EXIT H0 IS THE STEPLENGTH H USED IN THE FINAL
```

```
41.000   C            STEP OF THE INTEGRATION OR IF H HAS BEEN DECREASED HO IS
42.000   C            THE VALUE OF H USED IN THE PENULTIMATE STEP.
43.000   C     BLCK 2 :NP IS THE TOTAL NUMBER OF COUPLED EQUATIONS
44.000   C            NA IS THE NUMBER OF OPEN CHANNELS
45.000   C            NBND IS THE TOTAL NUMBER OF BOUND-STATE ORBITALS
46.000   C            GIVING RISE TO INHOMOGENEOUS TERMS.
47.000   C            WP ARRAY STORES THE ENERGY VALUES (IN ATOMIC UNITS)
48.000   C            FOR NP EQUATIONS; FIRST FOR NA OPEN AND THEN FOR NB=(NP-NA)
49.000   C            CLOSED CHANNELS.
50.000   C            LP ARRAY STORES THE ANGULAR MOMENTA VALUES FOR NP EQUATIONS.
51.000   C     BLCK 6 :X IS THE CURRENT VALUE OF THE MESH POINT.
52.000   C            Y0(I,J),I=1,---,NP,J=1,---,K3 IS THE INITIAL VALUE OF
53.000   C            THE SOLUTION AT X0.
54.000   C            Z0(I,J),I=1,---,NP,J=1,---,K3 IS THE INITIAL VALUE
55.000   C            OF THE FIRST DERIVATIVE OF THE SOLUTION AT X0.
56.000   C            F0(I,J),I=1,---,NP,J=1,---,K3 IS THE INITIAL VALUE
57.000   C            OF THE SECOND DERIVATIVE OF THE SOLUTION AT X0.
58.000   C            Y(I,J),Z(I,J),F(I,J),I=1,---,NP,J=1,---,K3 ARE THE
59.000   C            CURRENT VALUES OF THE SOLUTION AND ITS FIRST AND SECOND
60.000   C            DERIVATIVES RESPECTIVELY.
61.000   C     BLCK 7 :W ARRAY STORES THE MESH POINTS WHICH ARE CHOSEN
62.000   C            AUTOMATICALLY OVER THE RANGE OF INTEGRATION.
63.000   C            FUN(I,J,K),I=1,---,NP,J=1,---,K3,K=1,---,KO OR KI,
64.000   C            WHERE KO,KI ARE THE NUMBER OF INTEGRATION STEPS CARRIED
65.000   C            OUT IN THE OUTWARD,INWARD INTEGRATIONS RESPECTIVELY,
66.000   C            STORES THE SOLUTION AT THE CURRENT MESH POINT W(K).
67.000   C     BLCK 8 :KO IS DESCRIBED ABOVE.
68.000   C            NTOT1 IS DESCRIBED ABOVE.
69.000   C            FB1(I,J) IS THE SOLUTION OF THE ITH EQUATION FOR THE JTH
70.000   C            BOUNDARY CONDITION IN THE OUTWARD INTEGRATION AT XF.
71.000   C            FDR1(I,J) IS THE DERIVATIVE OF THE ITH EQUATION FOR THE
72.000   C            JTH BOUNDARY CONDITION IN THE OUTWARD INTEGRATION AT XF.
73.000   C            WO(K) STORES THE MESH POINTS USED IN THE OUTWARD
74.000   C            INTEGRATION FOR USE IN SUBROUTINE SIMPSN.
75.000   C            FUNO(I,J,K),I=1,---,NP,J=1,---,K3,K=1,---,KO STORES
76.000   C            THE SOLUTION IN THE OUTWARD INTEGRATION AT THE
77.000   C            CORRESPONDING MESH POINT WO(K) FOR USE IN SUBROUTINE SIMPSN.
78.000   C     BLCK 9 :KO IS DESCRIBED ABOVE.
79.000   C            NTOT2 IS DESCRIBED ABOVE.
80.000   C            FB2(I,J) IS THE SOLUTION OF THE ITH EQUATION FOR THE JTH
```

```
C             BOUNDARY CONDITION IN THE INWARD INTEGRATION AT XF.
C             FDR2(I,J) IS THE DERIVATIVE OF THE ITH EQUATION FOR THE
C             JTH BOUNDARY CONDITION IN THE INWARD INTEGRATION AT XF.
C             WI(K) STORES THE MESH POINTS USED IN THE INWARD INTEGRATION
C             FOR USE IN SUBROUTINE SIMPSN.
C             FUNI(I,J,K),I=1,---,NP,J=1,---,K3,K=1,   ,KI STORES
C             THE SOLUTION IN THE INWARD INTEGRATION AT THE CORRESPONDING
C             MESH POINT WI(K) FOR USE IN SUBROUTINE SIMPSN.
C     INFORM :IREAD CORRESPONDS TO THE UNIT USED FOR CARD READING.
C             IPRINT CORRESPONDS TO THE UNIT USED FOR CARD PUNCHING.
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION FEVEN(5,12),FODD(5,12),F2I(5,12),YODD(5,12)
      DIMENSION YODD1(5,12),Y1(5,12),Z1(5,12),FEVEN1(5,12),FODD1(5,12)
      DIMENSION F2I1(5,12),TR1(5,12),TR2(5,12),TR21(5,12),TR3(5,12)
      DIMENSION TRERR(5,12),DTRERR(5,12),TOL(5,12)
      COMMON/CEPS /C,EPS
      COMMON/BLCK 1/H0,X0,R0,RA
      COMMON/BLCK 2/WP(5),NP,NA,NBND,LP(5)
      COMMON/BLCK 6/X,Y0(5,12),Z0(5,12),F0(5,12),Y(5,12),Z(5,12),F(5,12)
      COMMON/BLCK 7/W(2000),FUN(5,12,2000)
      COMMON/BLCK 8/K0,NTOT1,FB1(5,7),FDR1(5,7),WO(2000),FUNO(5,12,2000)
      COMMON/BLCK 9/K1,NTOT2,FB2(5,12),FDR2(5,12),WI(2000),FUN1(5,12,200
     10)
      COMMON/INFORM/IREAD,IPRINT
C
C   NINC COUNTS THE NUMBER OF INCREASES IN STEPLENGTH REQUIRED
C   DURING THE COURSE OF INTEGRATION.
      NINC=0
C   NDEC COUNTS THE NUMBER OF DECREASES IN STEPLENGTH REQUIRED
C   DURING THE COURSE OF INTEGRATION.
      NDEC=0
C   THE INTEGER ISTEP COUNTS THE NUMBER OF STEPS CARRIED OUT
      ISTEP=0
C
C
C   DESCRIPTION AND INITIALIZATION OF VARIABLES REQUIRED FOR
C   BOOK-KEEPING AND ERROR CONTROL.
C
```

```
121.000 C   THE INTEGER KD IS INITIALLY SET TO ZERO AND IS SET TO ONE
122.000 C   WHEN THE ESTIMATED TRUNCATION ERROR IN ANY ONE OF THE COMPONENTS
123.000 C   OF THE SOLUTION IS TOO LARGE; KD IS RESET TO ZERO UNTIL THE ABOVE
124.000 C   CONDITION IS NEENCOUNTERED.
125.000       KD=0
126.000 C   THE INTEGER JHD IS INITIALLY SET TO ZERO; IF THE NEXT STEP WITH
127.000 C   THE CURRENT VALUE OF H WOULD TAKE THE CALCULATION BEYOND XF
128.000 C   A DECREASE IN STEPLENGTH IS FORCED SO THAT THE NEXT STEP ENDS
129.000 C   PRECISELY AT XF AND JHD IS SET TO 1.
130.000       JHD=0
131.000 C   THE INTEGER K IS INITIALLY SET TO 1; W(1) REPRESENTS THE FIRST
132.000 C   MESH POINT X0 AND W(K) REPRESENTS THE MESH POINT X0+(K-1)H.
133.000       5 K=1
134.000 C   J1 AND JS  REPRESENT RESPECTIVELY THE NUMBER OF SUCCESSFUL STEPS
135.000 C                 CARRIED OUT WITH THE CURRENT H AND THE PREVIOUS
136.000 C                 SUCCESSFUL H.
137.000       J1=0
138.000       JS=0
139.000 C   J2: THE STEPLENGTH IS INCREASED FROM H TO C*H IF FOR 3 SUCCESSIVE
140.000 C       STEPS THE PREDICTED TRUNCATION ERROR WITH A STEPLENGTH OF C*H IS
141.000 C       SUFFICIENTLY SMALL.  J2 IS USED AS A COUNTER FOR THIS PURPOSE.
142.000       J2=0
143.000 C   JD: WHEN A STEPLENGTH DECREASE IS NECESSARY A NEW H IS CHOSEN.
144.000 C       CALCULATIONS MAY SHOW THAT THIS IS INADEQUATE, SO A FURTHER
145.000 C       REDUCTION IS REQUIRED.  JD KEEPS TRACK OF THE STEPLENGTH
146.000 C       DECREASES IN SUCH A CASE.  IT IS INCREASED BY 1 WITH EACH
147.000 C       DECREASE AND IS SET TO ZERO WHEN AN ACCEPTABLE H HAS BEEN FOUND.
148.000       JD=0
149.000 C   C1  IS THE RATIO OF THE CURRENT H TO THE PREVIOUS SUCCESSFUL VALUE.
150.000       C1=1.0D0
151.000       C2=C1*C1
152.000 C
153.000       DO 35 J=1,K3
154.000       DO 35 I=1,NP
155.000       TR1(I,J)=1.0D2
156.000    35 CONTINUE
157.000 C
158.000       HMAX=(XF-W(1))/5.0D0
159.000 C   IF A STEPLENGTH GREATER THAN HMAX WERE USED THE TRUNCATION ERROR
160.000 C   WOULD NOT BE ESTIMATED BEFORE XF IS REACHED.
```

```
161.000           DH=DABS(H)
162.000           DHX=DABS(HMAX)
163.000           IF(DH.GT.DHX) H=HMAX
164.000     C
165.000           WRITE(IPRINT,140)
166.000           WRITE(IPRINT,143) W(1),H,((FUN(I,J,1),I=1,NP),J=1,K3)
167.000     C
168.000     C***INITIAL STEP
169.000           H2=H*H
170.000           DO 36 J=1,K3
171.000           DO 36 I=1,NP
172.000           Z(I,J)=H*ZO(I,J)
173.000           FEVEN(I,J)=H2*FO(I,J)/3.0D0
174.000           F2I(I,J)=FEVEN(I,J)
175.000           Y(I,J)=YO(I,J)+Z(I,J)+1.5D0*FEVEN(I,J)
176.000        36 CONTINUE
177.000           X=XO+H
178.000           K=K+1
179.000           W(K)=X
180.000           CALL DRV2(K2,K3)
181.000           DO 37 J=1,K3
182.000           DO 37 I=1,NP
183.000           FODD(I,J)=4.0D0*H2*F(I,J)/3.0D0
184.000           Y(I,J)=Y(I,J)+(FODD(I,J)-4.0D0*FEVEN(I,J))/8.0D0
185.000        37 CONTINUE
186.000           CALL DRV2(K2,K3)
187.000           DO 38 J=1,K3
188.000           DO 38 I=1,NP
189.000           FUN(I,J,K)=Y(I,J)
190.000           FODD(I,J)=4.0D0*H2*F(I,J)/3.0D0
191.000        38 CONTINUE
192.000           X=X+H
193.000           K=K+1
194.000           W(K)=X
195.000           DO 39 J=1,K3
196.000           DO 39 I=1,NP
197.000           Z1(I,J)=Z(I,J)+FEVEN(I,J)
198.000           Y(I,J)=YO(I,J)+2.0D0*Z1(I,J)+FODD(I,J)
199.000        39 CONTINUE
200.000           CALL DRV2(K2,K3)
```

```
201.000          DO 40 J=1,K3
202.000          DO 40 I=1,NP
203.000          FUN(I,J,K)=Y(I,J)
204.000          FEVEN(I,J)=H2*F(I,J)/3.0D0
205.000          Z(I,J)=Z1(I,J)+FODD(I,J)+FEVEN(I,J)
206.000       40 CONTINUE
207.000          IF(ISTEP.GT.0) GO TO 4
208.000          DO 41 J=1,K3
209.000          DO 41 I=1,NP
210.000          YODD1(I,J)=Y(I,J)-Z(I,J)+(1.4D1*FEVEN(I,J)+3.0D0*FODD(I,J)-2.0D0*F
211.000         12I(I,J))*6.25D-2
212.000       41 CONTINUE
213.000          X=W(2)
214.000          DO 42 J=1,K3
215.000          DO 42 I=1,NP
216.000          Y(I,J)=YODD1(I,J)
217.000       42 CONTINUE
218.000          CALL DRV2(K2,K3)
219.000          X=W(K)
220.000          DO 43 J=1,K3
221.000          DO 43 I=1,NP
222.000          FODD1(I,J)=4.0D0*H2*F(I,J)/3.0D0
223.000          Y(I,J)=FUN(I,J,K)
224.000          COLD=C1
225.000          Y1ERR=0.75D0*(FODD1(I,J)-FODD(I,J))/H
226.000          DY1ERR=DABS(Y1ERR)
227.000          DY=DABS(Y(I,J))
228.000          TOL(I,J)=EPS*DMAX1(1.0D0,DY)
229.000          IF(DY1ERR.GT.TOL(I,J)) KD=1
230.000          IF(DY1ERR.GT.TOL(I,J)) C1=(0.5D0*EPS/DY1ERR)**0.2D0
231.000          IF(COLD.LT.C1) C1=COLD
232.000       43 CONTINUE
233.000          IF(KD.EQ.0) GO TO 4
234.000          KD=0
235.000          H=C1*H
236.000          WRITE(IPRINT,142)
237.000          GO TO 5
238.000        4 WRITE(IPRINT,143) W(K),H,((FUN(I,J,K),I=1,NP),J=1,K3)
239.000          J1=J1+1
240.000          DO 44 J=1,K3
```

```
241.000          DO 44 I=1,NP
242.000          F2I(I,J)=FEVEN(I,J)
243.000       44 CONTINUE
244.000          ERRFAC=4.0D0/(4.5D1*H)
245.000    C    ERRFAC IS A FACTOR WHICH APPEARS IN THE TRUNCATION-ERROR ESTIMATE.
246.000          X=X+H
247.000          K=K+1
248.000          W(K)=X
249.000    C
250.000          ISTEP=1
251.000    C
252.000    C
253.000    C***GENERAL DE VUGELAERE LOOP
254.000        3 DO 45 J=1,K3
255.000          DO 45 I=1,NP
256.000          Z(I,J)=Z(I,J)+FEVEN(I,J)
257.000          YODD(I,J)=Y(I,J)+Z(I,J)
258.000          Y(I,J)=YODD(I,J)+FEVEN(I,J)-0.125D0*FODD(I,J)
259.000       45 CONTINUE
260.000          CALL DRV2(K2,K3)
261.000          DO 46 J=1,K3
262.000          DO 46 I=1,NP
263.000          FUN(I,J,K)=Y(I,J)
264.000          FODD(I,J)=4.0D0*H2*F(I,J)/3.0D0
265.000       46 CONTINUE
266.000          X=X+H
267.000          K=K+1
268.000          W(K)=X
269.000          DO 47 J=1,K3
270.000          DO 47 I=1,NP
271.000          Z(I,J)=Z(I,J)+FODD(I,J)
272.000          Y(I,J)=YODD(I,J)+Z(I,J)
273.000       47 CONTINUE
274.000          CALL DRV2(K2,K3)
275.000          DO 48 J=1,K3
276.000          DO 48 I=1,NP
277.000          FUN(I,J,K)=Y(I,J)
278.000          FEVEN(I,J)=H2*F(I,J)/3.0D0
279.000          Z(I,J)=Z(I,J)+FEVEN(I,J)
280.000          YODD1(I,J)=Y(I,J)-Z(I,J)+(1.4D1*FEVEN(I,J)+3.0D0*FODD(I,J)-2.0D0*F
```

```
281.000          12I(I,J))*6.25D-2
282.000           TR2(I,J)=YODD1(I,J)-FUN(I,J,K-1)
283.000           TR21(I,J)=TR2(I,J)
284.000        48 CONTINUE
285.000           IF(ISTEP.LE.2) GO TO 32
286.000           IF(J1.EQ.1.OR.(J1.EQ.0.AND.JS.EQ.1)) GO TO 30
287.000           GO TO 32
288.000        30 DO 49 J=1,K3
289.000           DO 49 I=1,NP
290.000           TR2(I,J)=BETA*TR21(I,J)
291.000           IF(J1.EQ.0) TR2(I,J)=BETA1*TR21(I,J)
292.000        49 CONTINUE
293.000        32 C11=C1
294.000           C21=C2
295.000           DO 56 J=1,K3
296.000           DO 56 I=1,NP
297.000           COLD=C1
298.000           TRERR(I,J)=(TR2(I,J)-TR1(I,J))*ERRFAC
299.000           DTRERR(I,J)=DABS(TRERR(I,J))
300.000           DY=DABS(Y(I,J))
301.000           TOL(I,J)=EPS*DMAX1(1.0D0,DY)
302.000           IF((DTRERR(I,J).GT.TOL(I,J)).AND.ISTEP.GT.1) KD=1
303.000           IF((DTRERR(I,J).GT.TOL(I,J)).AND.ISTEP.GT.1) C1=(0.5D0*TOL(I,J)/DT
304.000          1RERR(I,J))**0.25D0
305.000           IF(COLD.LT.C1) C1=COLD
306.000        56 CONTINUE
307.000           WRITE(IPRINT,141) W(K),H,((Y(I,J),TRERR(I,J),I=1,NP),J=1,K3)
308.000     C    IF TRUNCATION ERROR IS TOO LARGE GO TO 14.  THIS DOES NOT APPLY TO
309.000     C    THE FIRST STEP SINCE TWO STEPS ARE REQUIRED FOR AN ERROR ESTIMATE.
310.000     C    THE PARAMETER TOL PROVIDES AN ABSOLUTE OR RELATIVE ERROR CRITERION
311.000     C    ACCORDING AS THE ABSOLUTE VALUE OF Y IS LESS OR GREATER THAN ONE.
312.000           IF(KD.EQ.1) GO TO 14
313.000     C
314.000     C
315.000     C    CURRENT STEP ACCEPTED.  DATA STORED FROM PREVIOUS STEP UPDATED.
316.000           JD=0
317.000           J1=J1+1
318.000           DO 50 J=1,K3
319.000           DO 50 I=1,NP
320.000           TR1(I,J)=TR21(I,J)
```

```
321.000          F2(I,J)=FEVEN(I,J)
322.000          Z1(I,J)=Z(I,J)
323.000          FEVEN1(I,J)=FEVEN(I,J)
324.000          FODD1(I,J)=FODD(I,J)
325.000          Y1(I,J)=Y(I,J)
326.000       50 CONTINUE
327.000          IF(JHD.EQ.0) H1=H
328.000    C     UPDATE ERRFAC IF NECESSARY
329.000          IF(J1.GT.2) GO TO 31
330.000          ERRFAC=4.0D0/(4.5D1*H)
331.000          IF(J1.EQ.1) ERRFAC=0.8D0*(C2/H)/(3.0D0*C2+5.0D0*C1+1.0D0)
332.000       31 ISTEP=ISTEP+1
333.000    C
334.000    C
335.000    C     PREDICT TRUNCATION ERROR FOR THE NEXT STEP WITH A STEPLENGTH OF C*H.
336.000    C     IF THIS IS SUFFICIENTLY SMALL GO TO 13 TO INCREASE THE STEPLENGTH.
337.000       10 DO 51 J=1,K3
338.000          DO 51 I=1,NP
339.000          DTRERR(I,J)=DTRERR(I,J)*C**4
340.000          IF(DTRERR(I,J).GT.(0.5D0*TOL(I,J))) GO TO 25
341.000       51 CONTINUE
342.000          GO TO 13
343.000       25 J2=0
344.000       23 IF(JHD.EQ.1) GO TO 2
345.000          XXF=XF-X
346.000          XXF2=XXF/2.0D0
347.000          DXXF=DABS(XXF)
348.000          IF(DXXF.LT.1.0D-8) GO TO 2
349.000    C     IF THE NEXT STEP WOULD TAKE THE CALCULATION BEYOND XF REDUCE THE
350.000    C     STEPLENGTH; THE QUANTITY (XF-X-2H)H IS POSITIVE UNTIL
351.000    C     THE INTEGRATION PROCEEDS BEYOND XF. EXIT IF X EQUALS XF.
352.000          IF((XXF-2.0D0*H)*H.GE.0.0D0) GO TO 60
353.000          J2=0
354.000          JHD=1
355.000          C1=XXF2/H
356.000          H=XXF2
357.000          C2=C1*C1
358.000          GO TO 16
359.000    C     STEPLENGTH UNCHANGED FOR NEXT STEP. UPDATE X AND RETURN TO BEGINNING
360.000    C     OF LOOP.
```

```
361.000      60 K=K+1
362.000         X=X+H
363.000         W(K)=X
364.000         GO TO 3
365.000   C***END OF DE VOGELAERE LOOP
366.000   C
367.000   C
368.000       2 K3=K
369.000         HO=H
370.000         IF(C1.LT.1.0D0) HO=H1
371.000   C    PRINT THE FINAL VALUES OF ISTEP,NINC,NDEC AND RETURN TO CALLING
372.000   C    PROGRAM.
373.000         WRITE(IPRINT,112) ISTEP,NINC,NDEC
374.000         RETURN
375.000   C
376.000   C
377.000   C    THE CODING RELEVANT TO THE CHANGE OF STEPLENGTH IS CONTAINED
378.000   C    BETWEEN LINES OF DASHES.
379.000   C
380.000   C ------------------------------------------------------------------
381.000   C
382.000   C
383.000   C***STEPLENGTH DECREASE
384.000      14 JD=JD+1
385.000         JHD=0
386.000         KD=0
387.000         IF(J1.EQ.0) GO TO 20
388.000         JS=J1
389.000         J1=0
390.000         DO 52 J=1,K3
391.000         DO 52 I=1,NP
392.000         TR3(I,J)=TR1(I,J)
393.000         F211(I,J)=F21(I,J)
394.000      52 CONTINUE
395.000      20 X=X-2.0D0*H
396.000         K=K-2
397.000         W(K)=X
398.000         H=C1*H
399.000         NDEC=NDEC+1
400.000         WRITE(IPRINT,142)
```

```
401.000 C    IF I=2 ENTRY TO THIS SECTION MEANS THAT H WAS TOO LARGE IN THE LAST
402.000 C    TWO STEPS SO IT IS NECESSARY TO RESTART FROM XO WITH THE NEW H
403.000        IF(ISTEP.EQ.2) GO TO 5
404.000        DO 53 J=1,K3
405.000        DO 53 I=1,NP
406.000        Z(I,J)=Z1(I,J)
407.000        F2I(I,J)=F2I1(I,J)
408.000        FEVEN(I,J)=FEVEN1(I,J)
409.000        FODD(I,J)=FODD1(I,J)
410.000        Y(I,J)=Y1(I,J)
411.000     53 CONTINUE
412.000 C    C1 NOW BECOMES THE RATIO OF THE NEW H TO THE LAST SUCCESSFUL H.
413.000        C1=H/H1
414.000        C2=C1*C1
415.000 C    NOW ENTER RESTART SECTION
416.000        GO TO 16
417.000 C
418.000 C
419.000 C***STEPLENGTH INCREASE
420.000 C    NO ACTION TAKEN UNLESS J2=3.
421.000     13 J2=J2+1
422.000        KD=0
423.000        IF(J2.LT.3) GO TO 23
424.000        J2=0
425.000        JS=J1
426.000        J1=0
427.000        XXF=XF-X
428.000        IF((XXF-2.0D0*C*H)*H.LT.0.0D0) GO TO 15
429.000        DO 54 J=1,K3
430.000        DO 54 I=1,NP
431.000        TR3(I,J)=TR21(I,J)
432.000        F2I1(I,J)=F2I(I,J)
433.000     54 CONTINUE
434.000        NINC=NINC+1
435.000        C1=C
436.000        C2=C1*C1
437.000        H=C1*H
438.000        GO TO 16
439.000     15 XXF2=XXF/2.0D0
440.000        JHD=1
```

```
441.000          C1=XXF2/H
442.000          H=XXF2
443.000          C2=C1*C1
444.000    C
445.000    C
446.000    C***RESTART SECTION. THIS PROVIDES THE DATA NECESSARY FOR REENTRY TO
447.000    C    THE DEVOGELAERE LOOP WITH INCREASED OR DECREASED H.
448.000       16 H2=H*H
449.000          ALPHA=(2.0D0+C1)*C1*C2/3.0D0
450.000          DO 55 J=1,K3
451.000          DO 55 I=1,NP
452.000          Z(I,J)=C1*Z(I,J)
453.000          F2I(I,J)=C2*F2I(I,J)
454.000          FEVEN(I,J)=C2*FEVEN(I,J)
455.000          FODD(I,J)=C2*FODD(I,J)
456.000          FODD(I,J)=4.0D0*(1.0D0-C1)*FEVEN(I,J)+C1*FODD(I,J)
457.000          TR1(I,J)=ALPHA*TR3(I,J)
458.000       55 CONTINUE
459.000    C    SOME DIFFERENCES OCCUR IF THE PREVIOUS STEPLENGTH WAS USED ONLY ONCE
460.000          IF(JS.EQ.1) GO TO 21
461.000          GO TO 22
462.000    C       THEN CALCULATE APPROPRIATE ERROR TERM WHEN JS=1
463.000       21 ERRFAC=4.8D0*C2*C21/((C21*(-C2+7.0D0*C1+1.2D1)+2.0D0*C11*(-C2+6.0D
464.000         10*C1+1.0D1)+2.0D0*(C1+2.0D0))*H)
465.000          IF(JD.EQ.1) BETA1=BETA
466.000          GO TO 24
467.000       22 ERRFAC=1.6D0*(C2/H)/(1.2D1+7.0D0*C1-C2)
468.000       24 BETA=ALPHA/(C2*C2)
469.000          X=X+H
470.000          K=K+1
471.000          W(K)=X
472.000          GO TO 3
473.000    C
474.000    C ---------------------------------------------------------------
475.000    C
476.000    C    FORMAT STATEMENTS
477.000      112 FORMAT(1HO,'THE END OF THE RANGE OF INTEGRATION HAS BEEN REACHED'/
478.000         1/'NUMBER OF INTEGRATION STEPS CARRIED OUT',16/'NUMBER OF INCREASES
479.000         2 IN STEPLENGTH REQUIRED DURING THE COURSE OF INTEGRATION',I3/'NUMB
480.000         3ER OF DECREASES IN STEPLENGTH REQUIRED DURING THE COURSE OF INTEGR
```

```
      SUBROUTINE DEVUG(H,K2,K3,XF)
```



```
     4ATION',I3,///)
140 FORMAT(1H0,1HX,14X,1HH,14X,1HY,14X,5HTRERR/1H0)
141 FORMAT(1H ,2(D13.6,2X),3(D13.6,2X,D10.3,2X)/4(D13.6,2X,D10.3,2X)/4
   1(D13.6,2X,D10.3,2X))
142 FORMAT(8HORESTART)
143 FORMAT(1H ,2(D13.6,2X),3(D13.6,14X)/4(D13.6,14X)/4(D13.6,14X))
      END
```

```
488.000         SUBROUTINE DRV2(K2,K3)
489.000   C
490.000   C
491.000   C   THIS ROUTINE CALCULATES THE POTENTIAL FOR ELECTRON-HYDROGEN
492.000   C   SCATTERING IN THE STRONG-COUPLING APPROXIMATION, WHEN ONLY 1S AND 2S
493.000   C   ATOMIC STATES ARE INCLUDED IN THE EIGENFUNCTION EXPANSION, WITH
494.000   C   EXCHANGE NEGLECTED. THE SECOND DERIVATIVE OF THE SOLUTION AT THIS
495.000   C   MESH POINT MAY NOW BE CALCULATED AND IS STORED IN THE F MATRIX.
496.000   C   BLCK 3 :CN MATRIX STORES THE COEFFICIENTS OF X**(-K-1), K=1,---,MAXP
497.000   C            IN THE ASYMPTOTIC FORM OF THE POTENTIAL WHERE MAXP IS
498.000   C            THE MAXIMUM VALUE OF K.
499.000   C            ZZ IS THE NET CHARGE.
500.000   C
501.000   C
502.000         IMPLICIT REAL*8(A-H,O-Z)
503.000         COMMON/BLCK 1/H0,X0,R0,RA
504.000         COMMON/BLCK 2/WP(5),NP,NA,NBND,LP(5)
505.000         COMMON/BLCK 3/CN(5,5,4),ZZ,MAXP
506.000         COMMON/BLCK 6/X,Y0(5,12),Z0(5,12),F0(5,12),Y(5,12),Z(5,12),F(5,12)
507.000         COMMON/BLCK 7/W(2000),FUN(5,12,2000)
508.000         COMMON/BLCK 9/KI,NTOT2,FB2(5,12),FDR2(5,12),WI(2000),FUNI(5,12,200
509.000        10)
510.000   C
511.000         IF(K3.EQ.NTOT2.AND.X.GE.RA) GO TO 3
512.000         DO 2 J=1,K3
513.000         DO 2 I=1,NP
514.000         F(I,J)=Y(I,J)*(DFLOAT(LP(I)*LP(I)+LP(I))/(X*X)+2.0D0*(-ZZ/X-WP(I))
515.000        1)
516.000         DO 1 K=1,NP
517.000         IF(I.NE.K) V=4.0D0*DSQRT(2.0D0)*(2.0D0+3.0D0*X)*DEXP(-1.5D0*X)/2.7
518.000        1D1
519.000         IF(I.EQ.1.AND.K.EQ.1) V=-2.0D0*(1.0D0+1.0D0/X)*DEXP(-2.0D0*X)
520.000         IF(I.EQ.2.AND.K.EQ.2) V=-2.0D0*(1.0D0/X+0.75D0+X/4.0D0+X*X/8.0D0)*
521.000        1DEXP(-X)
522.000         F(I,J)=F(I,J)+V*Y(K,J)
523.000       1 CONTINUE
524.000       2 CONTINUE
525.000         RETURN
526.000       3 DO 5 J=1,K3
527.000         DO 5 I=1,NP
```

```
      F(I,J)=2.0D0*(-ZZ/X-WP(I))*Y(I,J)
      DO 4 K=1,NP
      DO 4 L=1,MAXP
      F(I,J)=F(I,J)-Y(K,J)*CN(I,K,L)/(X**(L+1))
    4 CONTINUE
      IF(K2.GT.0) GO TO 6
    5 CONTINUE
      RETURN
    6 WRITE(IPRINT,7)
      RETURN
    7 FORMAT(53HOEXPRESSIONS FOR THE INHOMOGENEOUS TERMS ARE REQUIRED)
      END
```

MATHEMATICS OF COMPUTATION, VOLUME 32, NUMBER 143
JULY 1978, PAGES 751–762

On de Vogelaere's Method for $y'' = f(x, y)$

By John P. Coleman and Julie Mohamed

Abstract. Easily calculated truncation-error estimates are given which permit efficient automatic error control in computations based on de Vogelaere's method. An upper bound for the local truncation error is established, the interval of absolute stability is found to be $[-2, 0]$, and it is shown that the global truncation error is of order h^4 where h is the steplength.

1. Introduction. Ordinary differential equations of the special form

$$y'' = f(x, y), \tag{1}$$

and systems of such equations, arise in a variety of physical contexts. Examples include atomic and nuclear scattering problems, molecular-dynamics calculations for liquids and gases, and stellar mechanics. A numerical method proposed by de Vogelaere [3] has been used extensively to solve equations of this type (e.g. [1], [8], [9] and [12]), and Chandra [2] has published a computer program which uses de Vogelaere's method to solve the differential equations arising in a close-coupling formulation of quantum mechanical scattering problems. Chandra's program makes no attempt to monitor the local truncation error, and leaves the choice of steplength strategy entirely to the user.

A major objective in recent work on numerical methods for nonstiff ordinary differential equations of first order has been the development of efficient codes which automatically select steps as large as possible while satisfying some error criterion specified by the user (see surveys by Shampine et al. [11] and by Lambert [7]). Adopting this philosophy, our aim has been to improve on existing implementations of de Vogelaere's method for the second-order equation (1) by incorporating a method of truncation-error estimation, and an automatic mesh-selection facility.

For ease of reference, and to establish notation, we present in Section 2 a derivation of de Vogelaere's method based on Taylor expansions. The estimation of the local truncation error, on which the choice of steplength depends, is discussed in Section 3. Despite the frequent use of de Vogelaere's method we are unaware of any previous error analysis, or any study of the stability of the method; later sections of the paper deal with these matters. A bound for the local truncation error is derived in Section 4, the stability region of the method is established in Section 5, and the global truncation error is examined in Section 6.

2. Derivation of de Vogelaere's Method. The differential equation (1) is to be solved, in some real interval $[a, b]$, subject to the initial conditions

Received August 4, 1977.
AMS (MOS) subject classifications (1970). Primary 65L05; Secondary 81.65.

$$y(x_0) = y_0, \qquad z(x_0) = z_0,$$

where y_0 and z_0 are specified numbers and

$$z(x) = \frac{dy}{dx}.$$

The mesh points, which in general are not evenly spaced, are denoted by x_n ($n = 0, 1, \ldots$), y_n is an approximation to the exact solution $y(x_n)$ at the mesh point x_n, and we shall also use the abbreviation

$$f_n = f(x_n, y_n).$$

Let h be the initial steplength, so that

$$x_1 = x_0 + h, \qquad x_2 = x_0 + 2h.$$

Then, by using the equations

$$hf_0' = f_0 - f_{-1} + \frac{h^2}{2} f_0'' + O(h^3)$$

and

$$hf_0' + \frac{h^2}{2} f_0'' = f_1 - f_0 - \frac{h^3}{6} f_0''' + O(h^4)$$

in Taylor expansions about x_0, we obtain

$$y(x_1) = y_0 + hz_0 + \frac{h^2}{6}(4f_0 - f_{-1}) + \frac{h^4}{8} f_0'' + O(h^5) \tag{2}$$

and

$$y(x_2) = y_0 + 2hz_0 + \frac{h^2}{3}(4f_1 + 2f_0) + \frac{2h^5}{45} f_0''' + O(h^6). \tag{3}$$

These expressions are valid provided that any errors in f_{-1} and f_1 are of order h^3 and h^4, respectively.

The de Vogelaere algorithm is obtained by neglecting $O(h^4)$ terms in (2) and $O(h^5)$ terms in (3). For a fixed steplength h, its general step, leading from x_{2n} to $x_{2n+2} = x_{2n} + 2h$, may be described as follows:

Given y_{2n}, z_{2n}, f_{2n} and f_{2n-1},

$$\text{(i)} \quad y_{2n+1} = y_{2n} + hz_{2n} + \frac{h^2}{6}(4f_{2n} - f_{2n-1}), \tag{4}$$

$$\text{(ii)} \quad f_{2n+1} = f(x_{2n+1}, y_{2n+1}), \tag{5}$$

$$\text{(iii)} \quad y_{2n+2} = y_{2n} + 2hz_{2n} + \frac{h^2}{3}(4f_{2n+1} + 2f_{2n}), \tag{6}$$

$$\text{(iv)} \quad f_{2n+2} = f(x_{2n+2}, y_{2n+2}), \tag{7}$$

$$\text{(v)} \quad z_{2n+2} = z_{2n} + \frac{h}{3}(f_{2n} + 4f_{2n+1} + f_{2n+2}). \tag{8}$$

The local truncation errors in y_{2n+2} and z_{2n+2} are of order h^5, and that in y_{2n+1} is of order h^4. This algorithm has some similarity with Runge-Kutta methods, but it involves only two function evaluations per step whereas a Runge-Kutta method of the same order requires three (see e.g. [10]). Unlike Runge-Kutta methods, the de Vogelaere algorithm is not self starting, but, as de Vogelaere [3] suggested, this difficulty is easily overcome since by taking

$$y_{-1} = y_0 - hz_0 + \frac{h^2}{2} f_0 \tag{9}$$

we can calculate f_{-1} with an error of order h^3.

An arbitrary change of steplength can be introduced without additional function evaluations. If a steplength h_1 is used as far as x_{2n}, the quantity f_{2n-1} refers to the mesh point $x_{2n-1} = x_{2n} - h_1$. If we now change the steplength to $h_2 = ch_1$, f_{2n-1} must be replaced in Eq. (4) by $\bar{f}_{2n-1}$, an approximation for f at $x_{2n} - h_2$. This can be achieved by defining

$$\bar{f}_{2n-1} = f_{2n} + c(f_{2n-1} - f_{2n}), \tag{10}$$

which has a local truncation error of order h_2^2.

3. **Truncation Error Estimates.** Equation (3) shows that the leading term in the truncation error in the step from x_{2n} to x_{2n+2} is

$$\frac{2h^5}{45} f'''_{2n}. \tag{11}$$

De Vogelaere [3] described a method for estimating this quantity when the steplength h is constant. To allow us to monitor the truncation error immediately after changes of steplength, it is necessary to introduce some modifications which are described in this section. We consider four separate cases.

3.1. *Fixed Steplength.* Since the truncation error in y_{2n+1} is of order h^4, and that in y_{2n+2} is of order h^5, $y(x_{2n+1})$ may be estimated more accurately by Taylor expansion about x_{2n}. By using the equations

$$h^2 f''_{2n+2} = f_{2n+2} - 2f_{2n+1} + f_{2n} + O(h^3)$$

and

$$hf'_{2n+2} = \frac{3}{2} f_{2n+2} - 2f_{2n+1} + \frac{1}{2} f_{2n} + O(h^3)$$

to replace the low-order derivatives, we obtain the new estimate

$$y^*_{2n+1} = y_{2n+2} - hz_{2n+2} + \frac{h^2}{24}(7f_{2n+2} + 6f_{2n+1} - f_{2n}), \tag{12}$$

which has a local truncation error of order h^5. Consequently,

$$y^*_{2n+1} - y_{2n+1} = \frac{h^4}{8} f''_{2n} + O(h^5). \tag{13}$$

Similarly, if the same steplength is used for the next step,

$$y^*_{2n+3} - y_{2n+3} = \frac{h^4}{8} f''_{2n+2} + O(h^5),$$

and the required truncation-error estimate is given by

$$\frac{2h^5}{45} f'''_{2n} \simeq \frac{h^4}{45}(f''_{2n+2} - f''_{2n})$$

$$\simeq \frac{8}{45}[(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})].$$

The truncation error per unit step, which is a more appropriate basis for decisions about the steplength, is approximated by

$$\frac{4}{45h}[(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})]. \tag{14}$$

3.2. *Immediately Following a Step Change.* Let h_1 be the steplength used in the step from x_{2n-2} to x_{2n}, and in the preceding step. Equations (6) and (8) can be used to put Eq. (12), with n replaced by $n-1$, in the form

$$y^*_{2n-1} = y_{2n-2} + h_1 z_{2n-2} + \frac{1}{24}h_1^2(7f_{2n-2} + 6f_{2n-1} - f_{2n}). \tag{15}$$

Then

$$\begin{aligned} y^*_{2n-1} - y_{2n-1} &= \frac{1}{24}h_1^2(-9f_{2n-2} + 6f_{2n-1} - f_{2n} + 4f_{2n-3}) \\ &= \frac{h_1^4}{8} f''_{2n-1} - \frac{h_1^5}{6} f'''_{2n-1} + O(h_1^6). \end{aligned} \tag{16}$$

If the steplength is now changed to $h_2 = ch_1$ for the next step, Eq. (10) combines with (4) to give

$$y_{2n+1} = y_{2n} + h_2 z_{2n} + \frac{1}{6}h_2^2[(3+c)f_{2n} - cf_{2n-1}]. \tag{17}$$

Equations (6) and (8) apply with $h = h_2$, and

$$y^*_{2n+1} = y_{2n} + h_2 z_{2n} + \frac{1}{24}h_2^2(7f_{2n} + 6f_{2n+1} - f_{2n+2}).$$

Steps similar to those used in deriving Eq. (16) then show that

$$y^*_{2n+1} - y_{2n+1} = \frac{h_2^4}{24}\left(1 + \frac{2}{c}\right)f''_{2n+1} - \frac{h_2^5}{36}\left(2 + \frac{3}{c} + \frac{1}{c^2}\right)f'''_{2n+1} + O(h_2^6), \tag{18}$$

and

$$(y^*_{2n+1} - y_{2n+1}) - \frac{1}{3}c^4\left(1 + \frac{2}{c}\right)(y^*_{2n-1} - y_{2n-1})$$

$$= \frac{h_2^5}{72}\left(-1 + \frac{7}{c} + \frac{12}{c^2}\right)f'''_{2n} + O(h_2^6).$$

The resulting estimate of the truncation error per unit step is

$$\frac{8c[(y^*_{2n+1} - y_{2n+1}) - \alpha(y^*_{2n-1} - y_{2n-1})]}{5h_1(12 + 7c - c^2)}, \tag{19}$$

where

$$\alpha = \frac{1}{3}c^3(2 + c). \tag{20}$$

3.3. *The Second Step After a Step Change.* If the steplength $h_2 = ch_1$, introduced for the step from x_{2n} to x_{2n+2}, is retained in the next step, the equation

$$y^*_{2n+3} - y_{2n+3} = \frac{h_2^4}{8}f''_{2n+3} - \frac{h_2^5}{6}f'''_{2n+3} + O(h_2^6)$$

follows directly from (16). This combines with Eq. (18) to give

$$\frac{1}{3}\left(1 + \frac{2}{c}\right)(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1}) = \frac{h_2^5}{36}\left(3 + \frac{5}{c} + \frac{1}{c^2}\right)f'''_{2n+2} + \cdots. \tag{21}$$

The local truncation error per unit step is estimated to be

$$\frac{4c[\beta(y^*_{2n+3} - y_{2n+3}) - (y^*_{2n+1} - y_{2n+1})]}{5h_1(1 + 5c + 3c^2)} \tag{22}$$

with

$$\beta = \frac{1}{3}\left(1 + \frac{2}{c}\right).$$

If we now continue to use the steplength h_2 the results of Subsection 3.1 apply to later steps.

3.4. *Two Step Changes in Succession.* The alternative to the situation discussed in Subsection 3.3 is that having completed the step from x_{2n} to x_{2n+2} with steplength h_2 we then adopt a new steplength $h_3 = c_1h_2$. The relevant mesh points are

$$x_{2n-1} = x_{2n} - h_1, \quad x_{2n}, \quad x_{2n+1} = x_{2n} + h_2,$$
$$x_{2n+2} = x_{2n} + 2h_2, \quad x_{2n+3} = x_{2n+2} + h_3, \quad x_{2n+4} = x_{2n+2} + 2h_3.$$

In this case, by analogy with Eq. (18),

$$y^*_{2n+3} - y_{2n+3} = \frac{h_3^4}{24}\left(1 + \frac{2}{c_1}\right)f''_{2n+3} - \frac{h_3^5}{36}\left(2 + \frac{3}{c_1} + \frac{1}{c_1^2}\right)f'''_{2n+3} + O(h_3^6). \tag{23}$$

The appropriate linear combination of (18) and (23) is

$$\beta(y^*_{2n+3} - y_{2n+3}) - \alpha_1(y^*_{2n+1} - y_{2n+1})$$

with

$$\beta = \frac{1}{3}\left(1 + \frac{2}{c}\right), \qquad \alpha_1 = \frac{1}{3}c_1^3(2 + c_1).$$

Our estimate for the local truncation error per unit step is then

$$\frac{24c^2c_1^2[\beta(y^*_{2n+3} - y_{2n+3}) - \alpha_1(y^*_{2n+1} - y_{2n+1})]}{5h_3[c^2(12 + 7c_1 - c_1^2) + c(20 + 12c_1 - 2c_1^2) + 2c_1 + 4]}. \tag{24}$$

4. A Bound for the Local Truncation Error. The error analysis described here is based on three functionals which are related to the truncation errors in the formulae (4), (6) and (8). For an arbitrary function $y(x)$, having $p + 1$ continuous derivatives, we define the functional

$$L_1[y(x), h] = y(x + h) - y(x) - hy'(x) - \frac{h^2}{6}[4y''(x) - y''(x - h)]. \tag{25}$$

By using Taylor's theorem in the form

$$y(x + jh) = y(x) + jhy'(x) + \cdots + \frac{(jh)^p}{p!}y^{(p)}(x) + \frac{h^{p+1}}{p!}\int_0^j (j - s)^p y^{(p+1)}(x)\,ds,$$

it can be shown that

$$L_1[y(x), h] = \frac{h^4}{6}\int_{-1}^{1} G_1(s)y^{\mathrm{iv}}(x + sh)\,ds \tag{26}$$

with

$$G_1(s) = \begin{cases} (1 + s), & -1 \leqslant s \leqslant 0, \\ (1 - s)^3, & 0 \leqslant s \leqslant 1. \end{cases}$$

Since $G_1(s)$ is of constant sign on the interval of integration, Eq. (26) may be written as

$$L_1[y(x), h] = h^4c_1y^{\mathrm{iv}}(x + \theta_1 h), \qquad -1 < \theta_1 < 1, \tag{27}$$

with

$$c_1 = \frac{1}{6}\int_{-1}^{1} G_1(s)\,ds = \frac{1}{8}.$$

The same approach applied to the functional

$$L_2[y(x), h] = y(x + 2h) - y(x) - 2hy'(x) - \frac{h^2}{3}[4y''(x + h) + 2y''(x)] \tag{28}$$

gives

$$L_2[y(x), h] = \frac{h^5}{4!}\int_0^2 G_2(s)y^{\mathrm{v}}(x + sh)\,ds,$$

where

$$G_2(s) = \begin{cases} (2-s)^4 - 16(1-s)^2, & 0 \leqslant s \leqslant 1, \\ (2-s)^4, & 1 \leqslant s \leqslant 2. \end{cases}$$

The kernel function $G_2(s)$ is of constant sign, and consequently

$$L_2[y(x), h] = h^5 c_2 y^{v}(x + \theta_2 h), \quad 0 < \theta_2 < 2, \tag{29}$$

with

$$c_2 = \frac{1}{24}\int_0^2 G_2(s)\, ds = \frac{2}{45}.$$

The third functional required is

$$L_3[y(x), h] = y'(x + 2h) - y'(x) - \frac{h}{3}[y''(x) + 4y''(x + h) + y''(x + 2h)], \tag{30}$$

and the standard expression for the truncation error in Simpson's rule gives

$$L_3[y(x), h] = -\frac{h^5}{90} y^{vi}(x + \theta_3 h), \quad 0 < \theta_3 < 2. \tag{31}$$

Let $y(x)$ be the exact solution of our initial-value problem. To investigate the local truncation error in the step from x_{2n} to x_{2n+2} we suppose that the starting values at x_{2n} are exact, i.e.

$$y_{2n} = y(x_{2n}), \qquad z_{2n} = y'(x_{2n}),$$
$$f_{2n} = y''(x_{2n}), \qquad f_{2n-1} = y''(x_{2n-1}).$$

Then the truncation error at x_{2n+1} is

$$\begin{aligned} y(x_{2n+1}) - y_{2n+1} &= L_1[y(x_{2n}), h] \\ &= \frac{h^4}{8} y^{iv}(x_{2n} + \theta_1 h), \quad -1 < \theta_1 < 1. \end{aligned} \tag{32}$$

Also, in view of the assumed starting values,

$$L_2[y(x_{2n}), h] = y(x_{2n+2}) - y_{2n} - 2hz_{2n} - \frac{h^2}{3}[4y''(x_{2n+1}) + 2f_{2n}],$$

and the truncation error at x_{2n+2} is

$$y(x_{2n+2}) - y_{2n+2} = L_2[y(x_{2n}), h] + \frac{4h^2}{3}[y''(x_{2n+1}) - f_{2n+1}]. \tag{33}$$

To obtain a bound on this error we assume a Lipschitz condition

$$|f(x, y) - f(x, \eta)| \leqslant K|y - \eta| \tag{34}$$

for all x in the appropriate interval $[a, b]$ and all finite y and η. Then, if

$$|y^{iv}(x)| \leqslant M_4, \qquad |y^{v}(x)| \leqslant M_5, \qquad x \in [a, b],$$

Eq. (33) gives the bound

$$|y(x_{2n+2}) - y_{2n+2}| \leqslant \frac{2h^5}{45} M_5 + \frac{Kh^6}{6} M_4. \tag{35}$$

In a similar manner it can be shown that

$$\begin{aligned} y'(x_{2n+2}) - z_{2n+2} &= L_3[y(x_{2n}), h] + \frac{4h}{3}[y''(x_{2n+1}) - f_{2n+1}] \\ &\quad + \frac{h}{3}[y''(x_{2n+2}) - f_{2n+2}], \end{aligned} \tag{36}$$

giving the bound

$$|y'(x_{2n+2}) - z_{2n+2}| \leqslant \frac{h^5}{90} M_6 + \frac{Kh^5}{18}(3 + Kh^2) M_4 + \frac{2Kh^6}{135} M_5, \tag{37}$$

where

$$|y^{vi}(x)| \leqslant M_6, \qquad x \in [a, b].$$

5. **Stability Analysis.** If $y(x)$ is the exact solution of the initial-value problem, the global truncation errors in the function and derivative values at the end of the nth de Vogelaere step are

$$\begin{aligned} y(x_{2n-1}) - y_{2n-1} &= e_n^{(1)}, \\ y(x_{2n}) - y_{2n} &= e_n^{(2)}, \\ y'(x_{2n}) - z_{2n} &= e_n^{(3)}/h. \end{aligned}$$

The factor of h in the third definition is introduced to simplify the form of later equations. Equation (4), combined with the definition of the functional L_1, gives

$$\begin{aligned} e_{n+1}^{(1)} &= e_n^{(2)} + e_n^{(3)} + \frac{2h^2}{3}[y''(x_{2n}) - f_{2n}] \\ &\quad - \frac{h^2}{6}[y''(x_{2n-1}) - f_{2n-1}] + L_1[y(x_{2n}), h]. \end{aligned} \tag{38}$$

Similarly, from Eqs. (6) and (8) and the definitions of the corresponding functionals,

$$\begin{aligned} e_{n+1}^{(2)} &= e_n^{(2)} + 2e_n^{(3)} + \frac{4h^2}{3}[y''(x_{2n+1}) - f_{2n+1}] \\ &\quad + \frac{2h^2}{3}[y''(x_{2n}) - f_{2n}] + L_2[y(x_{2n}), h] \end{aligned} \tag{39}$$

and

$$\begin{aligned} e_{n+1}^{(3)} &= e_n^{(3)} + \frac{h^2}{3}[y''(x_{2n}) - f_{2n} + 4\{y''(x_{2n+1}) - f_{2n+1}\} + y''(x_{2n+2}) - f_{2n+2}] \\ &\quad + hL_3[y(x_{2n}), h]. \end{aligned} \tag{40}$$

In accordance with normal practice (e.g. Lambert [6, p. 257]) the stability of the method is discussed with reference to the equation

$$y'' = \lambda^2 y.$$

In this case our equations for the cumulative errors simplify to

$$A\mathbf{e}_{n+1} = B\mathbf{e}_n + \boldsymbol{\phi}_{n+1} \tag{41}$$

where $\mathbf{e}_n$ is a column vector with components $e_n^{(1)}$, $e_n^{(2)}$ and $e_n^{(3)}$, $\boldsymbol{\phi}_{n+1}$ has as its components the three functionals occurring in Eqs. (38)–(40), and the matrices A and B are

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -\frac{4\bar{h}}{3} & 1 & 0 \\ -\frac{4\bar{h}}{3} & -\frac{\bar{h}}{3} & 1 \end{pmatrix}, \quad B = \begin{pmatrix} -\frac{\bar{h}}{6} & 1 + \frac{2\bar{h}}{3} & 1 \\ 0 & 1 + \frac{2\bar{h}}{3} & 2 \\ 0 & \frac{\bar{h}}{3} & 1 \end{pmatrix}$$

with $\bar{h} = \lambda^2 h^2$. Since A is nonsingular Eq. (41) may be written as

$$\mathbf{e}_{n+1} = C\mathbf{e}_n + A^{-1}\boldsymbol{\phi}_{n+1},$$

where

$$C = \begin{pmatrix} -\frac{\bar{h}}{6} & 1 + 2\frac{\bar{h}}{3} & 1 \\ -\frac{2\bar{h}^2}{9} & 1 + 2\bar{h} + \frac{8\bar{h}^2}{9} & 2\left(1 + \frac{2\bar{h}}{3}\right) \\ -\frac{2\bar{h}^2}{9}\left(1 + \frac{\bar{h}}{3}\right) & \frac{\bar{h}}{3}\left(6 + \frac{14\bar{h}}{3} + \frac{8\bar{h}^2}{9}\right) & 1 + 2\bar{h} + \frac{4\bar{h}^2}{9} \end{pmatrix}.$$

The characteristic polynomial of the matrix C is

$$p(r, \bar{h}) = 6r^3 - (12 + 23\bar{h} + 8\bar{h}^2)r^2 + (6 - 2\bar{h} - 4\bar{h}^2)r + \bar{h}.$$

The "Schur criterion" described on p. 78 of Lambert's book [6] can be used to show that $p(r, \bar{h})$ is a Schur polynomial, in other words that all its zeros lie inside the unit circle, if and only if $\bar{h} \in (-2, 0)$. Thus, the interval of absolute stability of de Vogelaere's method is $[-2, 0]$. The moduli of the zeros of $p(r, \bar{h})$ are plotted in Figure 1 for a range of values of $\bar{h}$. The three zeros, though distinct, have the same modulus when $\bar{h} = -1.732$ (to 3 decimal places).

6. The Cumulative Error. Bounds for the global truncation error can be obtained from Eqs. (38)–(40) and bounds established in Section 4. However, the dependence on a (fixed) steplength h is more readily obtained in an alternative approach described by Kopal [5, p. 219]. Let y be the exact solution of the initial-value problem

$$y'' = f(x, y), \quad y(x_0) = y_0, \quad z(x_0) = z_0$$

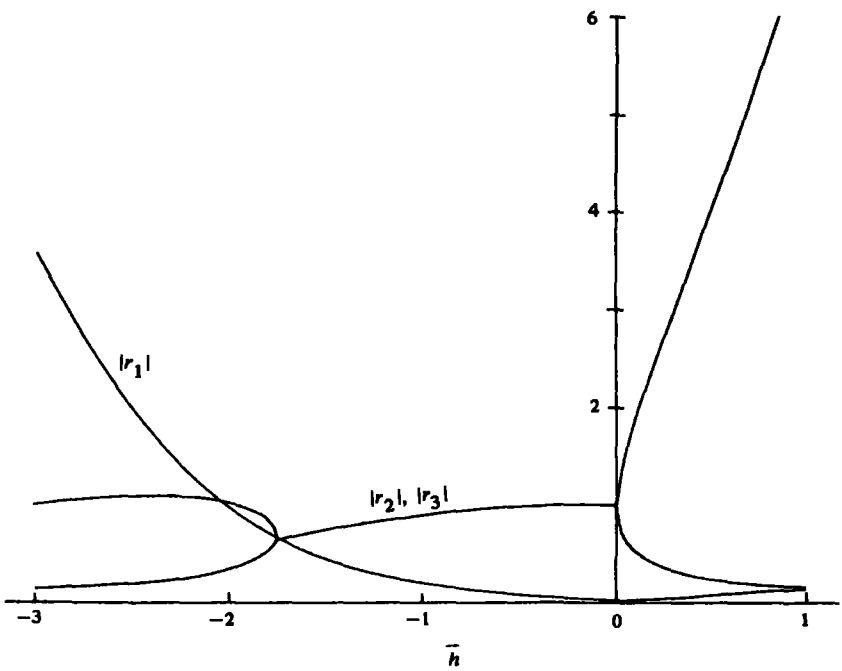

FIGURE 1
The moduli of the zeros r_1, r_2 and r_3 of $p(r, \bar{h})$. r_2 and r_3 are complex for $\bar{h} \in (-1.75, 0)$, and r_1 is always real

and z its derivative. Another solution of the differential equation is denoted by ξ and its derivative is η. Then, if the squares and higher powers of the differences $|\xi(x) - y(x)|$ and $|\eta(x) - z(x)|$ are neglected,

$$\frac{d}{dx}(\eta - z) = \frac{\partial f}{\partial y}(\xi - y), \qquad \frac{d}{dx}(\xi - y) = \eta - z. \tag{42}$$

When this is combined with the adjoint system

$$\lambda' = -\left(\frac{\partial f}{\partial y}\right)\mu, \qquad \mu' = -\lambda,$$

solved subject to the boundary conditions

$$\lambda(x_{2n}) = 1, \qquad \mu(x_{2n}) = 0,$$

Kopal's approach [5] gives the truncation-error estimate

$$y(x_{2n}) - y_{2n} \simeq \sum_{j=1}^{n} [\lambda(x_{2j})R_j + \mu(x_{2j})S_j]. \tag{43}$$

Here R_j and S_j represent the errors in evaluating the solution and its derivative in the jth step of the de Vogelaere method; more precisely

$$R_j = \xi_j(x_{2j}) - y_{2j}, \qquad S_j = \eta_j(x_{2j}) - z_{2j},$$

where $\xi_j(x)$ and $\eta_j(x)$ are the solutions of (42) on the interval $[x_{2j-2}, x_{2j}]$ satisfying

the initial conditions

$$\xi_j(x_{2j-2}) = y_{2j-2}, \qquad \eta_j(x_{2j-2}) = z_{2j-2}.$$

Regarding the right-hand side of (43) as a quadrature sum we may write

$$y(x_{2n}) - y_{2n} \simeq \frac{1}{2h} \int_{x_0}^{x_{2n}} [\lambda(x)R(x) + \mu(x)S(x)]\, dx, \tag{44}$$

and only the lowest power of h in R and S is required for our purpose. From Eq. (3), or by retaining only the lowest power of h in (33), we obtain

$$R(x) = \frac{2h^5}{45} y^{v}(x).$$

The h^5 contribution to the local truncation error in the derivative comes from two sources, the first and second terms on the right-hand side of Eq. (36); thus,

$$S(x) = -\frac{h^5}{90} y^{vi}(x) + \frac{h^5}{6} \frac{\partial f}{\partial y} y^{iv}(x).$$

It follows from (44) that the global error is of order h^4.

As an example we consider the initial-value problem

$$y'' = -y, \qquad y(0) = 0, \qquad y'(0) = 1.$$

In this case

$$y(x) = \sin x, \qquad \lambda(x) = \cos(x_{2n} - x), \qquad \mu(x) = \sin(x_{2n} - x),$$

$$R(x) = \frac{2h^5}{45} \cos x, \qquad S(x) = -\frac{7h^5}{45} \sin x,$$

and the global error estimate is

$$\begin{aligned} y(x_{2n}) - y_{2n} &\simeq \frac{h^4}{90} \int_0^{x_{2n}} [2\cos(x_{2n} - x)\cos x - 7\sin(x_{2n} - x)\sin x]\, dx \\ &= \frac{h^4}{180} [9x_{2n} \cos x_{2n} - 5\sin x_{2n}]. \end{aligned}$$

In particular, when $x_{2n} = \pi/2$ this estimate becomes $-h^4/36$ which agrees closely with numerical results.

7. **Conclusion.** The truncation-error estimates presented in Section 3 provide a practical means of efficient error control in applications of de Vogelaere's method. The error estimates are inexpensive, requiring no extra function evaluations, and only two function evaluations are lost if a particular step has to be discarded. We have used this approach in a program for quantum mechanical scattering calculations, allowing the computer to choose the steplength at each step so that the truncation error per unit step is less, but not too much less, than a specified tolerance.

An important conclusion from our error analysis is that the global error in de Vogelaere's method is of order h^4. This contrasts with linear multistep methods for

Eq. (1) which have a local truncation error of order h^{p+2} but a global error of order h^p (see Henrici [4, p. 314]). For example, Numerov's method (Lambert [6, p. 255], Kopal [5, p. 183]) has a local truncation error of order h^6 but a global error of order h^4 like de Vogelaere's method.

Acknowledgement. This research was supported in part by a Research Grant from the Science Research Council.

Added in Proof. We have now established a rigorous upper bound on the global truncation error, assuming the Lipschitz condition and derivative bounds introduced in Section 4. For any $h_0 > 0$, constants a and M exist such that, for all $h \leqslant h_0$,

$$|y(x_{2n}) - y_{2n}| \leqslant \epsilon \exp[a(x_{2n} - x_0)] + \frac{\exp[a(x_{2n} - x_0)] - 1}{2a} h^4 M$$

where

$$\epsilon = \max\{|y(x_0) - y_0|, h|y(x_{-1}) - y_{-1}|, |z(x_0) - z_0|\}.$$

In particular, if the initial conditions of Section 2 are satisfied exactly, then

$$\epsilon = h|y(x_{-1}) - y_{-1}|,$$

which is of order h^4 if Eq. (9) is used to compute y_{-1}. Details will appear in Julie Mohamed's Ph.D. thesis.

Department of Mathematics
University of Durham
Durham, England

1. S. BASAVAIAH & R. F. BROOM, "Characteristics of in-line Josephson tunneling gates," *IEEE Trans. Magnetics*, v. MAG-11, 1975, pp. 759–762.

2. N. CHANDRA, "A general program to study the scattering of particles by solving coupled inhomogeneous second-order differential equations," *Computer Phys. Commun.*, v. 5, 1973, pp. 417–429.

3. R. de VOGELAERE, "A method for the numerical integration of differential equations of second order without explicit first derivatives," *J. Res. Nat. Bur. Standards*, v. 54, 1955, pp. 119–125.

4. P. HENRICI, *Discrete Variable Methods in Ordinary Differential Equations*, Wiley, New York, 1962.

5. Z. KOPAL, *Numerical Analysis*, Chapman and Hall, London, 1955.

6. J. D. LAMBERT, *Computational Methods in Ordinary Differential Equations*, Wiley, London, 1973.

7. J. D. LAMBERT, "The initial value problem for ordinary differential equations," in *The State of the Art in Numerical Analysis* (D. A. H. Jacobs, Editor), Academic Press, London, 1977, pp. 451–500.

8. J. -M. LAUNAY, "Body-fixed formulation of rotational excitation: exact and centrifugal decoupling results for CO – He," *J. Phys. B: Atom. Molec. Phys.*, v. 9, 1976, pp. 1823–1838.

9. W. A. LESTER, JR., "De Vogelaere's method for the numerical integration of second-order differential equations without explicit first derivatives: application to coupled equations arising from the Schrödinger equation," *J. Computational Phys.*, v. 3, 1968, pp. 322–326.

10. R. E. SCRATON, "The numerical solution of second-order differential equations not containing the first derivative explicitly," *Comput. J.*, v. 6, 1964, pp. 368–370.

11. L. F. SHAMPINE, H. A. WATTS & S. M. DAVENPORT, "Solving nonstiff ordinary differential equations—the state of the art," *SIAM Rev.*, v. 18, 1976, pp. 376–411.

12. L. VERLET, "Computer 'experiments' on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules," *Phys. Rev.*, v. 159, 1967, pp. 98–103.