SOME NUMERICAL PROCEDURES FOR SOLVING SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS CONTAINING A SMALL PARAMETER

by

Walter Frederick Finden

Research Report CS-75-22

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

August 1975

ABSTRACT

Numerical procedures for solving systems of ordinary differential equations in which a small parameter multiplies some of the derivatives are discussed. Three existing procedures based on asymptotic expansions are discussed and then a method based on interpolation is presented. An error analysis is given for the interpolation scheme and then a discussion on numerical considerations follows. A description of the actual implementation of the interpolation scheme is then given followed by the results of applying this implementation to several examples. Finally another method based on an asymptotic approximation is presented along with an error analysis for this method and some implications when applied to linear systems with constant coefficients. Some numerical results illustrating the asymptotic nature of the method are also given.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

CHAPTER I

INTRODUCTION

1. The Basic Problem

Our basic problem will be the numerical solution of the system of ordinary differential equations

$$\begin{aligned} \frac{dx}{dt} &= f(x,y,t) \\ \mu\frac{dy}{dt} &= g(x,y,t) \end{aligned} \tag{1.1.1}$$

with initial conditions

$$\begin{aligned} x\Big|_{t=0} &= \alpha \\ y\Big|_{t=0} &= \beta. \end{aligned}$$

Here x, f and α are m-dimensional vectors, y, g and β are n-dimensional vectors and μ is a small positive parameter. Systems of the form (1.1.1) may be found in several areas such as chemical kinetics, missile guidance and nuclear reactor kinetics. Although (1.1.1) is written as a singular perturbation problem, oftentimes systems of ordinary differential equations that contain large numbers on the right side may be put in the form (1.1.1) by dividing some of the equations by the large numbers. Hence for purposes of approximations we may consider them to be written in the form (1.1.1) and apply some of the singular perturbation theory.

2. The Problem of Stiffness

Since (1.1.1) is an example of a stiff system of ordinary differential equations we first describe the problem of stiffness with the initial value problem

$$\frac{dz}{dt} = p(z,t), \quad z(0) = \gamma \tag{1.2.1}$$

where z, p and γ are k-dimensional vectors. Generally explicit methods for solving (1.2.1) numerically require that the step size h satisfy the relation

$$hL < c$$

where L is the Lipschitz constant, and c is some positive constant dependent on the method. But if the Jacobian $\frac{\partial p}{\partial z}$ has eigenvalues with large negative real parts the quantity L is very large requiring h to be very small. However, the initial transients associated with these eigenvalues with large negative real parts quickly die out at which point we would like the step size based on the time scale associated with the other eigenvalues and not the time scale of the large negative eigenvalues.

The system (1.1.1) has the property of stiffness since the Jacobian

$$\begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \\ \mu & \mu \end{pmatrix}$$

generally has some eigenvalues with large negative real parts if μ is small as long as $\frac{\partial g}{\partial y}$ has eigenvalues with negative real parts. The example

$$\begin{aligned} \frac{dx}{dt} &= y \\ \mu\frac{dy}{dt} &= x-y \end{aligned} \tag{1.2.2}$$

illustrates this point. The eigenvalues of the Jacobian of the system (1.2.2) are given by

$$\lambda_1 = \frac{-1-(1+4\mu)^{\frac{1}{2}}}{2\mu}$$

$$\lambda_2 = \frac{-1+(1+4\mu)^{\frac{1}{2}}}{2\mu} .$$

From the limits

$$\lim_{\mu\to 0} \lambda_1 = -\infty$$

$$\lim_{\mu\to 0} \lambda_2 = 1$$

we see that for very small μ we have extreme stiffness since the time scale associated with λ_1 is very much smaller than the time scale associated with λ_2.

3. Outline of Thesis

Many authors, for example [1], [2], [6], [9], [12], have dealt with the problem (1.2.1). However, we are interested here with the problem as written in (1.1.1). In Chapter II we describe methods by Vasileva [15] and Wasow [16], MacMillan [10] and Miranker [11]. In Chapter III we present

a new method based on an interpolation scheme. A discussion of this method may also be found in Finden [3]. We also give an error analysis along with a discussion on some of the implications made evident by these results. In Chapter IV we present some of the actual procedures used to implement the interpolation scheme and some of the numerical considerations involved with these procedures. We also give a brief discussion of the computer subroutine here and give a listing of the subroutine in Appendix C. In Chapter V we give some numerical results of applying the interpolation scheme, showing some of the benefits discussed in previous chapters. In Chapter VI we present another method for solving (1.1.1) based on an asymptotic approximation. Here we also discuss the accuracy of the solutions and give a discussion concerning the application of the method to linear systems with constant coefficients. We then give some numerical results that illustrate the asymptotic nature of the errors in the solutions. In Chapter VII we present some conclusions.

4. Notation

A note on some notational conventions used is in order here. If F is an m-dimensional function of an n-dimensional variable x, then F_x is used to denote the m×n-dimensional matrix $\frac{\partial F}{\partial x}$. If we consider F as a first order tensor and F_x as a second order tensor then we can consider $F_{x(r)} = \frac{\partial^r F}{\partial x^r}$ as an (r+1)st order tensor with dimensions m×n×n×n×... to (r+1) factors. Also just as $F_x x$ denotes a matrix-vector product whose result is an m-dimensional vector, we use $F_{x(r)}x^r$ to denote a tensor product whose result is an m-dimensional vector. The order of calculations is given by

$$F_x(r)x^r = (\ldots((F_x(r)\cdot x)\cdot x)\ldots)\cdot x \tag{1.4.1}$$

where the process is done r times. In tensor notation, letting F^i_j denote the components of F_x, $F_x x = y$ means $F^i_j x^j = y^i$. Similarly (1.4.1) means

$$F^i_{j_1,j_2,\ldots,j_r} x^{j_1} x^{j_2} \ldots x^{j_r} = G^i$$

where G is an m-dimensional vector.

5. Asymptotic Expansions

Since we will use asymptotic expansions in several places we give a definition for it here. Let the function $s(\mu)$ be defined on an interval $0 \le \mu \le \bar{\mu}$. Then the series

$$\sum_{i=0}^{\infty} a_i \mu^i \tag{1.5.1}$$

is said to represent $s(\mu)$ asymptotically, as $\mu \to 0^+$, if

$$\lim_{\mu \to 0^+} \frac{s(\mu) - \sum_{i=0}^{m} a_i \mu^i}{\mu^m} = 0$$

for all $m \ge 0$. We write

$$s(\mu) \sim \sum_{i=0}^{\infty} a_i \mu^i \text{ as } \mu \to 0^+ \tag{1.5.2}$$

to mean that $s(\mu)$ can be represented asymptotically by (1.5.1) as $\mu \to 0^+$. Of course the relation (1.5.2) does not imply that the series (1.5.1) converges. The advantage of (1.5.2) is that the first few terms of (1.5.1) are usually a good approximation to $s(\mu)$ if μ is small enough.

CHAPTER II

A SURVEY OF METHODS

Three methods are examined for the solution of the initial value problem

$$\begin{aligned} \frac{dx}{dt} &= f(x,y,t) \\ \mu\frac{dy}{dt} &= g(x,y,t) \\ x\Big|_{t=0} &= \alpha \\ y\Big|_{t=0} &= \beta. \end{aligned} \tag{2.0.1}$$

1. Asymptotic Expansions of Vasileva

Vasileva [15] makes the following definitions and these shall be used throughout.

If we formally set $\mu = 0$ in (2.0.1) we obtain the degenerate system

$$\begin{aligned} \frac{dx_0}{dt} &= f(x_0,y_0,t) \\ 0 &= g(x_0,y_0,t) \end{aligned} \tag{2.1.1}$$

or

$$\begin{aligned} \frac{dx_0}{dt} &= f(x_0,y_0,t) \\ y_0 &= \phi(x_0,t) \end{aligned} \tag{2.1.2}$$

with the initial condition

$$x_o|_{t=0} = \alpha$$

where $y = \phi(x,t)$, defined on a bounded closed set D, is a root of the equation $g(x,y,t) = 0$.

The adjoined system is defined by

$$\frac{dy}{d\tau} = g(x^*,y,t^*). \qquad (2.1.3)$$

The root $y = \phi(x,t)$ is positively stable in D if for all points $(x^*,t^*) \in D$, where $t^* \in (0,T)$ the points $y = \phi(x^*,t^*)$ are asymptotically stable stationary points, in the sense of Lyapunov, of the adjoined system (2.1.3), as $\tau \to \infty$. The root is negatively stable in D if the same conditions hold as $\tau \to -\infty$. In what follows we will only discuss the positive case. Obvious adjustments can be made for the negative case.

We define the domain of influence of a stable root $y = \phi(x,t)$ as the set of points (x^*,y^*,t^*) such that the solution of the adjoined system (2.1.3) with the initial conditions $y|_{\tau=0} = y^*$ tends to $\phi(x^*,t^*)$ as $\tau \to \infty$.

We call the root $y = \phi(x,t)$ isolated on the set D if there exists an $\varepsilon > 0$ such that the equation $g(x,y,t) = 0$ has no solution other than $\phi(x,t)$ for $|y-\phi(x,t)| < \varepsilon$.

We now assume throughout that:

A) The root $y = \phi(x,t)$ of the equation $g(x,y,t) = 0$ is an isolated, positively stable root in some bounded closed domain D.

B) The initial point $(\alpha,\beta,0)$ belongs to the domain of influence of this root;

C) The solution $x_0(t)$ of the degenerate system (2.1.2) belongs to D for $0 \le t \le T$;

D) The systems (2.0.1) and (2.1.2) have unique solutions in the interval $0 \le t \le T$;

E) The functions f and g have continuous partial derivatives of all orders.

Using these definitions, Vasileva [15] draws a comparison between the system (2.0.1) and the system

$$\frac{dp}{dt} = h(p,t,\mu),\ p|_{t=0} = \gamma, \tag{2.1.4}$$

where h depends regularly on μ. Whereas

$$\lim_{\mu\to 0} p(t,\mu) = p_0(t)$$

uniformly in t where $p_0(t)$ is the solution of

$$\frac{dp_0}{dt} = h(p_0,t,0) \quad p_0|_{t=0} = \gamma,$$

we have

$$\lim_{\mu\to 0} x(t,\mu) = x_0(t) \text{ uniformly in } 0 \le t \le T$$

$$\lim_{\mu\to 0} y(t,\mu) = y_0(t) \text{ uniformly in } 0 < t_1 \le t \le T$$

where $x_0(t)$ and $y_0(t)$ are solutions to the system (2.1.1). Notice the loss of initial conditions on y in (2.1.1) and also that $y(t,\mu)$ is discontinuous at $t = 0$, $\mu = 0$ and that we have uniform convergence only

in the interval $t_1 \leq t \leq T$. Here t_1 may be chosen arbitrarily close to 0 but must remain fixed as $\mu \to 0$.

Vasileva [15] gives a method for obtaining an asymptotic series in μ for the solutions $x(t,\mu)$ and $y(t,\mu)$ of the system (2.0.1). It should be pointed out that the usual way of obtaining such a series say for the system (2.1.4), would be as follows. We would form the series

$$p(t,\mu) = p_0(t)+\mu p_\mu(t,0)+\ldots+ \frac{\mu^n}{n!}p_{\mu(n)}(t,0)+R_{n+1} \tag{2.1.5}$$

by formally substituting (2.1.5) in (2.1.4) and equating like powers of μ. This gives us the variational equations for $p_{\mu(k)}$. Then using the initial conditions

$$p_{\mu(k)}\Big|_{t=0} = 0 \tag{2.1.6}$$

we can solve for the $p_{\mu(k)}$ and hence form (2.1.5). But this is not possible for the system (2.0.1) since the corresponding $x_{\mu(k)}$ and $y_{\mu(k)}$ are undefined at $t = 0$, $\mu = 0$ since $y(t,\mu)$ is discontinuous there. Hence we may not use the initial conditions similar to (2.1.6). Even if we had the correct initial conditions, our series would only be valid in some range $t_1 \leq t \leq T$ where $t_1 > 0$.

Vasileva forms an expansion of the form

$$\begin{aligned} x(t,\mu) &= (\bar{x})_r + (\tilde{x})_r - (\hat{x})_r + \mu^{r+1}R_{r+1} \\ y(t,\mu) &= (\bar{y})_r + (\tilde{y})_r - (\hat{y})_r + \mu^{r+1}S_{r+1} \end{aligned} \tag{2.1.7}$$

where $(\bar{x})_r$ and $(\bar{y})_r$ are series of the form

$$(\bar{x})_r = \sum_{i=0}^{r} \bar{x}_i \mu^i, \qquad (\bar{y})_r = \sum_{i=0}^{r} \bar{y}_i \mu^i, \tag{2.1.8}$$

$(\tilde{x})_r$ and $(\tilde{y})_r$ are series of the form

$$(\tilde{x})_r = \sum_{i=0}^{r} \tilde{x}_i \mu^i, \qquad (\tilde{y})_r = \sum_{i=0}^{r} \tilde{y}_i \mu^i \tag{2.1.9}$$

and $(\hat{x})_r$ and $(\hat{y})_r$ are series of the form

$$(\hat{x})_r = \sum_{i=0}^{r} \hat{x}_i \mu^i, \qquad (\hat{y})_r = \sum_{i=0}^{r} \hat{y}_i \mu^i. \tag{2.1.10}$$

The last series has the effect of cancelling the second series in the boundary layer where it is bad and cancelling the first series outside the boundary layer where it is bad.

To form these series, we need a slightly more restrictive condition on the stability of the root $y = \phi(x,t)$. We assume that

$$\mathrm{Re}[\lambda_i(\frac{\partial g}{\partial y}(x,\phi(x,t),t)] < 0. \tag{2.1.11}$$

Using assumptions A-E and (2.1.11) it can then be shown that the quantities R_{r+1} and S_{r+1} in (2.1.7) can be bounded for $0 \le t \le T$ and $0 \le \mu \le \bar{\mu}$. Of course, this implies that we may write

$$\begin{aligned} x(t,\mu) &\sim (\bar{x})_\infty + (\tilde{x})_\infty - (\hat{x})_\infty \\ y(t,\mu) &\sim (\bar{y})_\infty + (\tilde{y})_\infty - (\hat{y})_\infty. \end{aligned} \tag{2.1.12}$$

The first series is obtained by making the change of variable $t = \tau\mu$ in (2.0.1) to obtain

$$\frac{dx}{d\tau} = \mu\, f(x,y,\mu\tau)$$
$$\frac{dy}{d\tau} = g(x,y,\mu\tau). \tag{2.1.13}$$

We then formally substitute (2.1.8) into the system (2.1.13), equate like powers in μ and solve the corresponding sets of differential equations using as initial conditions

$$\bar{x}_0(0) = \alpha, \qquad \bar{y}_0(0) = \beta$$

and

$$\bar{x}_i(0) = \bar{y}_i(0) = 0 \quad \text{for } i > 0.$$

For example the first two sets of equations are

$$\frac{d\bar{x}_0}{d\tau} = 0$$
$$\frac{d\bar{y}_0}{d\tau} = g(\bar{x}_0,\bar{y}_0,0) \tag{2.1.14}$$

and

$$\frac{d\bar{x}_1}{d\tau} = f(\bar{x}_0,\bar{y}_0,0)$$

$$\frac{d\bar{y}_1}{d\tau} = g_x(\bar{x}_0,\bar{y}_0,0)\bar{x}_1 + g_y(\bar{x}_0,\bar{y}_0,0)\bar{y}_1 + g_t(\bar{x}_0,\bar{y}_0,0)\tau$$

yielding

$$\bar{x}_0(\tau) = \alpha$$
$$\frac{d\bar{y}_0}{d\tau} = g(\alpha,\bar{y}_0,0), \quad \bar{y}_0(0) = \beta \tag{2.1.15}$$

and

$$\frac{d\bar{x}_1}{d\tau} = f(\alpha,\bar{y}_0,0)$$
$$\frac{d\bar{y}_1}{d\tau} = g_x(\alpha,\bar{y}_0,0)\bar{x}_1+g_y(\alpha,\bar{y}_0,0)\bar{y}_1+g_t(\alpha,\bar{y}_0,0)\tau. \tag{2.1.16}$$

In general, we have the differential equations

$$\frac{d\bar{x}_i}{d\tau} = f_x(\alpha,\bar{y}_0,0)\bar{x}_{i-1}+f_y(\alpha,\bar{y}_0,0)\bar{y}_{i-1}+\bar{p}_{i-2}(\tau)$$
$$\frac{d\bar{y}_i}{d\tau} = g_x(\alpha,\bar{y}_0,0)\bar{x}_i+g_y(\alpha,\bar{y}_0,0)\bar{y}_i+\bar{q}_{i-1}(\tau). \tag{2.1.17}$$

Here $\bar{p}_{i-2}$ is a polynomial function of $\bar{x}_j$, $\bar{y}_j$, $j \le i-2$ and τ and $\bar{q}_{i-1}$ is a polynomial function of $\bar{x}_j$, $\bar{y}_j$, $j \le i-1$ and τ. In this series, the functions f, g and their partials are evaluated at $(\alpha,\bar{y}_0,0)$.

The second series is obtained by formally substituting (2.1.9) into the system (2.0.1), equating like powers of μ and solving the corresponding sets of differential equations. However, the initial conditions of $\tilde{x}_i$ and $\tilde{y}_i$ for $i > 0$ are not as simple and will be given after the third series if given. The first set of equations is system (2.1.1) with initial conditions $\tilde{x}_0(0) = \alpha, \tilde{y}_0(0) = \phi(\alpha,0)$. The second set of equations is

$$\frac{d\tilde{x}_1}{dt} = f_x(\tilde{x}_0,\tilde{y}_0,t)\tilde{x}_1 + f_y(\tilde{x}_0,\tilde{y}_0,t)\tilde{y}_1 \tag{2.1.18a}$$

$$\frac{d\tilde{y}_0}{dt} = g_x(\tilde{x}_0,\tilde{y}_0,t)\tilde{x}_1 + g_y(\tilde{x}_0,\tilde{y}_0,t)y_1. \tag{2.1.18b}$$

In general, we have the system of differential and algebraic equations

$$\frac{d\tilde{x}_i}{dt} = f_x(\tilde{x}_o,\tilde{y}_o,t)\tilde{x}_i + f_y(\tilde{x}_o,\tilde{y}_o,t)\tilde{y}_i + \tilde{p}_{i-1}(t) \tag{2.1.19a}$$

$$\frac{d\tilde{y}_{i-1}}{dt} = g_x(\tilde{x}_o,\tilde{y}_o,t)\tilde{x}_i + g_y(\tilde{x}_o,\tilde{y}_o,t)\tilde{y}_i + \tilde{q}_{i-1}(t) \tag{2.1.19b}$$

Here $\tilde{p}_{i-1}$ and $\tilde{q}_{i-1}$ are functions of $\tilde{x}_j$, $\tilde{y}_j$, $j \le i-1$ and t. For the second series the functions f, g and their partials are evaluated at $(\tilde{x}_o(t),\tilde{y}_o(t),t)$.

The third series is obtained by expanding each of $\tilde{x}_i$ and $\tilde{y}_i$ in (2.1.9) as a Taylor expansion in t, with r terms, about $t = 0$. We obtain

$$\tilde{x}_i(t) = \sum_{k=0}^{r} x_{ki}t^k, \quad \tilde{y}_i(t) = \sum_{k=0}^{r} y_{ki}t^k. \tag{2.1.20}$$

We then substitute these in (2.1.9) to obtain

$$x_A = \sum_{i=0}^{r}\sum_{k=0}^{r} x_{ki}t^k\mu^i, \quad y_A = \sum_{i=0}^{r}\sum_{k=0}^{r} y_{ki}t^k\mu^i, \tag{2.1.21}$$

or keeping only some of the terms such that for the subscripts i and k in (2.1.21), $i+k \le r$, we have

$$(\hat{x})_r = \sum_{i=0}^{r}\sum_{k=0}^{i} x_{k,i-k}t^k\mu^{i-k}, \quad (\hat{y})_r = \sum_{i=0}^{r}\sum_{k=0}^{i} y_{k,i-k}t^k\mu^{i-1}$$

We may put (2.1.21) into a different form by substituting $t = \mu\tau$ to obtain

$$(\hat{x})_r = \sum_{i=0}^{r} \sum_{k=0}^{i} x_{k,i-k} \tau^k \mu^i, \quad (\hat{y})_r = \sum_{i=0}^{r} \sum_{k=0}^{i} y_{k,i-k} \tau^k \mu^i$$

or

$$(\hat{x})_r = \sum_{i=0}^{r} \hat{x}_i \mu^i, \quad (\hat{y})_r = \sum_{i=0}^{r} \hat{y}_i \mu^i \tag{2.1.22}$$

where

$$\hat{x}_i = \sum_{k=0}^{i} x_{k,i-k} \tau^k, \quad \hat{y}_i = \sum_{k=0}^{i} y_{k,i-k} \tau^k. \tag{2.1.23}$$

Instead of calculating the Taylor series (2.1.20), Wasow [16] derives each of the quantities $\hat{x}_i$ and $\hat{y}_i$ by another method. Here (2.1.22) is formally substituted into (2.1.13). Equating like powers of μ and solving the corresponding differential equations, we obtain the quantities $\hat{x}_i$ and $\hat{y}_i$. The initial conditions for these sets of differential equations are the same as those for the second term $(\tilde{x})_r$ and $(\tilde{y})_r$. Hence $\hat{x}_0(0) = \alpha$, $\hat{y}_0(0) = \phi(\alpha,0)$. For $i > 0$, $\hat{x}_i(0)$ and $\hat{y}_i(0)$ must still be specified. We have, then,

$$\begin{aligned} &\frac{d\hat{x}_0}{d\tau} = 0 \\ &\frac{d\hat{y}_0}{d\tau} = g(\hat{x}_0,\hat{y}_0,0) \quad \hat{y}_0(0) = \phi(\alpha,0) \end{aligned} \tag{2.1.24}$$

or

$$\hat{x}_0(\tau) = \alpha, \quad \hat{y}_0(\tau) = \phi(\alpha,0). \tag{2.1.25}$$

Also the second set of differential equations is

$$\begin{aligned}\frac{d\hat{x}_1}{d\tau} &= f(\alpha,\phi(\alpha,0),0)\\ \frac{d\hat{y}_1}{d\tau} &= g_x(\alpha,\phi(\alpha,0),0)\hat{x}_1 + g_y(\alpha,\phi(\alpha,0),0)\hat{y}_1 + g_t(\alpha,\phi(\alpha,0),0)\tau.\end{aligned} \tag{2.1.26}$$

In general, we have the differential equations

$$\begin{aligned}\frac{d\hat{x}_i}{d\tau} &= f_x(\alpha,\phi(\alpha,0),0)\hat{x}_{i-1} + f_y(\alpha,\phi(\alpha,0),0)\hat{y}_{i-1} + \hat{p}_{i-2}(\tau)\\ \frac{d\hat{y}_i}{d\tau} &= g_y(\alpha,\phi(\alpha,0),0)\hat{x}_i + g_y(\alpha,\phi(\alpha,0),0)\hat{y}_i + \hat{q}_{i-1}(\tau)\end{aligned} \tag{2.1.27}$$

Here $\hat{p}_{i-2}$ is a polynomial function of $\hat{x}_j$, $\hat{y}_j$, $j \le i-2$ and τ and $\hat{q}_{i-1}$ is a polynomial function of $\hat{x}_j$, $\hat{y}_j$, $j \le i-1$ and τ. For $(\hat{x})_r$ and $(\hat{y})_r$, the functions f, g and their partials are evaluated at $(\alpha,\phi(\alpha,0),0)$.

We now give the initial conditions $\tilde{x}_i(0)$ and $\hat{x}_i(0)$, $i > 0$. Vasileva gives them in the form

$$\tilde{x}_i(0) = \hat{x}_i(0) = \int_0^\infty [\bar{f}_{i-1}(\tau)-\hat{f}_{i-1}(\tau)]d\tau \tag{2.1.28}$$

where $\bar{f}(\tau)$ is the i-th coefficient in the expansion of $f(x,y,t)$ of the type (2.1.8), that is

$$\bar{f}(\bar{x}_0+\mu\bar{x}_1+\ldots,\bar{y}_0+\mu\bar{y}_1+\ldots,\mu\tau) = \bar{f}_0+\mu\bar{f}_1+\mu^2\bar{f}_2+\ldots \quad .$$

Also $\hat{f}_i(\tau)$ is the i-th coefficient in the expansion of $f(x,y,t)$ of the type (2.1.22), that is

$$\hat{f}_i(\tau) = \sum_{k=0}^{i} f_{k,i-k}\tau^k.$$

Here f_{ki} is the coefficient in the expansion of the type (2.1.20), that is

$$\tilde{f}_i(t) = \sum_{k=0}^{r} f_{k,i}t^k$$

and

$$f(\tilde{x}_0+\mu\tilde{x}_1+\ldots,\tilde{y}_0+\mu\tilde{y}_1+\ldots,t) = \tilde{f}_0+\mu\tilde{f}_1+\mu^2\tilde{f}_2+\ldots .$$

In order to obtain the third series, the quantities $\hat{y}_i(0)$ are needed. These can be obtained by setting $\hat{y}_i(0) = \tilde{y}_i(0)$. The $\tilde{y}_i(0)$ are obtained by solving the algebraic equations (2.1.19b) having first obtained the $\tilde{x}_i(0)$.

Wasow [16] defines the in tial conditions $\tilde{x}_i(0)$ and $\hat{x}_i(0)$, $i > 0$ differently but equivalently as

$$\tilde{x}_i(0) = \hat{x}_i(0) = \int_0^{\infty} \frac{d\bar{x}_i}{d\tau}(\tau) - \frac{d\hat{x}_i}{d\tau}(\tau))d\tau. \qquad (2.1.29)$$

The condition (2.1.11) guarantees the convergence of (2.1.28) and (2.1.29).

Notice that because of (2.1.17) and (2.1.27) the integrand in (2.1.29) involves functions that have already been obtained.

Vasileva's method has the advantage of providing $z(t,\mu) = \begin{pmatrix} x(t,\mu) \\ y(t,\mu) \end{pmatrix}$ as a function of μ and as such has important theoretical value. We shall make use of this in material presented later. Each of the sets of

differential equations that must be solved to form (2.1.7) are not subject to the stiffness caused by the presence of μ in (2.0.1). However, even if only answers outside the boundary layer are required, it is still necessary to solve the equations associated with (2.1.8) and (2.1.10) in order to solve for the initial conditions (2.1.29) for $(\tilde{x})_r$ and $(\tilde{y})_r$. Note we are also required to approximate the integral (2.1.29). These expansions also require extensive preparation before they can be used.

2. An Asymptotic Method by MacMillan

MacMillan [10] denotes an approximate solution to (2.0.1) by u and v. Here only v is expanded formally as

$$v = \sum_{i=0}^{r} v_i \mu^i. \tag{2.2.1}$$

Once we have obtained the v_i, $0 \le i \le r$, as functions of u and t, we may solve

$$\frac{du}{dt} = f(u,v,t) \tag{2.2.2}$$

for u.

In order to obtain the v_i, we first expand f(u,v,t) and g(u,v,t) as Taylor expansions in v about $v = v_0$. Hence, we have

$$f(u,v,t) = f(u,v_0,t) + f_y\left(\sum_{i=1}^{r} v_i \mu^i\right) + \ldots + \frac{1}{(r-1)!} f_{y(r-1)}\left(\sum_{i=1}^{r} v_i \mu^i\right)^{r-1} + U_r \tag{2.2.3}$$

and

$$g(u,v,t) = g(u,v_0,t) + g_y(\sum_{i=1}^{r} v_i\mu^i) + \ldots$$

$$+ \frac{1}{r!} g_y(r)(\sum_{i=1}^{r} v_i\mu^i)^r + V_{r+1}. \qquad (2.2.4)$$

In this section f, g and their partials are evaluated at (u,v_0,t) unless otherwise specified.

By (2.2.1), we have

$$\frac{dv}{dt} = \sum_{i=0}^{r} \frac{dv_i}{dt} \mu^i. \qquad (2.2.5)$$

Since the v_i are functions of u and t, we have

$$\frac{dv_i}{dt} = \frac{\partial v_i}{\partial u}\frac{du}{dt} + \frac{\partial v_i}{\partial t} = \frac{\partial v_i}{\partial u} f(u,v,t) + \frac{\partial v_i}{\partial t}. \qquad (2.2.6)$$

Substituting (2.2.6) in (2.2.5), we have

$$\frac{dv}{dt} = (\sum_{i=0}^{r} \frac{\partial v_i}{\partial u} \mu^i) f(u,v,t) + \sum_{i=0}^{r} \frac{\partial v_i}{\partial t} \mu^i. \qquad (2.2.7)$$

We now substitute (2.2.7) and (2.2.4) in

$$\mu\frac{dv}{dt} = g(u,v,t)$$

and use (2.2.3) to obtain

$$\left(\sum_{i=0}^{r} \frac{\partial v_i}{\partial u}\mu^{i+1}\right)\left[f+f_y\left(\sum_{i=1}^{r} v_i\ ^{i}\right) + \ldots\right.$$

$$\left. + \frac{1}{(r-1)!} f_{y(r-1)}\left(\sum_{i=1}^{r} v_i\mu^i\right)^{r-1} + U_r\right] + \sum_{i=0}^{r} \frac{\partial v_i}{\partial t}\mu^{i+1}$$

$$= g + g_y\left(\sum_{i=1}^{r} v_i\mu^i\right) + \ldots + \frac{1}{r!} g_{y(r)}\left(\sum_{i=1}^{r} v_i\mu^i\right)^r + V_{r+1} \qquad (2.2.8)$$

We now formally equate like powers in μ in (2.2.8) and obtain the equations

$$0 = g(u,v_0,t) \qquad (2.2.9,1)$$

$$\frac{\partial v_0}{\partial u} f + \frac{\partial v_0}{\partial t} = g_y v_1 \qquad (2.2.9,2)$$

$$\frac{\partial v_1}{\partial u} f + \frac{\partial v_1}{\partial t} + \frac{\partial v_0}{\partial u} f_y v_1 = g_y v_2 + \frac{1}{2!} g_{y(2)} v_1^2 \qquad (2.2.9,3)$$

$$\vdots$$

$$\frac{\partial v_{r-1}}{\partial u} f + \frac{\partial v_{r-1}}{\partial t} + \ldots = g_y v_r + \ldots \ . \qquad (2.2.9,r)$$

The method is implemented by solving the system (2.0.1) for t in the range $0 \le t \le t_0$ and then solving the system (2.2.2) by first using (2.2.1). The v_i, $i = 0,1,2,\ldots,r$ are obtained by solving the algebraic equations (2.2.9). The value of t_0 depends on the particular value of μ and so may vary from problem to problem. One suggestion is to obtain v from time to time and let t_0 be the first value of t for which $|v(t)-y(t,\mu)|$ is within a certain tolerance.

MacMillan's method provides a solution for a particular value of μ and requires the solution of only one set of differential equations. However, it does require a certain amount of preparation before implementation and requires the solving of algebraic equations at each step of the integration. Also it may only be implemented outside the boundary layer and requires some other method to integrate a stiff system through the boundary layer to provide starting values.

3. Methods due to Miranker

Miranker [11] suggests two methods. The first method called the hybrid method must be applied to problems that can be written in the form of (2.0.1). The second method, called the purely numerical method may also be applied to stiff systems where the small parameter μ may not be identified. Both methods use Vasileva's expansions (2.1.7) with $r = 1$.

Actually only (2.1.9) is used to approximate the true solution since, as is shown in Wasow [16],

$$|\bar{x}_k(\tau) - \hat{x}_k(\tau)| + |\bar{y}_k(\tau) - \hat{y}_k(\tau)| \leq ce^{-\kappa t/\mu} \quad k = 0,1,2,\ldots ,$$

where $c > 0$ and $\kappa > 0$ are constants independent of t and μ. Hence, outside the boundary layer we may neglect (2.1.8) and (2.1.10).

Both methods, however, require the initial condition $\tilde{x}_1(0)$. For the hybrid method we have from (2.1.29), (2.1.16) and (2.1.26)

$$\tilde{x}_1(0) = \int_0^\infty \left(\frac{d\bar{x}_1}{d\tau} - \frac{d\hat{x}_1}{d\tau}\right)d\tau$$

$$= \int_0^\infty [f(\alpha,\bar{y}_0(\tau),0) - f(\alpha,\phi(\alpha,0),0)]d\tau. \qquad (2.3.1)$$

Hence it is necessary to calculate $\bar{y}_0(\tau)$ on a sufficiently fine grid in order to approximate (2.3.1).

The hybrid method may be stated as follows. Solve the system (2.1.1) for $\tilde{y}_0(0) = \phi(\alpha,0)$, $\tilde{x}_0(h)$ and $\tilde{y}_0(h)$. Now solve (2.1.15) for $\bar{y}_0(\tau)$ for several values of τ, say 0, k, 2k, ..., Mk, so that we may approximate $\tilde{x}_1(0)$ in (2.3.1) by some quadrature formula. Using the initial approximation, solve (2.1.18) for $\tilde{x}_1(h)$ and $\tilde{y}_1(h)$. We may now solve (2.1.1) for $\tilde{x}_0(t)$ and $\tilde{y}_0(t)$ and (2.1.18) for $\tilde{x}_1(t)$ and $\tilde{y}_1(t)$ for $t = 2h,3h,4h,\ldots$. The approximation is then given by

$$x(t,\mu) = \tilde{x}_0(t) + \mu\tilde{x}_1(t)$$

$$y(t,\mu) = \tilde{y}_0(t) + \mu\tilde{y}_1(t).$$

For the purely numerical method, since we regard μ as unidentifiable, we write (2.0.1) as

$$\frac{dz}{dt} = b(z,t,\mu)$$

$$z|_{t=0} = \gamma$$

where $z = \begin{pmatrix} x \\ y \end{pmatrix}$ and $\gamma = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ have dimension m+n. We then obtain z(h) by some self-starting method and test the following inequality componentwise:

$$\frac{|z_j(h)-\gamma_j|}{1+|\gamma_j|} < \delta \qquad (2.3.2)$$

where δ is a prescribed tolerance level. If (2.3.2) is satisfied for all components of z we repeat this last step for the next value of t. Otherwise, we place the components of z that fail (2.3.2) in the vector y and the rest in the vector x and reject this last integration step. Instead, we examine the system

$$\frac{dx}{dt} = f(x,y,t;\mu)$$

$$\frac{dy}{dt} = G(x,y,t;\mu).$$

Here f is assumed to be analytic in μ in a neighbourhood of $\mu = 0$ and G to have a simple pole at $\mu = 0$.

Using the above assumption we obtain

$$\frac{d\tilde{x}_0}{dt} = f(\tilde{x}_0,\tilde{y}_0,t;\mu),\ \tilde{x}_0(0) = \alpha$$

$$0 = G(\tilde{x}_0,\tilde{y}_0,t;\mu)$$

which corresponds to (2.1.1) and also

$$\frac{d(\mu\tilde{x}_1)}{dt} = f_x(\mu\tilde{x}_1) + f_y(\mu\tilde{y}_1) \tag{2.3.3a}$$

$$\frac{d\tilde{y}_0}{dt} = G_x(\mu\tilde{x}_1) + G_y(\mu\tilde{y}_1) \tag{2.3.3b}$$

which corresponds to (2.1.18). Notice that since μ is unidentifiable, f, G, f_x, f_y, G_x and G_y are all evaluated at $(\tilde{x}_0,\tilde{y}_0,t;\mu)$. Also we solve (2.3.3) for $\mu\tilde{x}_1$ and $\mu\tilde{y}_1$ which are well defined quantities as soon as we specify the initial condition $\mu\tilde{x}_1(0)$. If we solve (2.3.3b) for $\mu\tilde{y}_1$ and substitute in (2.3.3a) we obtain

$$\frac{d(\mu\tilde{x}_1)}{dt} = (f_x - f_y G_y^{-1} G_x)(\mu\tilde{x}_1) - f_y G_y^{-2}(G_t + G_x f)$$

$$\mu\tilde{y}_1 = -G_y^{-1} G_x(\mu\tilde{x}_1) - G_y^{-2}(G_t + G_x f).$$

The latter uses the fact that

$$\frac{d\tilde{y}_0}{dt} = -G_y^{-1}(G_x f + G_t).$$

In order to specify the initial condition $\mu\tilde{x}_1(0)$, we must obtain an approximation to

$$\mu\tilde{x}_1(0) = \mu \int_0^\infty [f(\alpha,\bar{y}_0(\tau),0;\mu) - f(\alpha,\hat{y}_0(\tau),0;\mu)]d\tau \tag{2.3.4}$$

which corresponds to (2.3.1). Note that $\bar{y}_0(\tau)$ is the solution of

$$\frac{d\bar{y}_0}{d\tau} = \mu G(\alpha,\bar{y}_0,0;\mu), \quad \bar{y}_0(0) = \beta$$

which corresponds to (2.1.15).

Using the information

$$\bar{y}_0(0) = \beta \quad , \quad \hat{y}_0(0) = \tilde{y}_0(0)$$
$$\frac{d\bar{y}_0}{d\tau}(0) = \mu G(\alpha,\beta,0;\mu), \quad \frac{d\hat{y}_0}{d\tau} = 0 \tag{2.3.5}$$

we may use a linear approximation for the integrand in (2.3.4) and obtain the approximation

$$\mu\tilde{x}_1(0) = \frac{1}{2}\frac{[f(\alpha,\beta,0;\mu)-f(\alpha,\tilde{y}_0(0),0;\mu)]^2}{f_y(\alpha,\beta,0;\mu)G(\alpha,\beta,0;\mu)} \tag{2.3.6}$$

by integrating from 0 to the positive root. Instead, we may use the information (2.3.5) to fit an exponential to the integrand in (2.3.4) and obtain the approximation

$$\mu\tilde{x}_1(0) = \frac{f(\alpha,\tilde{y}_0(0),0;\mu)-f(\alpha,\beta,0;\mu)}{f_y(\alpha,\beta,0;\mu)G(\alpha,\beta,0;\mu)} \, . \tag{2.3.7}$$

In both cases the arithmetic is done component-wise except for the matrix vector product in the denominator.

Miranker's method then provides answers outside the boundary layer and requires some initial preparation before implementation. It also requires, as do the other methods already mentioned, the evaluation of the partial derivatives of f and g with respect to x and y at each step in the integration. The equations defining $\tilde{x}_1$ and $\tilde{y}_1$ also require the inversion of G_y at each step. The problem of obtaining the quantity

$\tilde{x}_1$ involves solving the set of differential equations defining $\bar{y}_0(\tau)$ or in the second case of evaluating (2.3.6) or (2.3.7).

The purely numerical method provides an interesting way of applying asymptotic methods associated with singular perturbation problems to problems that are not easily written in the form (2.0.1).

CHAPTER III

INTERPOLATION PROCEDURE

1. Description

We define the auxiliary systems as

$$\begin{aligned} \frac{dx}{dt} &= f(x,y,t) \\ \mu_i \frac{dy}{dt} &= g(x,y,t) \end{aligned} \qquad i = 1,2,3,\ldots,q \qquad (3.1.1)$$

with the initial conditions

$$\begin{aligned} x(0,\mu_i) &= \alpha \\ y(0,\mu_i) &= \beta \end{aligned} \qquad i = 1,2,3,\ldots,q.$$

The solutions to these systems, called the auxiliary solutions, are denoted by $z_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$ where $x_i = x_i(t,\mu_i)$ and $y_i = y_i(t,\mu_i)$. The procedure is to choose various values of μ_i, $i = 1,2,3,\ldots,q$ and integrate (3.1.1) for each value and obtain the z_i at the values of t for which we desire a solution to (2.0.1). We also obtain the values $z_0(t,0) = \begin{pmatrix} x_0(t,0) \\ y_0(t,0) \end{pmatrix}$ from (3.1.1) by setting $\mu_0 = 0$. This of course is just the solution to the degenerate system (2.1.2). We then use Lagrange interpolation with the variable μ on the values $z_i(t,\mu_i)$, $i = 0,1,2,\ldots,q$ to obtain an approximation to $z(t,\mu)$, the true solution. Diagram 3.1 shows a case with $m = 1$, $n = 1$.

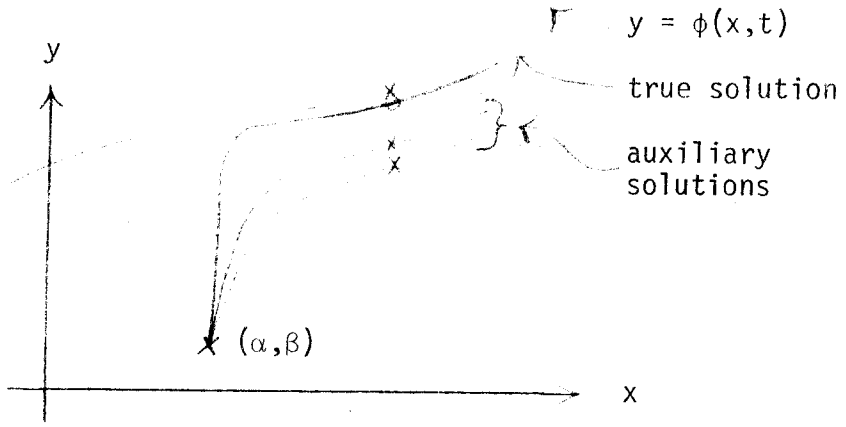

Diagram 3.1

If μ is very small we have a stiff system. Hence, we pick the μ_i, $i = 1,2,3,\ldots,q$ large enough so that the auxiliary systems are not too stiff but small enough so that we get a good approximation. The problem is to pick the value q and decide on the distribution of the μ_i, $i = 1,2,3,\ldots,q$. A look at the error expression is necessary to make these decisions.

2. Error Analysis

An examination of the errors involved will help us to choose the μ_i, $i = 1,2,3,\ldots,q$. For convenience, we let

$$0 < \mu_1 < \mu_2 < \mu_3 < \ldots < \mu_q.$$

As before $z(t,\mu)$ is the true solution to (2.0.1) and $z_i(t,\mu_i)$ $i = 0,1,2,\ldots,q$ are the true solutions to (2.1.2) and (3.1.1). We let

$w(t,\mu)$ be the approximation we are seeking and $w_i(t,\mu_i)$ be the numerical approximations to $z_i(t,\mu_i)$ for $i = 0,1,2,\ldots,q$. Also let

$$e(t,\mu) = w(t,\mu) - z(t,\mu) \tag{3.2.1}$$

and

$$e_i(t,\mu_i) = w_i - z_i,\ i = 0,1,2,\ldots,q. \tag{3.2.2}$$

Using (3.2.1) and applying Lagrange interpolation we have

$$e(t,\mu) = \sum_{i=0}^{q} L_i(\mu)w_i - z(t,\mu) \tag{3.2.3}$$

where

$$L_i(\mu) = \prod_{\substack{j=0 \\ j\neq i}}^{q} \frac{(\mu-\mu_j)}{(\mu_i-\mu_j)}\ ,\quad i = 0,1,2,\ldots,q. \tag{3.2.4}$$

Substituting (3.2.2) in (3.2.3) we obtain

$$e(t,\mu) = \sum_{i=0}^{q} L_i(\mu)[e_i+z_i]-z(t,\mu)$$

which becomes

$$e(t,\mu) = \sum_{i=0}^{q} L_i(\mu)e_i + \sum_{i=0}^{q} L_i(\mu)z_i - z(t,\mu)$$

or

$$e(t,\mu) = L_0e_0 + \sum_{i=1}^{q} L_i(\mu)e_i - \frac{L(\mu)}{(q+1)!}\frac{\partial^{q+1}z}{\partial\mu^{q+1}} \tag{3.2.5}$$

where each component of $\dfrac{\partial^{q+1}z}{\partial\mu^{q+1}}$ is evaluated at t and in general different values of $\bar{\mu}$ but where each $\bar{\mu}$ satisfies $0 \le \bar{\mu} \le \mu_q$.

$$L(\mu) = (\mu-\mu_0)(\mu-\mu_1)(\mu-\mu_2)\ldots(\mu-\mu_q)$$

$$= \mu(\mu-\mu_1)(\mu-\mu_2)\ldots(\mu-\mu_q). \qquad (3.2.6)$$

The last term in (3.2.5) is the error term in applying Lagrange interpolation assuming exact values for the auxiliary solutions. The first two terms of (3.2.5) are the errors associated with the numerical approximations to the degenerate system (2.1.2) and the auxiliary systems (3.1.1) respectively.

Examining the first factor of (3.2.6) we see that (3.2.5) implies this method provides us with at least a first order approximation and that the stiffer the system (the smaller μ is) the more accurate our approximation is (as is the case with most methods that handle stiff systems). We also see from (3.2.6) that the error caused by using μ_i in the interpolation formula is $O(\mu_i)$. Hence, decreasing μ_i decreases the error but it increases the stiffness of the auxiliary system and hence the cost of integrating it. Each auxiliary solution then should provide an additional correction to our approximation.

The quantity $\frac{\partial^{q+1} z}{\partial \mu^{q+1}}$ in (3.2.5) has significance. It might seem that we may make $e(t,\mu)$ arbitrarily small by picking q large enough. However, in general, the quantity $\frac{\partial^{q+1} z}{\partial \mu^{q+1}}$ becomes arbitrarily large as $q \to \infty$. Hence, the same type of asymptotic characteristic is shown here as $q \to \infty$ as for Vasileva's expansions (2.1.7) as $r \to \infty$. That is, in general, the expansions (2.1.7) diverge as $r \to \infty$ although they give a good

approximation for r and μ small. In examples run using the interpolation scheme a value of q > 5 has given no further advantage.

An estimation of the derivative $\frac{\partial^{q+1} z}{\partial \mu^{q+1}}$ in (3.2.5) is necessary here. To do this we obtain an asymptotic expansion first for $\frac{\partial z}{\partial \mu}$ and then for $\frac{\partial^s z}{\partial \mu^s}$ for any s. From Vasileva's expansions (2.1.12) we may write

$$z(t,\mu) \sim (\bar{z})_\infty + (\tilde{z})_\infty - (\hat{z})_\infty. \tag{3.2.7}$$

It may be shown that $\frac{\partial z}{\partial \mu}(t,\mu)$ exists for $0 \le \mu \le \bar{\mu}$ and $0 < t_1 \le t \le T$ for some $\bar{\mu}$ and T (see Appendix A or Vasileva [14]). Hence, $\frac{\partial z}{\partial \mu}$ may be represented asymptotically in this region by formally differentiating the right side of (3.2.7). Using (2.1.8), (2.1.9) and (2.1.10) we obtain

$$\frac{\partial z}{\partial \mu} \sim \sum_{i=1}^{\infty} i[\bar{z}_i + \tilde{z}_i - \hat{z}_i]\mu^{i-1}$$

$$-t \sum_{i=0}^{\infty} [\bar{z}^{(1)}(\tau) - \hat{z}^{(1)}(\tau)]\mu^{i-2} \tag{3.2.8}$$

where $\tau = t/\mu$, $\bar{z}^{(1)}(\tau)$ means $d\bar{z}/d\tau$ and $\hat{z}^{(1)}(\tau)$ means $d\hat{z}/d\tau$. In fact, Vasileva [14] has shown that $\frac{\partial^q z}{\partial \mu^q}$ exists on $0 \le \mu \le \bar{\mu}$ and any interval $0 < t_1 \le t \le T$ and therefore we may represent $\frac{\partial^q z}{\partial \mu^q}$ asymptotically by repeatedly differentiating (3.2.8). Doing this we obtain

$$\frac{\partial^q z}{\partial \mu^q} \sim \sum_{i=q}^{\infty} i(i-1)\ldots(i-q+1)\tilde{z}_i(t)\mu^{i-q}$$

$$+ \sum_{j=0}^{q} (-t)^j \sum_{i=0}^{\infty} v_{ij}[\bar{z}_i^{(j)}(\tau) - \hat{z}_i^{(j)}(\tau)]\mu^{i-(j+q)} \tag{3.2.9}$$

where the V_{ij} are polynomials in i and $\bar{z}^{(j)}(\tau)$ means $\frac{d^j\bar{z}}{d\tau^j}(\tau)$, $\hat{z}^{(j)}(\tau)$ means $\frac{d^j\hat{z}}{d\tau^j}(\tau)$.

We get a first approximation to $\frac{\partial^q z}{\partial\mu^q}$ by taking the first term for i in each of the sums in (3.2.9). We have

$$\frac{\partial^q z}{\partial\mu^q} \doteq q!\tilde{z}_q(t)+\sum_{j=0}^{q}(-t)^j V_{oj}[\bar{z}^{(j)}(\tau)-\hat{z}^{(j)}(\tau)]\mu^{-(j+q)}. \qquad (3.2.10)$$

Let $M_1 = \max_{0\le t\le T} q!|\tilde{z}_q(t)|$. We are assured M_1 exists since $\tilde{z}_q(t)$ is the solution of (2.1.19) with i = q. Let $M_2 = \max_{\substack{0\le t\le T\\ 0\le j\le q}} |(-t)^j V_{oj}|$. Using (3.2.10) we have

$$|\frac{\partial^q z}{\partial\mu^q}| \le M_1 + M_2 \sum_{j=0}^{q}|\bar{z}^{(j)}(\tau) - \hat{z}^{(j)}(\tau)|\mu^{-(j+q)}. \qquad (3.2.11)$$

We may show $|\bar{z}_i^{(j)}(\tau)-\hat{z}_i^{(j)}(\tau)| \le ce^{-\kappa t/\mu}$ (see Appendix B) where $c > 0$ and $\kappa > 0$ are constants independent of t and μ. For q finite we may also consider c and κ independent of j. Using (3.2.11) we have

$$|\frac{\partial^q z}{\partial\mu^q}| \le M_1 + M_2 ce^{-\kappa t/\mu} \sum_{j=0}^{q} \mu^{-(j+q)}$$

or letting $M_3 = cM_2$

$$|\frac{\partial^q z}{\partial\mu^q}| \le M_1 + (\frac{1-\mu^{q+1}}{1-\mu})M_3 e^{-\kappa t/\mu}\ \mu^{-2q}.$$

To a first approximation we have

$$\left|\frac{\partial^q z}{\partial \mu^q}\right| \leq M_1 + M_3 e^{-\kappa t/\mu}\, \mu^{-2q}. \tag{3.2.12}$$

For this expression to be bounded by some quantity M_4 we have

$$M_1 + M_3 e^{-\kappa t/\mu}\, \mu^{-2q} < M_4.$$

Solving for t we have

$$t > \frac{2q}{\kappa}\, \mu |\ell n \mu| + \mu \frac{M_5}{\kappa}$$

where $M_5 = -\ell n\left(\frac{M_4 - M_1}{M_3}\right)$ and M_4 is chosen so that

$$0 < \frac{M_4 - M_1}{M_3} < 1.$$

Hence, we have

$$t > M_6 \mu |\ell n \mu| + \mu M_7 \tag{3.2.13}$$

where M_6 and M_7 are positive constants independent of μ.

The result (3.2.13) shows that t must be outside the boundary layer, that is t given by (3.2.13), before a reasonable approximation can be expected. In fact, since $\frac{\partial^{q+1} z}{\partial \mu^{q+1}}$ in (3.2.5) is evaluated at $\bar{\mu}$, where $0 \leq \bar{\mu} \leq \mu_q$, this implies we may or may not have a reasonable approximation until t is $O(\mu_q |\ell n \mu_q|)$.

A comment about the accuracy of the degenerate and auxiliary solutions is necessary. In equation (3.2.5) we have separated the error introduced by solving the degenerate system (2.1.2) numerically from the error introduced by solving the auxiliary systems (3.1.1) numerically. Looking at (3.2.4) we have with $\mu_0 = 0$ that

$$L_0(\mu) = \prod_{j=1}^{q} \frac{\mu-\mu_j}{-\mu_j} \doteq 1$$

since $\mu_j >> \mu$, $j = 1,2,3,\ldots,q$. This implies that the degenerate system must be solved as accurately as we wish our approximation $w(t,\mu)$ to be. However, looking at (3.2.4) again we have

$$L_i(\mu) = \frac{\mu}{\mu_i} \prod_{\substack{j=1 \\ j\neq i}}^{q} \frac{\mu-\mu_j}{\mu_i-\mu_j}, \quad i = 1,2,3,\ldots,q$$

and this implies that we may tolerate much larger errors e_i in the auxiliary solutions. The errors e_i may be specified by

$$e_i \doteq \frac{\mu_i}{\mu} \prod_{\substack{j=1 \\ j\neq i}}^{q} \frac{\mu_i-\mu_j}{\mu-\mu_j} e_0, \quad i = 1,2,3,\ldots,q.$$

We may also see that making the μ_i smaller implies we must make e_i smaller if we are to keep $e(t,\mu)$ within a specified tolerance.

In what follows we will be concerned with the relative error and we will be using the L_∞ norm. Therefore, using (3.2.5) we write

$$|e(t,\mu)| \leq |L_0||e_0(t)| + \sum_{i=0}^{q} |L_i||e_i| + \frac{|L(\mu)|}{(q+1)!} \left|\frac{\partial^{q+1} z}{\partial \mu^{q+1}}\right|.$$

This gives us

$$\frac{|e(t,\mu)|}{|z(t,\mu)|} \leq |L_0| \frac{|e_0(t)|}{|z(t,\mu)|} + \sum_{i=0}^{q} |L_i| \frac{|e_i|}{|z(t,\mu)|} + \frac{|L(\mu)|}{(q+1)!} \cdot \frac{\left|\frac{\partial^{q+1} z}{\partial \mu^{q+1}}\right|}{|z(t,\mu)|}. \quad (3.2.14)$$

Since $z_0(t)$ is a first approximation to $z(t,\mu)$ for $t > M_6\mu|\ell n\mu| + \mu M_7$ then for sufficiently small μ, we write (3.2.14) in the more useful form

$$\frac{|e(t,\mu)|}{|z(t,\mu)|} \leq |L_0||\varepsilon_0(t)| + \sum_{i=0}^{q} |L_i||\varepsilon_i(t)| + \frac{|L(\mu)|}{(q+1)!} \cdot \frac{\left|\frac{\partial^{q+1} z}{\partial \mu^{q+1}}\right|}{|z_0(t)|}. \quad (3.2.15)$$

where

$$|\varepsilon_0(t)| = \frac{|e_0(t)|}{|z_0(t)|} \quad (3.2.16)$$

and

$$|\varepsilon_i(t)| = \frac{|e_i(t)|}{|z_0(t)|}, \quad i = 1,2,3,\ldots,q. \quad (3.2.17)$$

CHAPTER IV

NUMERICAL CONSIDERATIONS AND IMPLEMENTATION

In the following implementation we assume that a specified relative error tolerance ε is given. The algorithm proposed strives to provide answers within ε or failing that gives an estimate of the error incurred. The L_∞ norm is used throughout. For convenience, we use

$$|\varepsilon_0(t)| = \frac{|e_0(t)|}{Z}$$

and

$$|\varepsilon_i(t)| = \frac{|e_i(t)|}{Z}, \quad i = 1,2,3,\ldots,q$$

instead of (3.2.16) and (3.2.17), where

$$Z = \max(1,|(\overset{\alpha}{\phi}(\alpha,0))|)$$

We note here that the fourth-order classical Runge-Kutta method has been used to integrate the degenerate and auxiliary systems.

1. Methods used to Integrate the Degenerate and Auxiliary Systems

Our first consideration is obtaining the solution to the degenerate system (2.1.2). We have assumed that there are no problems concerning stiffness with the system (2.1.2) and that we may obtain the correct solution $y = \phi(x,t)$ of $g(x,y,t) = 0$ easily. In this case we have

used the classical fourth-order Runge-Kutta method in a one-step, two-step manner so that we expect to be within a specified error tolerance. We have already noted that this error tolerance ε_o must be at least as small as the error tolerance ε for the over all problem.

The one-step, two-step procedure is done the following way. If we assume we have the values of $z_o(t^*)$ and that the current step size in the integration is h, then we calculate $z_o^h(t^*+kh)$ and $z_o^{h/2}(t^*+kh)$ where the superscript denotes the step size used and k is an integer set at the start. We then test

$$\frac{|z_o^h(t^*+kh)-z_o^{h/2}(t^*+kh)|}{Z} \leq \varepsilon_o. \tag{4.1.1}$$

If (4.1.1) is not satisfied we test

$$\frac{|z_o^{h/2^i}(t^*+kh)-z_o^{h/2^{i+1}}(t^*+kh)|}{Z} \leq \varepsilon_o$$

for i = 1,2,3,... until it is satisfied and we accept the value of $z_o^{h/2^{i+1}}(t^*+kh)$ as $z_o(t^*+kh)$ and reapply the procedure. If (4.1.1) is satisfied then we accept $z_o^{h/2}(t^*+kh)$ as $z_o(t^*+kh)$ and test

$$\frac{|z_o^h(t^*+kh)-z_o^{h/2}(t^*+kh)|}{Z} < \frac{\varepsilon_o}{100}. \tag{4.1.2}$$

If (4.1.2) is satisfied we double h before reapplying the procedure. If $z_0(t_1)$ is the value we are interested in and if $t^*+kh > t_1$, then we temporarily redefine $k = [\frac{t_1-t^*}{h}]$ and apply the procedure. If in this case $k = 0$ we redefine $h = t_1-t^*$ and apply the procedure with $k = 1$.

The solution of (2.1.2) is more difficult if $\phi(x,t)$ is not easily obtained. In this case consideration should be given to using a method that minimizes the number of function evaluations since using a fourth order Runge-Kutta method for example would involve four calculations of $f(x,\phi(x,t),t)$ and hence four calculations of $\phi(x,t)$ at each step. An explicit multi-step method would require only one calculation of $\phi(x,t)$ as long as we saved previous values.

If for any reason an implicit multi-step method were chosen the following should be considered. Let the r-step formula be given by

$$x^{(p)} = \sum_{i=1}^{r} a_i x^{(p-i)} + h \sum_{i=0}^{r} b_i f(x^{(p-i)}, y^{(p-i)}, t^{(p-i)}).$$

Then applying this to (2.1.1) we have

$$x_0^{(p)} - hb_0 f(x_0^{(p)}, y_0^{(p)}, t_0^{(p)}) = R \qquad (4.1.3)$$

$$g(x_0^{(p)}, y_0^{(p)}, t_0^{(p)}) = 0 \qquad (4.1.4)$$

where R contains only values that are known. The normal procedure might be to use the Newton-Raphson method to solve (4.1.4) for $y_0^{(p)} = \phi(x_0^{(p)},t)$ and then Newton-Raphson on (4.1.3). But note that (4.1.4) must be solved for each iteration in (4.1.3) and that in addition the quantity $\frac{\partial \phi}{\partial x}$ is also

needed at each iteration in (4.1.3). The other procedure might be to solve (4.1.3) and (4.1.4) simultaneously, introducing a large algebraic system with m+n unknowns. In particular cases we should examine the algebraic structure of (4.1.3) and (4.1.4) and consult Ortega and Rheinboldt [13] for example.

Hull, Enright, Fellen and Sedgwick [7] have collected statistics in comparing various methods applied to various problems using different tolerance levels. They suggest that a method by Bulirsch and Stoer is best when function evaluations are not expensive. However, if function evaluations are expensive a method due to Krogh is recommended. Krogh's method is an implementation of a variable order Adams method. In general, Runge-Kutta methods are not recommended especially for stringent tolerances.

As far as the auxiliary systems (3.1.1) are concerned we have used the same one-step, two-step method used for the degenerate system. In relation to the full system (2.0.1) these systems are moderately stiff. As we have mentioned Hull, Enright, Fellen and Sedgwick [7] do not recommend the use of a Runge-Kutta method. However, we have selected a Runge-Kutta method for the following reasons. As we have pointed out in Chapter III, the auxiliary systems need only be solved with a much more relaxed error tolerance. Hence, as soon as we have found a step size that integrates the system stably, we are probably very close to a value of h that satisfies our error tolerance.

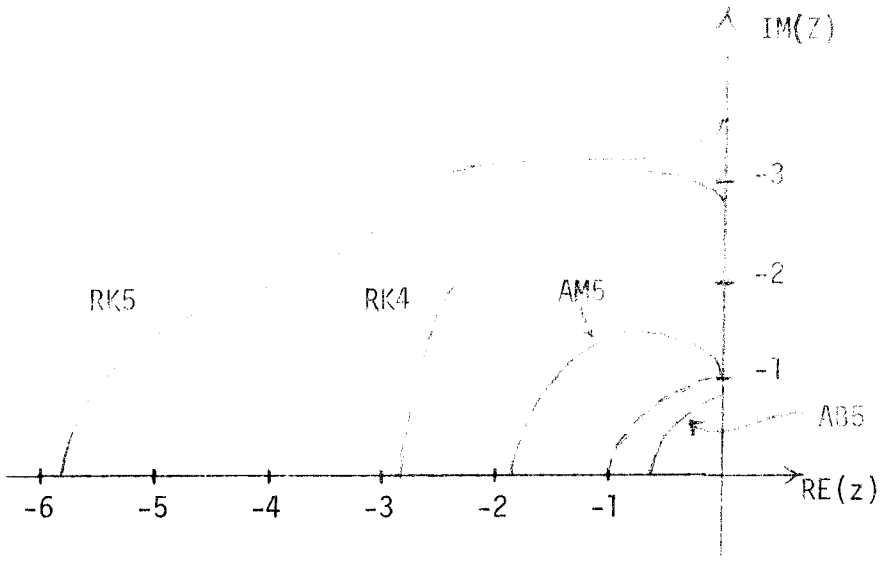

Diagram 4.1

In Diagram 4.1 we have shown the stability regions for the Adams-Bashforth method of order 5 (AB5), Adams-Moulton method of order 5 (AM5), the classical Runge-Kutta method of order 4 (RK4) and a fifth order Runge-Kutta method (RK5) due to Lawson [8]. These graphs were obtained from Gear [5] and Lawson [8]. We note that the stability regions for Runge-Kutta methods are usually larger than for multi-step methods. For example, using AB5 and one correction with AM5 increases the stability region as shown roughly by the dotted line in Diagram 4.1. This means we have about two function evaluations for one unit as compared to the RK5 process which requires six function evaluations giving approximately one function evaluation for one unit.

The increased region of stability either allows a larger value of the step size or a smaller value for the auxiliary parameter (or some combination of both), since as noted earlier a smaller value of μ_i causes a larger value for the Lipschitz constant. This is important since smaller values of μ_i either allow q to be smaller or allow smaller values of ε for a specific amount of computing.

We have not investigated any other methods except the classical fourth order Runge-Kutta method but the fifth order Runge-Kutta method due to Lawson seems particularly suited because of its extended stability region as seen in Diagram 4.1. The multistep methods recommended in Hull, Enright, Fellen and Sedgwick [7] would not appear to be of advantage here for the same reasons as discussed with the Adams methods. The extrapolation procedures of Bulirsch and Stoer have not been examined. It should be noted that the conclusions given in this paper were for nonstiff systems.

2. Implementation of the Interpolation Scheme

At this point, we assume we are given the parameters μ_i, $i = 1,2,3,\ldots,q$. We then use Aitken's interpolation in the following manner. For convenience we define

$$z_{io}(t) = z_i(t), \quad i = 0,1,2,\ldots,q.$$

The degenerate solution $z_{oo}(t)$ and the first auxiliary solution are then used to form $z_{11}(t)$ as

$$z_{11} = \frac{(\mu_1-\mu)z_{00}+\mu z_{10}}{\mu_1} . \qquad (4.2.1)$$

If the test

$$\frac{|z_{11}(t)-z_{00}(t)|}{Z} \leq \varepsilon \qquad (4.2.2)$$

is satisfied we accept $z_{11}(t)$. Note that in this case we expect only that the degenerate solution which satisfies

$$z(t,\mu) = z_{00}(t) + O(\mu)$$

outside the boundary layer also has a relative error of ε for this particular μ.

If (4.2.2) is not satisfied we form

$$z_{ij} = \frac{(\mu_i-\mu)z_{j-1,j-1}-(\mu_{j-1}-\mu)z_{i,j-1}}{\mu_i-\mu_{j-1}}, \; j = 1,2,3,\ldots,i \qquad (4.2.3)$$

for each value of $i = 2,3,4,\ldots$ until the relation

$$\frac{|z_{i,i}(t)-z_{i-1,i-1}(t)|}{Z} \leq \varepsilon \qquad (4.2.4)$$

is satisfied. We then accept the value of $z_{ii}(t)$.

The advantage of using Aitken interpolation here is that we simply integrate and apply each auxiliary system as necessary and we do not integrate those auxiliary systems that are not necessary. This becomes important for many examples where for larger values of t we are able to drop off some of the auxiliary parameters.

In implementing this procedure we simply accept the value of $z_{ii}(t)$ for which the quantity

$$\frac{|z_{ii}(t)-z_{i-1,i-1}(t)|}{Z}, \quad i = 1,2,3,\ldots,q$$

is a minimum if (4.2.4) is not satisfied for $i = 1,2,3,\ldots,q$.

3. Selection of the Auxiliary Parameters

The efficiency of the interpolation scheme depends on the selection of the auxiliary parameters μ_i, $i = 1,2,3,\ldots,q$. Using the error expression (3.2.15) we may assume that we may make ε_i, $i = 0,1,2,\ldots,q$ as small as we like. If the auxiliary parameters are chosen too large then the last term of (3.2.15) may be too large to satisfy the error tolerance ε. This could happen two ways. The first obviously is that $L(\mu)$ may be too large. The second is that the effective boundary layer, that is for $t < M_6\mu|\ell n\mu| + \mu M_7$, may be pushed beyond a value of t for which we desire a solution.

If the auxiliary parameters are chosen too small then the auxiliary systems will become too stiff to integrate efficiently and the error tolerances ε_i, $i = 1,2,3,\ldots,q$ become smaller.

The value of q must also be chosen. If q is too small, then the auxiliary parameters must be smaller so that $L(\mu)$ is small enough. If q is too large the quantity $\frac{\partial^q z}{\partial \mu^q}$ may be too large.

The implementation of this method allows for the auxiliary parameters to be chosen by the user or to be chosen for the user. The approach taken for the selection of the parameters for the user is to find some rough approximation to the quantity $|\frac{\partial^q z}{\partial \mu^q}|$. To do this we use the relation (3.2.12) as

$$|\frac{\partial^q z}{\partial \mu^q}| \leq M_1 + \frac{M_3 e^{-\kappa t/\bar{\mu}}}{\bar{\mu}^{2q}} \tag{4.3.1}$$

to describe the behaviour.

We would like to choose the auxiliary parameters so that the effects of the second term in (4.3.1) have been reduced. For any particular example we choose a starting, trial set of auxiliary parameters ν_i, $i = 1,2,3,\ldots,q$. A way of choosing these will be explained later but we may assume that $\nu_i = 2\nu_{i-1}$, $i = 2,3,4,\ldots,q$. We then form the Lagrange interpolation approximation $w^{(i)}(t,\mu)$, $i = q,q-1,q-2$, to $z(t,\mu)$ where $w^{(i)}(t,\mu)$ uses the auxiliary solutions $z_i(t,\nu_i)$ and $z_{i-1}(t,\nu_{i-1})$ and the degenerate solution $z_0(t)$. Since $\nu_q > \nu_{q-1} > \nu_{q-2} > \nu_{q-3}$ we may assume that the relative error $e^{(i)}(t,\mu) = \frac{w^{(i)}(t,\mu)-z(t,\mu)}{Z}$ may be approximated roughly by

$$e^{(i)}(t,\mu) = \frac{w^{(i)}(t,\mu)-w^{(i-1)}(t,\mu)}{Z}, \quad i = q,q-1.$$

Using the error formula for Lagrange interpolation we may calculate an approximation for the quantities $\left|\frac{\partial^3 z^{(i)}}{\partial \mu^3}\right|$ by

$$\frac{}{Z}$$

$$\gamma^{(i)} = \frac{\left|\frac{\partial^3 z(i)}{\partial \mu^3}\right|}{Z} = \frac{3!\,|e^{(i)}|}{\mu(\mu-\nu_i)(\mu-\nu_{i-1})}, \quad i = q, q-1.$$

If the test

$$\frac{|\gamma^{(q)} - \gamma^{(q-1)}|}{\min_{i=q,q-1} \gamma^{(i)}} < \delta \tag{4.3.2}$$

where δ is some preassigned tolerance level, is satisfied then we may assume that for this set of ν_i, $i = 1,2,3,\ldots,q$ the effects of the second term of (4.3.1) have been eliminated. If (4.3.2) is not satisfied then we divide each ν_i, $i = 1,2,3,\ldots,q$ by 2 and reapply the procedure. Note that in the latter case we need only integrate one more auxiliary system corresponding to ν_{q-3}, form one more Lagrange interpolation using ν_{q-2} and ν_{q-3}, form the error $e^{(q-1)}(t,\mu)$ and form the quantity $\gamma^{(q-1)}$. All the other quantities are available from the previous iteration.

Assuming we have a set of ν_i, $i = 1,2,3,\ldots,q$ such that the effects of $\frac{e^{-\kappa t/\theta_1}}{\theta_1^6}$ have been eliminated, where $0 \le \theta_1 \le \nu_q$, we now assume that the effects of $\frac{e^{-\kappa t/\theta_2}}{\theta_2^{2q}}$ have also been eliminated where $0 \le \theta_2 \le \nu_{q-1}$. This is certainly true for a sufficiently large value of t since θ_2 is likely to be smaller than θ_1. Even in the examples where a slight problem is encountered and the tolerance ε is not quite reached at this value of t, this might be considered a small price to pay for a superior set of auxiliary parameters when applied at larger values of t.

With this assumption we form the table z_{ij}, $i = 0,1,2,\ldots,q$ and $j = 0,1,2,\ldots,i$ as in Aitken interpolation using the equations (4.2.1) and (4.2.3) with the parameters ν_i instead of μ_i, $i = 1,2,3,\ldots,q$. The relative error $\bar{e}$ using q-1 auxiliary parameters can be estimated roughly by

$$\bar{e} = \frac{z_{q-1,q-1} - z_{q,q}}{Z}$$

and this may be used in the error expression for Lagrange interpolation to estimate the quantity $\frac{|\frac{\partial^q z}{\partial \mu^q}|}{Z}$ by the equation

$$\frac{|\frac{\partial^q z}{\partial \mu^q}|}{Z} = \frac{q!}{|L_1(\mu)|}|\bar{e}| \tag{4.3.3}$$

where

$$L_1(\mu) = \mu \prod_{i=1}^{q-1} (\mu - \nu_i). \tag{4.3.4}$$

We now use this rough approximation for $\frac{|\frac{\partial^q z}{\partial \mu^q}|}{Z}$ to obtain a suitable range $[\mu_{min},\mu_{max}]$ for the auxiliary parameters μ_i, $i = 1,2,3,\ldots,q$ that will be used in the actual solution to (2.0.1). Any of the quantities ν_i, $i = 1,2,3,\ldots,q$ that satisfy $\mu_{min} \leq \nu_i \leq \mu_{max}$ will be kept since we already have the solutions for these parameters. It would be a waste to discard any acceptable ν_i especially since we have integrated its system through the most expensive range of t.

We calculate this range $[\mu_{min},\mu_{max}]$ as if μ_{min} and μ_{max} were the first and last values of μ_i, $i = 1,2,3,\ldots,q$ such that the μ_i are equally spaced. Also we should pick these μ_i such that the error expression for Lagrange interpolation is the same as the error tolerance ε. Hence, we have

$$\frac{|L(\mu)|}{q!}\,\frac{|\frac{\partial^q z}{\partial \mu^q}|}{Z} = \varepsilon \tag{4.3.5}$$

where $L(\mu)$ is given by

$$L(\mu) = \mu \prod_{i=1}^{q-1} (\mu-\mu_i). \tag{4.3.6}$$

Using (4.3.3) and (4.3.5) and solving for $|L(\mu)|$ we have

$$|L(\mu)| = \frac{\varepsilon|L_1(\mu)|}{|\bar{e}|}. \tag{4.3.7}$$

Since the μ_i are equally spaced we have from (4.3.6)

$$\begin{aligned} |L(\mu)| &\doteq \mu \prod_{i=1}^{q-1} \mu_i \\ &= \mu\mu_1^{q-1} \prod_{i=1}^{q-1} i \\ &= \mu\mu_{min}^{q-1}(q-1)!. \end{aligned} \tag{4.3.8}$$

We may use (4.3.8) along with (4.3.4) and (4.3.7) to solve for μ_{min} by the equation

$$\mu_{min} = \left(\frac{\varepsilon \left| \prod_{i=1}^{q-1} (\mu - \nu_i) \right|}{|\bar{e}|(q-1)!} \right)^{\frac{1}{q-1}} .$$

We then have $\mu_{max} = (q-1)\mu_{min}$.

As we have stated we keep the ν_i, $i = 1,2,3,\ldots,q$ such that $\mu_{min} \leq \nu_i \leq \mu_{max}$. We will call these values μ_i, $i = 1,2,3,\ldots,s$. We still need to find μ_i, $i = s+1,s+2,\ldots,q$ if $s \neq q$. However, the μ_i must still satisfy (4.3.7) and therefore we have

$$\prod_{i=1}^{s} (\mu - \mu_i) \cdot \prod_{i=s+1}^{q-1} (\mu - \mu_i) = \frac{\varepsilon |L_1(\mu)|}{|\bar{e}|}$$

which we may write approximately as

$$\prod_{i=s+1}^{q-1} \mu_i = \frac{\varepsilon |L_1(\mu)|}{|\bar{e}| \prod_{i=1}^{s} (\mu - \mu_i)} .$$

If we assume that the μ_i, $i = s+1,s+2,\ldots,q$ are equally spaced we have using (4.3.4) again that

$$\mu_{s+1} = \left(\frac{\varepsilon \prod_{i=1}^{q-1} (\mu - \nu_i)}{\bar{e}(q-s-1)! \prod_{i=1}^{s} (\mu - \mu_i)} \right)^{\frac{1}{q-s-1}} .$$

We also have

$$\mu_{s+i} = i\,\mu_{s+1},\ i = 2,3,\ldots,q-s.$$

Of course, if any of the μ_i, $i = s+1,s+2,\ldots,q$ are chosen close to any of the μ_i, $i=1,2,3,\ldots s$ we adjust the former slightly.

It only remains to find a way to choose the starting, trial set of auxiliary parameters, ν_i, $i = 1,2,3,\ldots,q$. They should be chosen so that θ_1 satisfies

$$\frac{e^{-\kappa t/\theta_1}}{\theta_1^6} \leq A \tag{4.3.9}$$

where $0 < \theta_1 < \nu_q$, and A is some tolerance level. The relation (4.3.9) may be rearranged as

$$\theta_1 \ell n\ \theta_1 + \frac{\ell n\ A}{6}\theta_1 - \frac{\kappa t}{6} > 0.$$

The function

$$P(\theta_1) = \theta_1 \ell n\ \theta_1 + \frac{\ell n\ A}{6}\theta_1 + \frac{\kappa t}{6}$$

is strictly positive if t is large enough and in this case we arbitrarily set $\nu_q = \frac{1}{100}$. We should note that ν_q should be small enough that the asymptotic nature of (3.2.9) is preserved.

If t is small enough the function $P(\theta_1)$ has two roots, r_1 and r_2 such that if $r_1 < r_2$ then

$$P(\theta_1) \begin{cases} > 0 & \text{if } 0 \leq \theta_1 < r_1 \\ < 0 & \text{if } r_1 < \theta_1 < r_2 \\ > 0 & \text{if } r_2 < \theta_1 \end{cases}$$

We therefore solve $P(\theta_1) = 0$ for r_1 and let $\nu_q = r_1$. If $r_1 > \frac{1}{100}$ we set $\nu_q = \frac{1}{100}$. We then let

$$\nu_i = \frac{\nu_{i+1}}{2}, \quad i = q-1, q-2, \ldots, 1$$

A note on the solution of $P(\theta_1) = 0$ is in order here. Since the minimum value of $P(\theta_1)$ occurs at $\rho_0 = e^{-(1+\frac{\ell n\ A}{6})}$ and $r_1 < \rho_0 < r_2$ we choose ρ_0 as a starting value and form

$$\rho_i = \frac{\rho_{i-1}}{10}, \quad i = 1,2,3,\ldots$$

until we have $P(\rho_{i-1}) < 0$ and $P(\rho_i) > 0$. If Newton's method is guaranteed to converge using ρ_i as a starting value we then use Newton's method. If convergence is not guaranteed we form

$$\rho = \frac{\rho_{i-1}+\rho_i}{2} . \tag{4.3.10}$$

If $P(\rho) > 0$ we let $\rho_i = \rho$ and reapply the test to use Newton's method. If $P(\rho) < 0$ we let $\rho_{i-1} = \rho$ and recalculate ρ using (4.3.10). This procedure is guaranteed to find r_1.

4. Comments on the Actual Subroutine that Implements the Procedure

A listing of the subroutine, written in FORTRAN, that implements this interpolation scheme may be found in Appendix C. The purpose of the subroutine is to allow the user to give starting values of z at t_0 and to

obtain approximations to $z(t_i,\mu)$ at values of t_i, $i = 1,2,3,\ldots$. For each value of t_i, the user must reference the subroutine again. As the explanation of the procedures used has shown, a maximum error tolerance ε should be given and the subroutine will try to give answers within that error tolerance. If it cannot, it will return an estimate of what error was made. In this case, the user should reset ε for the next step because it is quite possible to obtain satisfactory answers at the next value of t. However, ε should never be made smaller than the initial value since this value is used to calculate an error tolerance for the degenerate system and the auxiliary systems. It would be impossible to use a smaller value of ε during subsequent steps without recalculating the auxiliary solutions from the start or suffering a boundary layer effect in an interval of t for which there should be none. The cause for the latter case will be explained later.

On the first call on the subroutine the user has the option of choosing the auxiliary parameters or allowing the subroutine to choose the auxiliary parameters by setting a flag in the parameter list. For subsequent calls this flag will already have been set on the previous call so that all the preliminary routines may be avoided. Also for subsequent calls the subroutine saves the values of the degenerate solution and the auxiliary solutions plus the values of t for each of the auxiliary solutions. The latter is necessary since nothing is done to an auxiliary solution unless it is needed and it might be possible, although not probable, that more auxiliary solutions would be necessary at later steps.

The user may use this procedure to obtain highly accurate answers for values of t such that the effects of the quantity $\left|\frac{e^{-\kappa t/\bar{n}}}{\bar{\mu}^{2q}}\right|$ have been eliminated. If answers for smaller values of t are desired first, the user must be content with using smaller auxiliary parameters. However, it is possible to change the selection of auxiliary parameters for larger values of t and this might be quite desirable. Doing this after obtaining answers at some value t_1 would effectively mean restarting the whole procedure at t_1, including all the initialization. Unfortunately, this also involves a boundary layer effect. To see this we examine Diagram 4.2 which shows an example with n = 1.

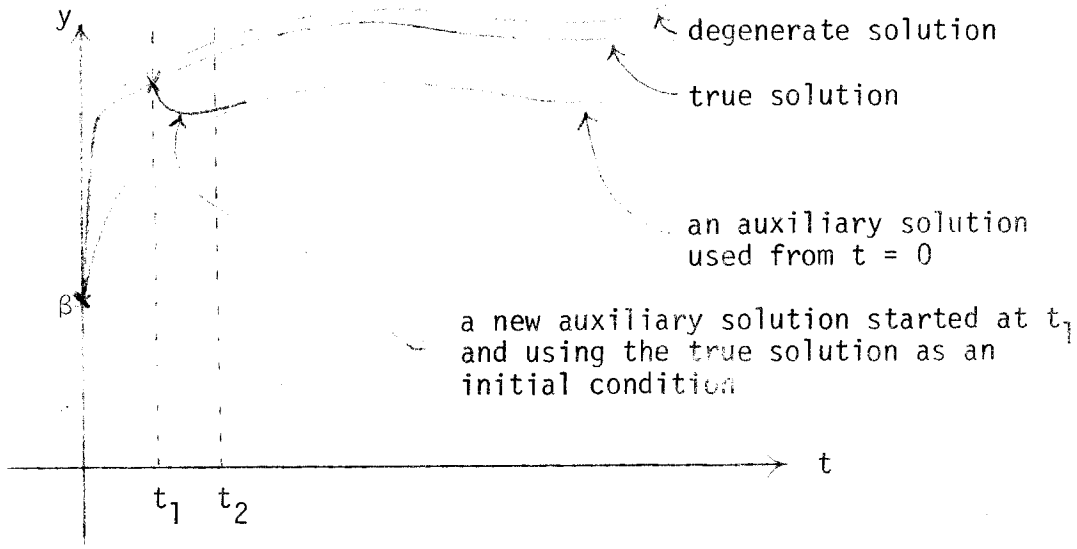

Diagram 4.2

For convenience we have used the same value of μ_i for each of the auxiliary parameters. The solution being started at t_1 using the true solution for a starting value experiences a boundary layer effect in the interval $t_1 \leq t \leq t_2$. This effect although much smaller than the original boundary layer effect still causes problems in the quantity $\frac{\partial^q z}{\partial \mu^q}$.

An illustration of this effect may be seen by looking at the system (2.0.1) but using the initial conditions

$$x\Big|_{t=0} = \alpha$$

$$y\Big|_{t=0} = \phi(\alpha,0).$$

We might expect no boundary layer effect by starting on the degenerate solution. However, a look at Vasileva's expansions (2.1.7) shows differently. Solving the equations (2.1.15) and (2.1.24) we find

$$\bar{x}_0(\tau) - \hat{x}_0(\tau) = 0$$

and

$$\bar{y}_0(\tau) - \hat{y}_0(\tau) = 0.$$

Looking at (2.1.29) we also find $\hat{x}_1(0) = 0$ and therefore looking at (2.1.16) and (2.1.26) we also have

$$\bar{x}_1(\tau) - \hat{x}_1(\tau) = 0.$$

However, from (2.1.18a) we have

$$\tilde{y}_1(0) = \hat{y}_1(0) = g_y^{-1}(\alpha,\phi(\alpha,0),0)\ \frac{d\tilde{y}_o}{dt}(0)$$

and we have therefore

$$\bar{y}_1(\tau) - \hat{y}_1(\tau) \neq 0.$$

This means that

$$\bar{x}_i(\tau) - \hat{x}_i(\tau) \neq 0$$

and

$$\bar{y}_i(\tau) - \hat{y}_i(\tau) \neq 0, \quad i = 2,3,4,\ldots$$

and hence we have a boundary layer effect.

These last statements also show that in order to pick a set of auxiliary parameters, it is necessary to integrate a set of auxiliary parameters through the boundary layer instead of perhaps trying to avoid the boundary layer by starting on the degenerate solution for test purposes.

CHAPTER V

NUMERICAL RESULTS OF THE INTERPOLATION SCHEME

1. Some Sample Problems

In this chapter we present some numerical results showing various features of using the interpolation scheme. Some comparisons are made with a subroutine package due to Gear recognized as a very efficient routine for handling stiff systems. The following sample initial value problems are used.

Problem 5.1 $m = 1, n = 1$

$$\frac{dx}{dt} = y, \quad x(0) = 1$$

$$\mu\frac{dy}{dt} = x-y, \quad y(0) = 0.$$

The stable root $\phi(x,t)$ is given by

$$\phi(x,t) = x.$$

The degenerate solution is given by

$$x_0(t) = e^t$$
$$y_0(t) = e^t$$

and we see that this linear system does not have an asymptotically stable null solution. The true solution is easily obtained for this example and is used for comparison purposes.

Problem 5.2 $m = 1, n = 1$

$$\frac{dx}{dt} = y, \qquad x(0) = 1$$

$$\mu\frac{dy}{dt} = x^2 - y^2, \quad y(0) = 0.$$

The stable root $\phi(x,t)$ for these initial conditions is given by

$$\phi(x,t) = x.$$

Again, the degenerate solution is given by

$$x_o(t) = e^t$$
$$y_o(t) = e^t$$

and we see that the solution to this system with a quadratic function $g(x,y,t)$ grows exponentially also. This example is taken from Wasow [16], p.275.

Problem 5.3 $m = 1, n = 2$

$$\frac{dx}{dt} = -(x-3y^{(1)}-y^{(2)}-1)(-x+y^{(1)}+y^{(2)}+2), \quad x(0) = 2$$

$$\mu\frac{dy^{(1)}}{dt} = (x+3y^{(1)}+y^{(2)})(x-2y^{(1)}-y^{(2)}), \quad y^{(1)}(0) = 1$$

$$\mu\frac{dy^{(2)}}{dt} = (2x-y^{(1)}-y^{(2)})(x+3y^{(1)}+2y^{(2)}), \quad y^{(2)}(0) = 1.$$

The stable root $\phi(x,t)$ for these initial conditions is given by

$$\phi(x,t) = \begin{pmatrix} -x \\ 3x \end{pmatrix}.$$

The degenerate solution is stable at the point

$$x_o = 1.$$

Problem 5.4 $m = 3, n = 2$

$$\frac{dx^{(1)}}{dt} = -2x^{(1)} - x^{(2)} + x^{(3)} + 4y^{(1)} - y^{(5)}, \quad x^{(1)}(0) = 1$$

$$\frac{dx^{(2)}}{dt} = x^{(1)} - x^{(2)} - 2x^{(3)} + y^{(1)} + 3y^{(2)}, \quad x^{(2)}(0) = -1$$

$$\frac{dx^{(3)}}{dt} = 5x^{(1)} - x^{(2)} - 3x^{(3)} - y^{(1)} + 5y^{(2)}, \quad x^{(3)}(0) = 2$$

$$\mu\frac{dy^{(1)}}{dt} = -2x^{(1)} + x^{(2)} - 8x^{(3)} - y^{(1)} - 2y^{(2)}, \quad y^{(1)}(0) = -2$$

$$\mu\frac{dy^{(2)}}{dt} = x^{(1)} - 2x^{(2)} + 3x^{(3)} + y^{(1)} - 4y^{(2)}, \quad y^{(2)}(0) = 3.$$

The stable root $\phi(x,t)$ is given by

$$\phi(x) = \begin{pmatrix} -5/3x^{(1)} + 4/3x^{(2)} - 19/3x^{(3)} \\ \\ -1/6x^{(1)} - 1/6x^{(2)} - 5/6x^{(3)} \end{pmatrix}.$$

The eigenvalues of the matrix $\frac{\partial g}{\partial y}$ are -2 and -3. The system has an asymptotically stable null solution.

Problem 5.5 $m = 7, n = 3$

$$\frac{dx^{(1)}}{dt} = y^{(1)}, \quad x^{(1)}(0) = 1$$

$$\frac{dx^{(2)}}{dt} = x^{(1)} + y^{(3)}, \quad x^{(2)}(0) = 1$$

$$\frac{dx^{(3)}}{dt} = x^{(3)} - y^{(2)}, \quad x^{(3)}(0) = 1$$

$$\frac{dx^{(4)}}{dt} = x^{(2)} + x^{(3)} + y^{(1)}, \quad x^{(4)}(0) = 1$$

$$\frac{dx^{(5)}}{dt} = -x^{(5)} + x^{(7)}, \quad x^{(5)}(0) = 1$$

$$\frac{dx^{(6)}}{dt} = x^{(3)} + x^{(6)}, \quad x^{(6)}(0) = 1$$

$$\frac{dx^{(7)}}{dt} = x^{(2)} - x^{(7)} + y^{(1)}, \quad x^{(7)}(0) = 1$$

$$\mu\frac{dy^{(1)}}{dt} = x^{(1)^2} - y^{(1)^2}, \quad y^{(1)}(0) = 0$$

$$\mu\frac{dy^{(2)}}{dt} = x^{(1)} + x^{(3)} + x^{(5)} + 4y^{(2)} - 3y^{(3)} + t, \quad y^{(2)}(0) = 1$$

$$\mu\frac{dy^{(3)}}{dt} = x^{(2)} + x^{(4)} + x^{(7)} + 10y^{(2)} - 7y^{(3)} - t^2, \quad y^{(3)}(0) = 1.$$

The stable function $\phi(x,t)$ for these initial conditions is given by

$$\phi(x) = \begin{pmatrix} x^{(1)} \\ 7/2(x^{(1)}+x^{(3)}+x^{(5)}+t)-3/2(x^{(2)}+x^{(4)}+x^{(7)}-t^2) \\ 5(x^{(1)}+x^{(3)}+x^{(5)}+t)-2(x^{(2)}+x^{(4)}+x^{(7)}-t^2) \end{pmatrix}.$$

2. The Results

The results for these problems are given in Tables 5.1 to 5.22. The columns headed "error in" are absolute errors. For Problem 5.1 the interpolation scheme is compared to the true solution. For Problems 5.2 to 5.5 the interpolation scheme is compared to results obtained by using a subroutine package due to Gear. The column headed "max error expected" is the value of ε used in the interpolation scheme unless ε is not satisfied. In this case an estimate of the relative error incurred is given.

The columns headed CPU_G are the times in milliseconds taken to integrate the systems from the previous value of t to the current value of t using Gear's subroutine. The columns headed CPU_I are the corresponding times in milliseconds taken by the interpolation scheme. Because of the difference in the error criteria for the two methods it should be remembered that we used an error tolerance in Gear's subroutine small enough so that ε was satisfied for the whole range of t. It should also be remembered that the timing function used may be in error by as much as 10%.

Tables 5.1 to 5.5 give some numerical results for Problem 5.1. Table 5.1 shows results for values of $t = \frac{1}{16}, \frac{2}{16}, \frac{3}{16}, \ldots, 1$. As indicated the auxiliary parameters were selected by the subroutine. In Table 5.2 the interpolation scheme selected small values of u_i for the first value of t which was small. We then had the scheme select a new set of auxiliary parameters for the larger values of t using results obtained at the first value of t. Tables 5.3, 5.4 and 5.5 show results for large values of t. The maximum error expected is consistent with

the results since the errors given for x and y are absolute errors and ε is a relative error.

Tables 5.6 to 5.10 give results for Problem 5.2. In Table 5.6 the parameters were selected by the subroutine. The results compare favourably with Gear's method. In Table 5.7 a smaller set of auxiliary parameters were selected by the user and produced worse results. Tables 5.8 and 5.9 show that with larger values of μ_1, μ_2 and μ_3 than in Table 5.6 we get better results. However it was found that with larger values of μ_i than this, the error tolerance ε was not satisfied. In Table 5.10 we have used a more relaxed error tolerance ε and have allowed the subroutine to select the parameters. We note that the interpolation scheme compares well in this case.

Tables 5.11 to 5.17 show results for Problem 5.3. In Tables 5.11 and 5.12 we have let the subroutine select the parameters. When the first value of t is larger a more efficient selection of auxiliary parameters occurs when applied at larger values of t(t=1 for example). Table 5.13 shows results in which a smaller selection of auxiliary parameters than Table 5.12 was chosen by the user producing more inefficient times. Tables 5.14, 5.15, 5.16 and 5.17 show results in which increasingly larger selections of auxiliary parameters than in Table 5.12 were chosen by the user producing increasingly better times. Notice in Table 5.17 that ε was not achieved at t = .0625 and an estimate was given of the actual relative error. However, we notice that ε was achieved at larger values of t and the subroutine predicted this.

Tables 5.18 to 5.21 give results for Problem 5.4. In Table 5.18

the auxiliary parameters were selected by the sub-routine. In Table 5.19 a smaller selection of parameters selected by the user produced more inefficient times. Table 5.20 shows that a larger selection of parameters produces more efficient times. Notice however that ε was not achieved at $t = .2500$ and $t = .3125$ although for larger t the error tolerance was once again achieved. In Table 5.21 a more relaxed error tolerance was used.

Table 5.22 shows results for Problem 5.5. The error tolerance was not satisfied at $t = .0625$ but was satisfied at larger t. This was successfully predicted by the subroutine.

Problem 5.1 with $\mu=10^{-6}$

t	error in		max error expected
	x	y	
0.0625	0.86D-13	0.50D-12	0.10D-09
0.1250	0.81D-14	-0.95D-11	0.10D-09
0.1875	-0.11D-12	0.26D-11	0.10D-09
0.2500	-0.22D-12	-0.12D-12	0.10D-09
0.3125	-0.19D-12	0.39D-12	0.10D-09
0.3750	-0.47D-12	-0.25D-12	0.10D-09
0.4375	-0.46D-12	-0.25D-12	0.10D-09
0.5000	-0.61D-12	-0.31D-12	0.10D-09
0.5625	-0.11D-11	-0.89D-12	0.10D-09
0.6250	-0.17D-11	-0.14D-11	0.10D-09
0.6875	-0.21D-11	-0.18D-11	0.10D-09
0.7500	-0.31D-11	-0.28D-11	0.10D-09
0.8125	-0.36D-11	-0.33D-11	0.10D-09
0.8750	-0.45D-11	-0.59D-11	0.10D-09
0.9375	-0.55D-11	-0.51D-11	0.10D-09
1.0000	-0.66D-11	-0.62D-11	0.10D-09

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected for the user by the subroutine were
.00107, .00142, .00190, .00253, .00337.

Table 5.1

Problem 5.1 with $\mu=10^{-6}$

t	error in		max error expected
	x	y	
0.0010	0.14D-14	0.28D-14	0.10D-09
0.1010	-0.[illegible]6D-13	0.79D-12	0.10D-09
0.2010	-0.28D-12	-0.13D-12	0.10D-09
0.3010	-0.90D-12	-0.92D-12	0.10D-09
0.4010	-0.22D-11	0.25D-11	0.10D-09
0.5010	-0.25D-11	-0.25D-11	0.10D-09
0.6010	-0.37D-11	-0.49D-11	0.10D-09
0.7010	-0.56D-11	0.72D-11	0.10D-09
0.8010	-0.63D-11	-0.63D-11	0.10D-09
0.9010	-0.81D-11	-0.80D-11	0.10D-09
1.0010	-0.11D-10	0.30D-10	0.10D-09
1.1010	-0.12D-10	-0.12D-10	0.10D-09
1.2010	-0.15D-10	-0.15D-10	0.10D-09
1.3010	-0.20D-10	0.12D-09	0.10D-09
1.4010	-0.22D-10	-0.22D-10	0.10D-09
1.5010	-0.27D-10	-0.26D-10	0.10D-09

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected for the user at the first value of t ranged from .00001 to .00005.
The auxiliary parameters selected for each t thereafter were
.00262,.00349,.00466,.00621,.00828 .

Table 5.2

Problem 5.1 with $\mu=10^{-6}$

t	error in		max error expected
	x	y	
1.0000	0.67D-11	0.52D-11	0.10D-09
2.0000	0.80D-10	0.79D-10	0.10D-09
3.0000	0.36D-09	0.36D-09	0.10D-09
4.0000	0.13D-08	0.13D-08	0.10D-09
5.0000	0.45D-08	0.47D-08	0.10D-09
6.0000	0.14D-07	0.14D-07	0.10D-09
7.0000	0.44D-07	0.44D-07	0.10D-09
8.0000	0.14D-06	0.14D-06	0.10D-09
9.0000	0.43D-06	0.45D-06	0.10D-09
10.0000	0.13D-05	0.13D-05	0.10D-09
11.0000	0.38D-05	0.38D-05	0.10D-09
12.0000	0.11D-04	0.11D-04	0.10D-09
13.0000	0.34D-04	0.34D-04	0.10D-09
14.0000	0.99D-04	0.10D-03	0.12D-09
15.0000	0.29D-03	0.29D-03	0.10D-09
16.0000	0.84D-03	0.85D-03	0.21D-09

The error tolerance $\varepsilon=10^{-10}$
The auxiliary parameters selected for the user by the subroutine were
.00422, .00563, .00750, .01000, .01333.

Table 5.3

Problem 5.1 with $\mu=10^{-6}$

t	error in x	error in y	max error expected
1.0000	-0.87D-12	-0.86D-12	0.11D-11
2.0000	-0.50D-11	-0.50D-11	0.21D-11
3.0000	-0.25D-10	-0.25D-10	0.32D-11
4.0000	-0.94D-10	-0.93D-10	0.43D-11
5.0000	-0.32D-09	-0.32D-09	0.62D-11
6.0000	-0.11D-08	-0.10D-08	0.82D-11
7.0000	-0.33D-08	-0.33D-08	0.11D-10
8.0000	-0.10D-07	-0.10D-07	0.14D-10
9.0000	-0.31D-07	-0.30D-07	0.17D-10
10.0000	-0.93D-07	-0.91D-07	0.21D-10
11.0000	-0.27D-06	-0.27D-06	0.26D-10
12.0000	-0.80D-06	-0.77D-06	0.33D-10
13.0000	-0.23D-05	-0.22D-05	0.40D-10
14.0000	-0.65D-05	-0.63D-05	0.48D-10
15.0000	-0.19D-04	-0.18D-04	0.58D-10
16.0000	-0.52D-04	-0.49D-04	0.70D-10

The error tolerance $\varepsilon=10^{-12}$
The auxiliary parameters selected for the user by the subroutine were
.00316, .00422, .00563, .00407, .01000.

Table 5.4

Problem 5.1 with $\mu=10^{-6}$

t	error in		max error expected
	x	y	
1.0000	0.62D-13	0.62D-13	0.10D-11
2.0000	0.35D-12	0.35D-12	0.10D-11
3.0000	0.14D-11	0.16D-11	0.10D-11
4.0000	0.58D-11	0.62D-11	0.10D-11
5.0000	0.17D-10	0.17D-10	0.10D-11
6.0000	0.55D-10	0.55D-10	0.10D-11
7.0000	0.18D-09	0.19D-09	0.10D-11
8.0000	0.55D-09	0.55D-09	0.10D-11
9.0000	0.17D-08	0.17D-08	0.10D-11
10.0000	0.52D-08	0.54D-08	0.10D-11
11.0000	0.16D-07	0.17D-07	0.10D-11
12.0000	0.48D-07	0.48D-07	0.13D-11
13.0000	0.14D-06	0.15D-06	0.18D-11
14.0000	0.42D-06	0.43D-06	0.24D-11
15.0000	0.12D-05	0.13D-05	0.31D-11
16.0000	0.37D-05	0.38D-05	0.40D-11

The error tolerance $\varepsilon=10^{-12}$
The auxiliary parameters selected by the user were
.00100, .00200, .00300, .00400, .00500.

Table 5.5

Problem 5.2 with $u=10^{-6}$

t	error in x	error in y	max error expected	CPU_G	CPU_I
0.0625	-0.31D-12	-0.32D-12	0.10D-09	3730	2718
0.1250	-0.86D-12	-0.79D-12	0.10D-09	258	131
0.1875	-0.15D-11	-0.10D-11	0.10D-09	205	130
0.2500	-0.20D-11	-0.20D-11	0.10D-09	239	142
0.3125	-0.26D-11	-0.23D-11	0.10D-09	251	134
0.3750	-0.32D-11	-0.25D-11	0.10D-09	252	148
0.4375	-0.40D-11	-0.40D-11	0.10D-09	261	170
0.5000	-0.48D-11	-0.26D-11	0.10D-09	244	180
0.5625	-0.59D-11	-0.59D-11	0.10D-09	275	184
0.6250	-0.73D-11	-0.73D-11	0.10D-09	277	168
0.6875	-0.83D-11	-0.32D-11	0.10D-09	248	221
0.7500	-0.96D-11	-0.11D-10	0.10D-09	250	240
0.8125	-0.11D-10	-0.11D-10	0.10D-09	304	264
0.8750	-0.13D-10	-0.13D-10	0.10D-09	249	279
0.9375	-0.15D-10	-0.15D-10	0.10D-09	269	280
1.0000	-0.17D-10	-0.17D-10	0.10D-09	289	281

The error tolerance $\varepsilon=10^{-10}$
The auxiliary parameters selected for the user by the subroutine were
.00253, .00337, .00449, .00828, .01656.

Table 5.6

Problem 5.2 with $\mu=10^{-6}$

t	error in x	error in y	max error expected	CPU_G	CPU_I
0.0625	-0.34D-12	-0.74D-12	0.10D-09	3751	1560
0.1250	-0.89D-12	-0.12D-11	0.10D-09	259	170
0.1875	-0.15D-11	-0.41D-11	0.10D-09	265	165
0.2500	-0.21D-11	-0.21D-11	0.10D-09	240	174
0.3125	-0.26D-11	-0.26D-11	0.10D-09	253	161
0.3750	-0.32D-11	-0.24D-11	0.10D-09	256	160
0.4375	-0.40D-11	-0.41D-11	0.10D-09	265	195
0.5000	-0.49D-11	-0.48D-11	0.10D-09	245	211
0.5625	-0.60D-11	-0.57D-11	0.10D-09	275	214
0.6250	-0.74D-11	0.12D-10	0.10D-09	277	200
0.6875	-0.84D-11	0.40D-10	0.10D-09	248	230
0.7500	-0.96D-11	-0.97D-11	0.10D-09	249	266
0.8125	-0.11D-10	-0.11D-10	0.10D-09	308	309
0.8750	-0.13D-10	-0.14D-10	0.10D-09	250	321
0.9375	-0.15D-10	-0.15D-10	0.10D-09	276	360
1.0000	-0.17D-10	-0.22D-10	0.10D-09	290	327

The error tolerance $\varepsilon=10^{-10}$
The auxiliary parameters selected by the user were
.00200, .00300, .00400, .00500, .00600.

Table 5.7

Problem 5.2 with $\mu=10^{-6}$

t	error in x	error in y	max error expected	CPU_G	CPU_I
0.0625	-0.27D-12	0.11D-12	0.10D-09	3734	1458
0.1250	-0.81D-12	-0.66D-12	0.10D-09	259	118
0.1875	-0.14D-11	-0.14D-11	0.10D-09	266	123
0.2500	-0.20D-11	-0.20D-11	0.10D-09	239	121
0.3125	-0.25D-11	-0.25D-11	0.10D-09	251	124
0.3750	-0.31D-11	-0.32D-11	0.10D-09	252	121
0.4375	-0.39D-11	-0.39D-11	0.10D-09	263	157
0.5000	-0.48D-11	-0.49D-11	0.10D-09	247	131
0.5625	-0.59D-11	-0.62D-11	0.10D-09	277	144
0.6250	-0.72D-11	-0.80D-11	0.10D-09	277	144
0.6875	-0.83D-11	-0.10D-10	0.10D-09	249	162
0.7500	-0.95D-11	-0.93D-11	0.10D-09	249	134
0.8125	-0.11D-10	-0.11D-10	0.10D-09	304	202
0.8750	-0.13D-10	-0.11D-10	0.10D-09	249	223
0.9375	-0.15D-10	-0.15D-10	0.10D-09	269	252
1.0000	-0.17D-10	-0.17D-10	0.10D-09	288	260

The error tolerance $\varepsilon=10^{-10}$
The auxiliary parameters selected by the user were
.00300, .00400, .00500, .00600, .00700.

Table 5.8

Problem 5.2 with $\mu=10^{-6}$

t	error in x	error in y	max error expected	CPU_G	CPU_I
0.0625	0.21D-12	-0.40D-10	0.10D-09	3737	2116
0.1250	-0.45D-12	-0.35D-12	0.10D-09	258	96
0.1875	-0.11D-11	-0.11D-11	0.10D-09	265	85
0.2500	-0.16D-11	-0.17D-11	0.10D-09	238	87
0.3125	-0.21D-11	-0.23D-11	0.10D-09	251	85
0.3750	-0.26D-11	-0.34D-11	0.10D-09	252	86
0.4375	-0.34D-11	-0.31D-11	0.10D-09	260	97
0.5000	-0.42D-11	-0.14D-11	0.10D-09	245	96
0.5625	-0.53D-11	-0.26D-11	0.10D-09	275	108
0.6250	-0.66D-11	-0.64D-11	0.10D-09	277	114
0.6875	-0.76D-11	-0.76D-11	0.10D-09	248	122
0.7500	-0.88D-11	-0.85D-11	0.10D-09	248	137
0.8125	-0.11D-10	-0.10D-10	0.10D-09	304	159
0.8750	-0.12D-10	-0.66D-11	0.10D-09	249	164
0.9375	-0.14D-10	-0.14D-10	0.10D-09	271	177
1.0000	-0.16D-10	-0.16D-10	0.10D-09	291	168

The error tolerance $\varepsilon=10^{-10}$
The auxiliary parameters selected by the user were
.00500, .00600, .00700, .00800, .00900.

Table 5.9

Problem 5.2 with $\mu=10^{-6}$

t	error in x	error in y	max error expected	CPU_G	CPU_I
0.0625	0.61D-08	-0.13D-05	0.27D-06	1909	1596
0.1250	0.42D-10	0.92D-09	0.15D-07	145	181
0.1875	-0.73D-09	-0.12D-08	0.10D-07	176	59
0.2500	-0.10D-08	-0.10D-08	0.10D-07	156	50
0.3125	-0.14D-08	-0.14D-08	0.10D-07	159	33
0.3750	-0.19D-08	-0.15D-08	0.10D-07	172	44
0.4375	-0.21D-08	-0.21D-08	0.10D-07	168	44
0.5000	-0.27D-08	-0.23D-08	0.10D-07	161	30
0.5625	-0.33D-08	-0.32D-08	0.10D-07	163	58
0.6250	-0.39D-08	-0 22D-08	0.10D-07	150	34
0.6875	-0.48D-08	-0.48D-08	0.10D-07	173	58
0.7500	-0.53D-08	-0.53D-08	0.10D-07	167	45
0.8125	-0.64D-08	-0.64D-08	0.10D-07	174	41
0.8750	-0.71D-08	-0.71D-08	0.10D-07	144	81
0.9375	-0.79D-08	-0.79D-08	0.10D-07	137	56
1.0000	-0.88D-08	-0.88D-08	0.10D-07	137	67

The error tolerance $\varepsilon=10^{-8}$.
The auxiliary parameters selected for the user by the subroutine were
.01147, .02294, 03441, .04588, .05735.

Table 5.10

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.5000	0.20D-10	-0.12D-10	0.50D-10	0.10D-09	12634	13421
0.5625	-0.14D-11	0.88D-12	-0.39D-11	0.10D-09	982	2857
0.6250	-0.11D-11	0.66D-12	-0.59D-11	0.10D-09	1032	679
0.6875	-0.70D-12	0.95D-12	-0.44D-11	0.10D-09	995	656
0.7500	-0.38D-12	0.26D-11	0.31D-11	0.10D-09	855	597
0.8125	0.63D-11	-0.11D-10	0.25D-10	0.10D-09	851	525
0.8750	0.43D-11	-0.83D-11	0.13D-10	0.10D-09	852	484
0.9375	0.26D-11	-0.63D-11	0.11D-10	0.10D-09	938	436
1.0000	0.13D-11	-0.39D-11	0.85D-11	0.10D-09	911	398
1.0625	0.99D-12	-0.34D-11	0.54D-11	0.10D-09	855	347
1.1250	0.79D-12	-0.26D-11	0.42D-11	0.10D-09	888	337
1.1875	0.10D-10	0.36D-11	0.12D-10	0.10D-09	803	290
1.2500	0.12D-10	-0.27D-11	0.18D-10	0.10D-09	916	314
1.3125	0.13D-10	-0.51D-11	0.28D-10	0.10D-09	712	334
1.3750	0.61D-12	-0.18D-11	-0.11D-11	0.10D-09	730	496
1.4375	0.63D-12	0.51D-12	0.61D-11	0.10D-09	674	398

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected for the user by the subroutine were
.00316, .00422, .00563, .00725, .01000.

Table 5.11

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.0625	-0.65D-11	0.11D-10	-0.25D-10	0.10D-09	10715	7591
0.1250	-0.29D-11	0.64D-11	-0.13D-10	0.10D-09	1227	1689
0.1875	-0.95D-12	0.36D-11	-0.56D-11	0.10D-09	1073	1510
0.2500	0.11D-11	0.78D-12	0.13D-11	0.10D-09	1057	1338
0.3125	-0.66D-11	-0.14D-10	0.58D-11	0.10D-09	1093	1133
0.3750	0.26D-11	-0.18D-11	0.64D-11	0.10D-09	1009	1452
0.4375	0.26D-11	-0.21D-11	0.74D-11	0.10D-09	1061	1227
0.5000	0.26D-11	-0.26D-11	0.87D-11	0.10D-09	941	1140
0.5625	0.24D-11	-0.24D-11	0.69D-11	0.10D-09	1032	1104
0.6250	0.21D-11	-0.22D-11	0.63D-11	0.10D-09	809	1005
0.6875	0.19D-11	-0.19D-11	0.59D-11	0.10D-09	880	959
0.7500	0.16D-11	-0.18D-11	0.47D-11	0.10D-09	940	858
0.8125	0.14D-11	-0.65D-12	0.71D-11	0.10D-09	754	806
0.8750	0.12D-11	0.32D-13	0.82D-11	0.10D-09	885	741
0.9375	0.96D-12	-0.73D-12	0.42D-11	0.10D-09	892	718
1.0000	0.96D-13	0.20D-11	-0.25D-11	0.10D-09	880	810

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected for the user by the subroutine were
.00142, .00190, .00253, .00337, .00449.

Table 5.12

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.0625	-0.45D-11	0.76D-11	-0.17D-10	0.10D-09	10953	6557
0.1250	-0.20D-11	0.42D-11	-0.93D-11	0.10D-09	1265	1837
0.1875	-0.54D-12	0.23D-11	-0.41D-11	0.10D-09	1091	1862
0.2500	0.12D-11	-0.60D-12	-0.29D-12	0.10D-09	1072	1522
0.3125	-0.39D-11	-0.99D-11	0.57D-11	0.10D-09	1108	1279
0.3750	0.24D-11	-0.18D-11	0.64D-11	0.10D-09	1027	1595
0.4375	0.24D-11	-0.22D-11	0.67D-11	0.10D-09	1073	1381
0.5000	0.24D-11	-0.27D-11	0.54D-11	0.10D-09	943	1261
0.5625	0.22D-11	-0.23D-11	0.63D-11	0.10D-09	1033	1241
0.6250	0.19D-11	-0.18D-11	0.64D-11	0.10D-09	804	1149
0.6875	0.17D-11	-0.18D-11	0.54D-11	0.10D-09	885	1078
0.7500	0.15D-11	-0.16D-11	0.46D-11	0.10D-09	950	1309
0.8125	0.13D-11	-0.70D-11	-0.14D-10	0.10D-09	764	1020
0.8750	0.12D-11	-0.12D-11	0.37D-11	0.10D-09	896	1003
0.9375	-0.12D-12	0.15D-11	-0.32D-11	0.10D-09	902	933
1.0000	0.30D-12	0.11D-11	-0.11D-11	0.10D-09	891	919

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00100, .00180, .00260, .00350, .00400.

Table 5.13

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.0625	0.17D-11	0.27D-12	0.54D-11	0.10D-09	10805	6964
0.1250	0.11D-11	0.22D-11	0.15D-10	0.10D-09	1232	1747
0.1875	-0.29D-11	0.98D-11	-0.13D-10	0.10D-09	1072	1207
0.2500	0.66D-12	0.42D-11	-0.18D-11	0.10D-09	1054	1085
0.3125	0.25D-11	0.85D-12	0.47D-11	0.10D-09	1091	1084
0.3750	0.33D-11	-0.11D-11	0.81D-11	0.10D-09	1036	1091
0.4375	0.35D-11	-0.27D-11	0.74D-11	0.10D-09	1049	1060
0.5000	0.35D-11	-0.31D-11	0.87D-11	0.10D-09	921	1008
0.5625	0.31D-11	0.68D-11	0.42D-10	0.10D-09	1049	952
0.6250	0.27D-11	-0.43D-11	0.35D-11	0.10D-09	798	905
0.6875	0.24D-11	-0.28D-11	0.70D-11	0.10D-09	878	806
0.7500	0.19D-11	-0.22D-11	0.67D-11	0.10D-09	938	720
0.8125	0.16D-11	-0.23D-11	0.42D-11	0.10D-09	759	692
0.8750	0.13D-11	-0.97D-12	0.68D-11	0.10D-09	892	631
0.9375	0.10D-11	0.95D-12	0.11D-10	0.10D-09	890	583
1.0000	0.78D-12	-0.61D-13	0.59D-11	0.10D-09	876	593

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00200, .00250, .00300, .00350, .00400.

Table 5.14

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.0625	0.24D-11	0.15D-10	-0.64D-11	0.10D-09	10803	6756
0.1250	0.17D-11	-0.38D-11	-0.36D-11	0.10D-09	1240	1566
0.1875	-0.49D-11	0.14D-10	-0.29D-10	0.10D-09	1084	1158
0.2500	0.17D-12	0.64D-11	-0.91D-11	0.10D-09	1053	973
0.3125	0.28D-11	0.18D-11	0.18D-11	0.10D-09	1095	995
0.3750	0.40D-11	-0.10D-11	0.78D-11	0.10D-09	1034	972
0.4375	0.44D-11	-0.26D-11	0.11D-10	0.10D-09	1071	967
0.5000	0.43D-11	-0.49D-11	0.63D-11	0.10D-09	947	949
0.5625	0.39D-11	-0.37D-11	0.11D-10	0.10D-09	1051	902
0.6250	0.34D-11	-0.51D-11	0.57D-11	0.10D-09	803	839
0.6875	0.28D-11	-0.11D-11	0.17D-10	0.10D-09	903	777
0.7500	0.24D-11	-0.69D-11	-0.58D-11	0.10D-09	952	683
0.8125	0.19D-11	-0.39D-12	0.13D-10	0.10D-09	775	621
0.8750	0.15D-11	-0.22D-11	0.48D-11	0.10D-09	902	587
0.9375	0.11D-11	-0.49D-12	0.76D-11	0.10D-09	900	537
1.0000	0.84D-12	-0.25D-11	-0.12D-11	0.10D-09	887	541

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00230, .00280, .00330, .00380, .00430.

Table 5.15

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_I
0.0625	-0.52D-11	0.43D-09	-0.58D-09	0.10D-09	10756	6357
0.1250	0.43D-11	-0.49D-11	0.40D-10	0.10D-09	1230	1545
0.1875	0.23D-11	-0.75D-11	-0.11D-10	0.10D-09	1061	1360
0.2500	-0.19D-11	0.19D-10	-0.29D-10	0.10D-09	1053	362
0.3125	0.40D-11	0.76D-11	-0.38D-11	0.10D-09	1080	798
0.3750	0.68D-11	0.73D-12	0.10D-10	0.10D-09	1013	792
0.4375	0.78D-11	-0.33D-11	0.17D-10	0.10D-09	1054	793
0.5000	0.77D-11	-0.56D-11	0.20D-10	0.10D-09	957	821
0.5625	0.70D-11	-0.64D-11	0.20D-10	0.10D-09	1053	791
0.6250	0.59D-11	-0.62D-11	0.18D-10	0.10D-09	802	759
0.6875	0.48D-11	-0.33D-11	0.25D-10	0.10D-09	880	702
0.7500	0.38D-11	-0.70D-11	0.58D-11	0.10D-09	946	615
0.8125	0.30D-11	-0.12D-10	-0.15D-10	0.10D-09	764	559
0.8750	0.21D-11	-0.12D-11	0.16D-10	0.10D-09	894	503
0.9375	0.14D-11	-0.83D-14	0.14D-10	0.10D-09	900	449
1.0000	0.92D-12	0.23D-12	0.11D-10	0.10D-09	878	444

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00300, .00350, .00400, .00450, .00500.

Table 5.16

Problem 5.3 with $\mu=10^{-6}$

t	error in x	error in $y^{(1)}$	error in $y^{(2)}$	max error expected	CPU_G	CPU_T
0.0625	-0.12D-09	-0.16D-08	0.17D-08	0.63D-09	10820	5909
0.1250	0.14D-10	-0.30D-10	0.42D-10	0.10D-09	1278	1519
0.1875	0.72D-11	-0.50D-11	0.10D-09	0.10D-09	1097	1366
0.2500	0.38D-11	-0.65D-11	0.11D-11	0.10D-09	1064	1096
0.3125	0.74D-11	0.21D-10	-0.39D-10	0.10D-09	1104	716
0.3750	0.15D-10	0.36D-11	0.68D-11	0.10D-09	1037	657
0.4375	0.65D-12	-0.14D-10	-0.39D-10	0.10D-09	1063	1134
0.5000	0.47D-12	-0.92D-11	-0.27D-10	0.10D-09	949	751
0.5625	0.41D-12	-0.58D-11	-0.18D-10	0.10D-09	1042	756
0.6250	0.41D-12	-0.37D-11	-0.12D-10	0.10D-09	824	773
0.6875	0.53D-12	-0.20D-11	-0.79D-11	0.10D-09	890	754
0.7500	0.22D-12	0.26D-10	0.97D-10	0.10D-09	953	697
0.8125	0.60D-11	-0.12D-10	0.14D-10	0.10D-09	766	516
0.8750	0.40D-11	-0.35D-11	0.30D-10	0.10D-09	903	451
0.9375	0.30D-11	-0.33D-10	-0.80D-10	0.10D-09	892	395
1.0000	0.12D-11	0.32D-11	0.31D-10	0.10D-09	893	364

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00400, .00450, .00500, .00550, .00600.

Table 5.17

Problem 5.4 with $\mu=10^{-6}$

t	error in $x^{(1)}$ / $y^{(1)}$	error in $x^{(2)}$ / $y^{(2)}$	error in $x^{(3)}$	max error expected	CPU_G	CPU_I
0.0625	-0.23D-10 -0.81D-10	-0.35D-11 -0.11D-10	0.18D-10	0.10D-09	8902	7091
0.1250	-0.44D-10 -0.94D-10	-0.14D-10 -0.21D-10	0.10D-10	0.10D-09	1227	1027
0.1875	-0.12D-09 0.13D-09	-0.55D-10 0.29D-10	-0.55D-12	0.10D-09	1232	1264
0.2500	-0.91D-10 0.33D-09	-0.57D-10 0.75D-10	-0.37D-10	0.10D-09	1158	1140
0.3125	-0.25D-11 0.19D-09	-0.13D-10 0.35D-10	-0.29D-10	0.10D-09	1250	1054
0.3750	0.43D-10 0.21D-09	0.25D-11 0.30D-10	-0.44D-10	0.10D-09	1195	1344
0.4375	0.64D-10 0.49D-10	0.22D-10 0.35D-11	-0.19D-10	0.10D-09	1106	1223
0.5000	0.70D-10 -0.76D-10	0.33D-10 -0.18D-10	0.97D-12	0.10D-09	1103	1127
0.5625	0.92D-11 -0.87D-10	0.92D-11 -0.10D-10	0.12D-10	0.10D-09	1040	1025
0.6250	0.94D-11 -0.16D-09	0.15D-10 -0.28D-10	0.26D-10	0.10D-09	1105	1296
0.6875	-0.26D-10 -0.91D-10	-0.36D-11 -0.12D-10	0.20D-10	0.10D-09	1066	1211
0.7500	-0.41D-10 -0.11D-10	-0.15D-10 0.12D-11	0.92D-11	0.10D-09	972	1178
0.8125	-0.24D-10 0.44D-10	-0.11D-10 0.96D-11	-0.20D-11	0.10D-09	998	1048
0.8750	-0.38D-10 0.11D-09	-0.21D-10 0.19D-10	-0.12D-10	0.10D-09	1001	1241
0.9375	-0.47D-11 0.11D-09	-0.81D-11 0.17D-10	-0.17D-10	0.10D-09	1079	1210
1.0000	0.10D-10 0.67D-10	0.38D-11 0.88D-11	-0.14D-10	0.10D-09	974	1179

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected for the user by the subroutine were .00142, .00190, .00253, .00337, .00449.

Table 5.18

Problem 5.4 with $\mu=10^{-6}$

t	error in $x^{(1)}$ / $y^{(1)}$	error in $x^{(2)}$ / $y^{(2)}$	error in $x^{(3)}$	max error expected	CPU_G	CPU_I
0.0625	-0.63D-12 -0.52D-11	0.15D-12 -0.62D-12	0.12D-11	0.10D-09	8792	7194
0.1250	-0.11D-10 -0.78D-10	-0.21D-11 -0.21D-10	0.75D-11	0.10D-09	1217	1236
0.1875	-0.30D-11 0.83D-11	-0.13D-11 0.63D-11	0.82D-11	0.10D-09	1217	1372
0.2500	-0.12D-12 0.17D-10	-0.11D-11 0.28D-11	-0.20D-11	0.10D-09	1144	1351
0.3125	0.47D-11 0.75D-10	-0.17D-11 0.15D-10	-0.98D-11	0.10D-09	1265	1265
0.3750	0.11D-10 0.14D-10	0.34D-11 0.81D-12	-0.48D-11	0.10D-09	1184	1596
0.4375	0.14D-10 -0.24D-10	0.73D-11 -0.55D-11	0.12D-11	0.10D-09	1138	1470
0.5000	0.77D-11 -0.47D-10	0.65D-11 -0.78D-11	0.68D-11	0.10D-09	1278	1329
0.5625	-0.64D-11 -0.65D-10	0.14D-11 -0.12D-10	0.10D-10	0.10D-09	1079	1318
0.6250	-0.21D-10 -0.23D-10	-0.62D-11 -0.18D-11	0.79D-11	0.10D-09	1030	1409
0.6875	-0.20D-10 0.20D-10	-0.11D-10 0.51D-11	0.16D-11	0.10D-09	1001	1382
0.7500	-0.20D-10 0.95D-10	-0.11D-10 0.10D-10	-0.57D-11	0.10D-09	989	1325
0.8125	-0.14D-10 0.04D-10	-0.85D-11 0.12D-10	-0.72D-11	0.10D-09	1006	1198
0.8750	-0.24D-11 0.79D-10	-0.53D-11 0.12D-10	-0.13D-10	0.10D-09	986	1365
0.9375	0.18D-10 0.55D-10	0.47D-11 0.67D-11	-0.12D-10	0.10D-09	944	1318
1.0000	0.33D-10 0.15D-11	0.14D-10 -0.29D-11	-0.58D-11	0.10D-09	894	1301

The error tolerance $\varepsilon=10^{-10}$. The auxiliary parameters selected by the user were .00100, .00200, .00300, .00400, .00500.

Table 5.19

Problem 5.4 with $\mu=10^{-6}$

t	error in $x^{(1)}$ / $y^{(1)}$	error in $x^{(2)}$ / $y^{(2)}$	error in $x^{(3)}$	max error expected	CPU_G	CPU_I
0.0625	-0.25D-10 0.60D-11	-0.80D-11 0.34D-10	0.96D-11	0.10D-09	8829	6636
0.1250	-0.28D-11 0.18D-10	-0.74D-12 0.56D-11	0.75D-12	0.10D-09	1225	1033
0.1875	0.15D-09 -0.13D-09	0.69D-10 -0.28D-10	-0.25D-12	0.13D-09	1224	992
0.2500	0.18D-09 -0.31D-09	0.63D-10 -0.56D-10	0.36D-10	0.13D-09	1139	1094
0.3125	0.18D-10 -0.27D-09	0.25D-10 -0.46D-10	0.42D-10	0.13D-09	1232	1133
0.3750	-0.24D-10 -0.10D-09	0.71D-12 -0.25D-10	0.29D-10	0.10D-09	1207	1147
0.4375	-0.35D-10 -0.71D-10	-0.93D-11 -0.10D-10	0.17D-10	0.10D-09	1152	1142
0.5000	-0.32D-10 -0.15D-10	-0.11D-10 -0.44D-12	0.83D-11	0.10D-09	1217	1141
0.5625	-0.27D-10 0.19D-10	-0.11D-10 0.60D-11	0.25D-11	0.10D-09	1068	1139
0.6250	-0.25D-10 0.32D-10	-0.11D-10 0.79D-11	-0.78D-13	0.10D-09	1032	1144
0.6875	-0.21D-10 0.50D-10	-0.11D-10 0.98D-11	-0.65D-11	0.10D-09	1036	1164
0.7500	-0.10D-10 0.65D-10	-0.79D-11 0.11D-10	-0.95D-11	0.10D-09	961	1124
0.8125	0.62D-11 0.51D-10	-0.37D-12 0.67D-11	-0.10D-11	0.10D-09	996	986
0.8750	0.21D-10 0.29D-10	0.74D-11 0.13D-11	-0.90D-11	0.10D-09	993	995
0.9375	0.31D-10 -0.21D-10	0.15D-10 -0.71D-11	-0.19D-11	0.10D-09	918	992
1.0000	0.28D-10 -0.65D-10	0.17D-10 -0.13D-10	0.66D-11	0.10D-09	902	1008

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00200, .00300, .00400, .00500, .00600.

Table 5.20

Problem 5.4 with $\mu=10^{-6}$

t	error in $x^{(1)}$ / $y^{(1)}$	error in $x^{(2)}$ / $y^{(2)}$	error in $x^{(3)}$	max error expected	CPU_G	CPU_I
0.0625	0.97D-09 0.90D-08	0.23D-09 0.28D-08	-0.49D-09	0.10D-07	4047	3235
0.1250	0.19D-08 0.21D-08	0.75D-09 0.63D-09	-0.32D-09	0.10D-07	898	551
0.1875	0.10D-08 0.47D-08	0.72D-10 0.58D-09	-0.10D-08	0.10D-07	865	985
0.2500	0.30D-08 0.15D-08	0.11D-08 -0.72D-10	-0.77D-09	0.10D-07	884	616
0.3125	0.32D-08 -0.54D-08	0.15D-08 -0.28D-08	0.83D-10	0.10D-07	912	575
0.3750	0.26D-08 -0.69D-08	0.15D-08 -0.13D-08	0.75D-09	0.10D-07	799	641
0.4375	-0.39D-10 -0.77D-08	0.52D-09 -0.12D-08	0.13D-08	0.10D-07	965	667
0.5000	-0.78D-09 -0.50D-08	-0.34D-10 -0.31D-08	0.80D-09	0.10D-07	838	383
0.5625	-0.16D-08 -0.16D-09	-0.61D-09 0.52D-10	0.38D-09	0.10D-07	761	592
0.6250	-0.13D-08 0.36D-08	-0.73D-09 0.62D-09	-0.41D-09	0.10D-07	860	662
0.6875	-0.11D-08 0.30D-08	-0.71D-09 -0.70D-09	-0.53D-09	0.10D-07	686	574
0.7500	-0.11D-08 0.29D-08	-0.65D-09 0.35D-09	-0.46D-09	0.10D-07	776	604
0.8125	0.88D-09 0.26D-08	0.22D-09 0.34D-09	-0.57D-09	0.10D-07	829	526
0.8750	0.10D-08 0.15D-08	0.35D-09 -0.14D-09	-0.47D-09	0.10D-07	648	431
0.9375	0.11D-08 0.87D-09	0.46D-09 0.11D-10	-0.33D-09	0.10D-07	639	584
1.0000	0.16D-08 0.11D-08	0.71D-09 0.18D-09	-0.34D-09	0.10D-07	639	413

The error tolerance $\varepsilon=10^{-8}$.
The auxiliary parameters selected for the user by the subroutine were .00253, .00337, .00449, .01018, .02035.

Table 5.21

Problem 5.5 with $\mu=10^{-6}$

t	error in z using L_∞ norm	max error expected	CPU_G	CPU_I
0.0625	0.38D-09	0.46D-09	20763	12618
0.1250	0.25D-10	0.10D-09	3124	5687
0.1875	0.11D-10	0.10D-09	3369	5614
0.2500	-0.15D-10	0.10D-09	3119	5617
0.3125	-0.38D-10	0.10D-09	3031	5017
0.3750	-0.29D-10	0.10D-09	3245	5007
0.4375	-0.14D-10	0.10D-09	3263	5008
0.5000	-0.14D-10	0.10D-09	3146	5015
0.5625	-0.20D-10	0.10D-09	3019	5017
0.6250	-0.30D-10	0.10D-09	2977	5024
0.6875	-0.37D-10	0.10D-09	3069	4849
0.7500	-0.43D-10	0.10D-09	2977	4394
0.8125	-0.53D-10	0.10D-09	3027	4131
0.8750	-0.56D-10	0.10D-09	3008	4040
0.9375	-0.26D-10	0.10D-09	2620	4146
1.0000	-0.32D-10	0.10D-09	2738	4132

The error tolerance $\varepsilon=10^{-10}$.
The auxiliary parameters selected by the user were
.00100, .00200, .00300, .00400, .00500.

Table 5.22

CHAPTER VI

ANOTHER ASYMPTOTIC METHOD

1. Motivation

With this method we try to replace the system

$$\frac{dx}{dt} = f(x,y,t) \tag{2.0.1}$$

$$\frac{dy}{dt} = \frac{g(x,y,t)}{\mu}$$

$$x(0,\mu) = \alpha, \quad y(0,\alpha) = \beta$$

by the system

$$\frac{dx}{dt} = f(x,y,t)$$

$$\frac{dy}{dt} = h(x,y,t,\mu) \tag{6.1.1}$$

for some function h. We have not specified any initial conditions on x and y in (6.1.1) yet. For the solution to (6.1.1) to be an approximation to (2.0.1) we would certainly need $h(x,y,t,\mu)$ to be a good approximation to $\frac{g(x,y,t)}{\mu}$. But to be able to obtain the solution to (6.1.1) without any stiffness problems we would hope that h depends continuously on μ for $0 \le \mu \le \bar{\mu}$. For t in the boundary layer this is too much to hope for but for t outside the boundary layer we have $|g(x,y,t)| = O(\mu)$. This implies that we should solve the full system (2.0.1) for t in the interval $0 \le t \le t_1$ obtaining $x(t_1,\mu)$ and $y(t_1,\mu)$ and then use these values as starting values for the system (6.1.1).

The approximation h for the quantity $\frac{g(x,y,t)}{\mu}$ we shall obtain will be given by

$$\frac{g(x,y,t)}{\mu} = h(x,y,t,\mu) + O(\mu^2) \tag{6.1.2}$$

and we will show that the resulting approximations we obtain by solving (6.1.1) will be second order approximations also.

2. Preliminary Results

In the derivation of the function $h(x,y,t,\mu)$ we will need the following lemmas.

Lemma 6.1 *We are given the square matrices* A, B, *and* C *such that*

$$A = B + \mu C + O(\mu^2) \tag{6.2.1}$$

where B *is nonsingular. Then we also have*

$$A^{-1} = B^{-1} - \mu B^{-1}CB^{-1} + O(\mu^2). \tag{6.2.2}$$

Proof If B is nonsingular then for μ small enough, A^{-1} exists. Using (6.2.1) and (6.2.2) we have

$$AA^{-1} = (B+\mu C+O(\mu^2))(B^{-1}-\mu B^{-1}CB^{-1}+O(\mu^2))$$

$$= I + \mu(CB^{-1}-BB^{-1}CB^{-1}) - \mu^2(CB^{-1}CB^{-1}) + O(\mu^2)$$

Simplifying we have

$$AA^{-1} = I + O(\mu^2)$$

which proves the lemma.

Lemma 6.2 *Given the square matrix* $A = A(t)$ *such that* $\frac{d}{dt}A(t)$ *exists and* A^{-1} *exists on some interval* $a \le t \le b$, *then*

$$\frac{d}{dt}(A^{-1}) = -A^{-1}\frac{d}{dt}(A)A^{-1}.$$

The proof is trivial.

Lemma 6.3 *Suppose* a, b, c *are* n *dimensional column vectors and* D *is an* $n \times n$ *matrix such that*

$$a = b + \mu c + \mu Da + O(\mu^2). \tag{6.2.3}$$

Then we also have

$$a = b + \mu c + \mu Db + O(\mu^2). \tag{6.2.4}$$

Proof From (6.2.3) we have

$$a = b + \mu c + \mu Da + O(\mu^2)$$

$$= b + \mu c + \mu D(b+\mu c+\mu Da) + O(\mu^2)$$

$$= b + \mu c + \mu Db + O(\mu^2)$$

which gives us (6.2.4).

Lemma 6.4 *Suppose we are given the* n *vector function* $g = g(x,y)$ *where* x *is an* m *vector and* y *is an* n *vector. Also suppose* a *and* b *are* n *vectors. If* g *has continuous second partial derivatives with respect to* y *then we have*

$$\frac{\partial^2 g}{\partial y^2}\, ab = \frac{\partial^2 g}{\partial y^2}\, ba.$$

Proof Using tensor notation we have $g = (g^i)$, $\frac{\partial g}{\partial y} = (\frac{\partial g^i}{\partial y^j}) = (g^i_j)$ and $\frac{\partial^2 g}{\partial y^2} = (\frac{\partial^2 g^i}{\partial y^j \partial y^k}) = (g^i_{jk})$ where in the last case $g^i_{jk} = g^i_{kj}$ since g has continuous second partials with respect to y. The i-th component of the product $\frac{\partial^2 g}{\partial y^2}\, ab$ is given by

$$c^i = g^i_{jk}\, a^k b^j$$

$$= g^i_{kj}\, b^j a^k$$

and this is the i-th component of $\frac{\partial^2 g}{\partial y^2}\, ba$.

Lemma 6.5 *Let* $A(t)$ *be an* $n \times n$ *continuous matrix for* $a \le t \le b$ *and let the real parts of its characteristic roots all be less than* -2κ *on* $a \le t \le b$ *for some* $\kappa > 0$. *Let* $\psi(t,t_0,\mu)$ *be the principal matrix solution of*

$$\frac{dX}{dt} = \frac{A(t)}{\mu} X$$

with initial condition at $t = t_0$ *such that* $a \le t_0 \le t \le b$. *Then there exists a constant* c *such*

$$|\psi(t,t_0,\mu)| \le c e^{-\kappa(t-t_0)/\mu}.$$

The proof of this lemma is given by Flatto and Levinson [4].

3. The Approximation

Expanding $g(x,y,t)$ in a Taylor series about $y = \phi(x,t)$ we have

$$g(x,y,t) = g(x,\phi(x,t),t) + g_y(x,\phi(x,t),t)(y-\phi(x,t)) + \frac{1}{2}g_{yy}(x,\phi(x,t),t)(y-\phi(x,t))^2 + \frac{1}{6}g_{yyy}(y-\phi(x,t))^3 \qquad (6.3.1)$$

where the y value of each component of g_{yyy} is in the interval spanned by $y(t,\mu)$ and $\phi(x,t)$. Let

$$w(t,\mu) = \frac{y(t,\mu)-\phi(x(t,\mu),t)}{\mu} . \qquad (6.3.2)$$

Using (6.3.1) and (6.3.2) and noting that $g(x,\phi(x,t),t) = 0$ we have

$$\frac{g(x,y,t)}{\mu} = g_y(x,\phi(x,t),t)w + \frac{1}{2}\mu g_{yy}(x,\phi(x,t),t)w^2 + \frac{1}{6}\mu^2 g_{yyy}w^3. \qquad (6.3.3)$$

Using (6.3.2) we may write

$$\frac{dw}{dt} = \frac{\frac{dy}{dt} - \phi_x f - \phi_t}{\mu}$$

$$= \frac{g(x,y,t)}{\mu^2} - \frac{\phi_x f + \phi_t}{\mu} . \qquad (6.3.4)$$

The last step follows using (2.0.1). Using Taylor's theorem and (6.3.2) we have

$$g(x,y,t) = \mu g_y w^3 \qquad (6.3.5)$$

where g_y is evaluated at $x(t,\mu)$ and t and

where the y value of each component of g_y is in the interval spanned by $y(t,\mu)$ and $\phi(x,t)$. Returning to (6.3.4) we have

$$\frac{dw}{dt} = \frac{g_y w}{\mu} - \frac{\phi_x f+\phi_t}{\mu} . \tag{6.3.6}$$

The solution to (6.3.6) may be expressed as

$$w(t,\mu) = \psi(t,t_o,\mu)w(t_o,\mu) - \psi(t,t_o,\mu) \int_{t_o}^{t} \psi^{-1}(s,t_o,\mu)[\frac{\phi_x f+\phi_t}{\mu}]ds \tag{6.3.7}$$

where $\psi(t,t_o,\mu)$ is the principal matrix solution to the system

$$\frac{d\psi}{dt} = \frac{g_y \psi}{\mu} \tag{6.3.8}$$

with initial condition at $t = t_o$. Note that from (6.3.5) g_y here may be considered a function of t.

We make the slightly more restrictive assumption than (2.1.11) that

$$Re[\lambda_i (\frac{\partial g}{\partial y})|_{y=\phi(x,t)}] < -2\kappa \quad \text{where } \kappa > 0.$$

Therefore, we may apply Lemma 6.5 since we have $\lim_{\mu\to 0} y(t,\mu) = \phi(\tilde{x}_o,t)$ and therefore for μ small enough we have

$$Re[\lambda_i (\frac{\partial g}{\partial y})] < -2\kappa \qquad \text{where } \frac{\partial g}{\partial y} \text{ is evaluated as in (6.3.5)}$$

This gives us

$$|\psi(t,t_o,\mu)| \le ce^{-\kappa(t-t_o)/\mu} . \tag{6.3.9}$$

We may therefore neglect the first term of (6.3.7) for $t > 2t_0$ say if we are interested in an asymptotic representation of $w(t,\mu)$. We therefore rewrite (6.3.7) as

$$w(t,\mu) = -\psi(t,t_0,\mu) \int_{t_0}^{t} \psi^{-1}(s,t_0,\mu)[\frac{\phi_x f+\phi_t}{\mu}]ds \ . \tag{6.3.10}$$

Now if ψ is the principal matrix solution to (6.3.8) then we also have as the adjoint equation

$$\frac{d\psi^{-1}}{dt} = -\psi^{-1} \frac{g_y}{\mu} \tag{6.3.11}$$

or rearranging (6.3.11) we have

$$\psi^{-1} = -\mu \frac{d\psi^{-1}}{dt} g_y^{-1} \tag{6.3.12}$$

where g_y^{-1} is evaluated at the same arguments as g_y is in (6.3.5). Substituting (6.3.12) in (6.3.10) we obtain

$$w(t,\mu) = \psi(t,t_0,\mu) \int_{t_0}^{t} \frac{d\psi^{-1}}{ds} g_y^{-1}(\phi_x f+\phi_t)ds. \tag{6.3.13}$$

Integrating (6.3.13) by parts we obtain

$$\begin{aligned} w(t,\mu) &= \psi(t,t_0,\mu)[\psi^{-1}(s,t_0,\mu)g_y^{-1}(\phi_x f+\phi_t)]_{s=t_0}^{s=t} \\ &\quad - \psi(t,t_0,\mu) \int_{t_0}^{t} \psi^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'ds \\ &= g_y^{-1}(\phi_x f-\phi_t) - \psi(t,t_0,\mu)\{g_y^{-1}(\phi_x f+\phi_t)\}_{t=t_0} \\ &\quad - \psi(t,t_0,\mu) \int_{t_0}^{t} \psi^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'ds \end{aligned} \tag{6.3.14}$$

where "'" means differentiation with respect to t. Using (6.3.9) we may neglect the second term in (6.3.14) and obtain

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) - \psi(t,t_o,\mu)\int_{t_o}^{t}\psi^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'ds.$$

We again substitute for ψ^{-1} using (6.3.12) and obtain

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) + \mu\psi(t,t_o,\mu)\int_{t_o}^{t}\frac{d\psi^{-1}}{ds}g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'ds.$$

Using integration by parts again we obtain

$$\begin{aligned} w(t,\mu) &= g_y^{-1}(\phi_x f+\phi_t) \\ &\quad + \mu\psi(t,t_o,\mu)[\psi^{-1}(s,t_o,\mu)g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}']_{s=t_o}^{s=t} \\ &\quad - \mu\psi(t,t_o,\mu)\int_{t_o}^{t}\psi^{-1}\{g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'\}'ds \\ &= g_y^{-1}(\phi_x f+\phi_t) + \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' \\ &\quad + \mu\psi(t,t_o,\mu)[g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}']_{t=t_o} \\ &\quad - \mu\psi(t,t_o,\mu)\int_{t_o}^{t}\psi^{-1}\{g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'\}'ds. \end{aligned} \tag{6.3.15}$$

We use (6.3.9) again and neglect the third term of (6.3.15). We also substitute for ψ^{-1} using (6.3.12) and obtain

$$\begin{aligned} w(t,\mu) &= g_y^{-1}(\phi_x f+\phi_t) + \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' \\ &\quad + \mu^2\psi(t,t_o,\mu)\int_{t_o}^{t}\frac{d\psi^{-1}}{ds}g_y^{-1}\{g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}'\}'ds \end{aligned}$$

which we write as

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) + \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' + O(\mu^2). \qquad (6.3.16)$$

We must note however that g_y^{-1} is evaluated such that the y values are in the interval spanned by y and $\phi(x,t)$. However, using (6.3.3) and (6.3.5) we may write

$$g_y = g_y(x,\phi(x,t),t) + \mu\frac{g_{yy}}{2}(x,\phi(x,t),t)w + O(\mu^2).$$

Using Lemma 6.1 we may write

$$g_y^{-1} = g_y^{-1}(x,\phi(x,t),t) - \mu g_y^{-1}(x,\phi(x,t),t)\frac{g_{yy}}{2}w\, g_y^{-1}(x,\phi(x,t),t)+O(\mu^2) \qquad (6.3.17)$$

If we substitute (6.3.17) into (6.3.16) we need only substitute in the first term since we are only concerned with a second order approximation.

We then obtain

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) - \frac{\mu}{2}g_y^{-1}g_{yy}wg_y^{-1}(\phi_x f+\phi_t) + \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' + O(\mu^2) \qquad (6.3.18)$$

where now g_y^{-1} and g_{yy} are evaluated at $(x,\phi(x,t),t)$. This will be the case in what follows.

Since the quantity g_{yy} is symmetric in the lower indices we may use Lemma 6.4 to rewrite (6.3.18) as

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) - \tfrac{\mu}{2} g_y^{-1} g_{yy}\, g_y^{-1}(\phi_x f+\phi_t)w$$

$$+ \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' + O(\mu^2). \qquad (6.3.19)$$

We may then use Lemma 6.3 to solve (6.3.19) for $w(t,\mu)$ resulting in

$$w(t,\mu) = g_y^{-1}(\phi_x f+\phi_t) - \tfrac{\mu}{2} g_y^{-1} g_{yy}[g_y^{-1}(\phi_x f+\phi_t)]^2$$

$$+ \mu g_y^{-1}\{g_y^{-1}(\phi_x f+\phi_t)\}' + O(\mu^2). \qquad (6.3.20)$$

We now substitute (6.3.20) in (6.3.3), keeping only second order terms. This gives us

$$\frac{g(x,y,t)}{\mu} = (\phi_x f+\phi_t) - \tfrac{\mu}{2} g_{yy}[g_y^{-1}(\phi_x f+\phi_t)]^2$$

$$+ \mu\{g_y^{-1}(\phi_x f+\phi_t)\}' + \tfrac{\mu}{2} g_{yy}[g_y^{-1}(\phi_x f+\phi_t)]^2 + O(\mu^2)$$

$$= (\phi_x f+\phi_t) + \mu\{g_y^{-1}(\phi_x f+\phi_t)\}' + O(\mu^2). \qquad (6.3.21)$$

Expanding the second term in (6.3.21) using Lemma 6.2 we have

$$\frac{g(x,y,t)}{\mu} = (\phi_x f+\phi_t) + \mu g_y^{-1}[\phi_{xx} f^2+2\phi_{xt} f+\phi_x f_x f+\phi_x f_t+\phi_{tt}$$

$$+(g_{yx} f+g_{yy}(\phi_x f+\phi_t)+g_{yt}) g_y^{-1}(\phi_x f+\phi_t)] - \mu g_y^{-1}\phi_x f_y \frac{g}{\mu} + O(\mu^2).$$

Using Lemma 6.3 we have our result

$$h(x,y,t,\mu) = (\phi_x f+\phi_t) + \mu g_y^{-1}[\phi_{xx} f^2+2\phi_{xt} f + \phi_x f_x f + \phi_x f_t$$

$$+ \phi_{tt} - (g_{yx} f+g_{yy}(\phi_x f+\phi_t)+g_{yt}) g_y^{-1}(\phi_x f+\phi_t)$$

$$+ \phi_x f_y(\phi_x f+\phi_t)] \qquad (6.3.22)$$

where we are applying (6.1.2). Of course for the autonomous case we have

$$h(x,y,t,\mu) = \phi_x f + \mu g_y^{-1}[\phi_{xx}f^2+\phi_x f_x f - (g_{yx}f+g_{yy}\phi_x f)g_y^{-1}\phi_x f + \phi_x f_y \phi_x f]. \tag{6.3.23}$$

In the formulas (6.3.22) and (6.3.23) the functions f, f_x and f_y are evaluated at (x,y,t) and the functions g_y^{-1}, g_{yx}, g_{yy} and g_{yt} are evaluated at $(x,\phi(x,t),t)$.

4. Accuracy of the Solutions and a Note on Implementation

We now ask how the solutions of (2.0.1) and (6.1.1) compare given (6.1.2). Here we let the solution to (2.0.1) be denoted by $z = \binom{x}{y}$ and the solution to (6.1.1) be denoted by $\bar{Z} = \binom{\bar{x}}{\bar{y}}$. We also let $u = x-\bar{x}$ and $v = y-\bar{y}$. Using (2.0.1) and (6.1.1), we have

$$\begin{aligned} \frac{du}{dt} &= f(x,y,t) - f(\bar{x},\bar{y},t) \\ \frac{dv}{dt} &= \frac{g(x,y,t)}{\mu} - h(\bar{x},\bar{y},t,\mu). \end{aligned} \tag{6.4.1}$$

We assume that we may write (6.1.2) as

$$\frac{g(x,y,t)}{\mu} = h(x,y,t,\mu) + \mu^2 k(t,\mu) \tag{6.4.2}$$

where $k(t,\mu)$ is bounded for $0 \le \mu \le \bar{\mu}$ and $0 < t_1 \le t \le T$. Using (6.4.2) in (6.4.1) we have

$$\frac{du}{dt} = f(x,y,t) - f(\bar{x},\bar{y},t)$$

$$\frac{dv}{dt} = h(x,y,t,\mu) - h(\bar{x},\bar{y},t,\mu) + \mu^2 k(t,\mu).$$

Using Taylor's Theorem on the functions f and h, we have

$$\begin{aligned} \frac{du}{dt} &= f_x u + f_y v \\ \frac{dv}{dt} &= h_x u + h_y v + \mu^2 k(t,\mu) \end{aligned} \tag{6.4.3}$$

where the x and y values in each of the components of f_x, f_y, h_x and h_y are in the intervals spanned by x and $\bar{x}$, and y and $\bar{y}$ respectively.

For $0 < t_1 \le t \le T$ and $0 \le \mu \le \bar{\mu}$ we certainly have the functions $x(t,\mu)$ and $y(t,\mu)$ bounded. The same may also be said for $\bar{x}$ and $\bar{y}$ since they are solutions to (6.1.1) and the function h is made up of continuous functions of $\bar{x}$ and $\bar{y}$ and depends continuously on μ. Since the functions f_x, f_y, h_x and h_y are continuous functions of their arguments we may conclude that the system (6.4.3) may be written in the form

$$\frac{dU}{dt} = A(t,\mu)U + \mu^2 K(t,\mu) \tag{6.4.4}$$

where $U = \begin{pmatrix} u \\ v \end{pmatrix}$, $K(t,\mu) = \begin{pmatrix} 0 \\ k,(t,\mu) \end{pmatrix}$ and the matrix $A(t,\mu)$ is bounded on $0 < t_1 \le t \le T$ and $0 \le \mu \le \bar{\mu}$. The solution to (6.4.4) may be written as

$$U = \psi(t,t_1,\mu)U_1 + \mu^2 \psi(t,t_1,\mu) \int_{t_1}^{t} \psi^{-1}(s,t_1,\mu)K(s,\mu)ds \tag{6.4.5}$$

where $U_1 = U(t_1)$ and $\psi(t,t_1,\mu)$ is the principal matrix solution for the system

$$\frac{dU}{dt} = A(t,\mu)U$$

with initial conditions at t_1.

If we assume $U_1 = 0$ then it is clear from (6.4.5) that we have a second order approximation. In practice of course $U_1 \neq 0$ but we may control U_1 simply by solving the full system (2.0.1) accurately enough.

In any particular application μ is fixed. This means that we must take t_1 large enough so that we are not bothered by any boundary layer effects. One suggestion for implementation then would be to integrate the full system (2.0.1) and make periodic calculations on the system (6.1.1) using initial values obtained from (2.0.1) until the two systems agree within a specified tolerance. From this point, we solve only the system (6.1.1).

5. A Note on Using Linear Systems with Constant Coefficients

For the case of linear systems with constant coefficients we may write (2.0.1) as

$$\begin{aligned} \frac{dx}{dt} &= Ax + By \\ \frac{dy}{dt} &= \frac{C}{\mu}x + \frac{D}{\mu}y \end{aligned} \qquad (6.5.1)$$

where A, B, C and D are constant matrices of order $m \times m$, $m \times n$, $n \times m$ and $n \times n$ respectively. Assumption (2.1.11) implies

$$Re[\lambda_i(D)] < 0.$$

We also assume here that D has only linear elementary divisors.

We also have $\phi(x,t) = -D^{-1}Cx$ and $\phi_x(x,t) = -D^{-1}C$. Using (6.3.23) we may write the approximating system (6.1.1) as

$$\frac{dx}{dt} = Ax + By$$
$$\frac{dy}{dt} = R(Ax + By) \tag{6.5.2}$$

where

$$R = -D^{-1}[C + \mu[D^{-1}C(A-BD^{-1}C)]]. \tag{6.5.3}$$

We realize that the eigenvalues of the matrix

$$\begin{pmatrix} A & B \\ \frac{C}{\mu} & \frac{D}{\mu} \end{pmatrix} \tag{6.5.4}$$

cause the system (6.5.1) to be stiff because of the presence of μ. We also realize that the system (6.5.2) is not stiff as a result of the eigenvalues of the matrix

$$\begin{pmatrix} A & B \\ RA & RB \end{pmatrix}. \tag{6.5.5}$$

In fact, we may show the following propositions.

(A) *For the matrix* (6.5.5), n *eigenvalues are zero and* m *eigenvalues are the eigenvalues of the matrix* A + BR.

(B) *The eigenvalues of* A + BR *approach the eigenvalues of* $A-BD^{-1}C$ *as* $\mu \to 0$.

(C) *For the matrix* (6.5.4), n *eigenvalues approach* $-\infty$ *as* $\mu \to 0$.

(D) *For the same matrix,* m *eigenvalues remain finite as* $\mu \to 0$ *and these may be approximated by the eigenvalues of the matrix* $A - BD^{-1}C$.

These statements show that using (6.5.2) instead of (6.5.1) effectively replaces the large negative eigenvalues associated with (6.5.1) by zero while leaving the other eigenvalues almost the same. Of course, if the eigenvalues of D are all negative as we have assumed then the contribution of the components in the solution from these large negative eigenvalues is negligible anyway.

Proposition A may be shown as follows. We may factor the matrix (6.5.5) as

$$\begin{pmatrix} A & B \\ RA & RB \end{pmatrix} = \begin{pmatrix} I & 0 \\ R & 0 \end{pmatrix} \begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} .$$

But the eigenvalues of the product of two matrices XY are the same as the eigenvalues of the product YX. (See for example Wilkinson [17], page 54). Therefore, the eigenvalues of (6.5.5) are the same as the eigenvalues of

$$\begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix} \begin{pmatrix} I & 0 \\ R & 0 \end{pmatrix} = \begin{pmatrix} A+BR & 0 \\ 0 & 0 \end{pmatrix}$$

and the result follows immediately.

The proof of Proposition B follows directly from the following lemma discussed for example in Wilkinson [17], pp.66-77.

Lemma 6.6 *The eigenvalues of the* $s \times s$ *matrix* $X + \mu Y$ *approach the eigenvalues of* X *as* $\mu \to 0$. *In addition if* X *has linear elementary divisors then the eigenvalues* p_i, $i = 1,2,\ldots,s$ *of* $X + \mu Y$ *are related to the eigenvalues* q_i, $i = 1,2,\ldots,s$ *of* X *as*

$$q_i = p_i + O(\mu), \quad i = 1,2,\ldots,s.$$

Using this lemma and (6.5.3) we have Proposition B.

Proposition C may be shown as follows. The eigenvalues λ_i, $i = 1,2,\ldots,m+n$ of (6.5.4) satisfy

$$\frac{1}{\mu}\begin{pmatrix} A\mu & B\mu \\ C & D \end{pmatrix} V_i = \lambda_i V_i$$

or

$$\begin{pmatrix} A\mu & B\mu \\ C & D \end{pmatrix} V_i = (\mu\lambda_i) V_i$$

where V_i, $i = 1,2,\ldots,m+n$ are the corresponding eigenvectors. This implies that $\delta_i = \mu\lambda_i$, $i = 1,2,\ldots,m+n$ are eigenvalues of the matrix

$$\begin{pmatrix} A\mu & B\mu \\ C & D \end{pmatrix}.$$

But we claim that

$$\delta_i = O(\mu), \quad i = 1,2,\ldots,m \tag{6.5.6}$$

and

$$\delta_i = r_i + O(\mu), \quad i = m+1,m+2,\ldots,m+n \tag{6.5.7}$$

where r_i, $i = m+1, m+2, \ldots, m+n$ are eigenvalues of the matrix D. We may show this claim by examining the matrix

$$\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix}. \tag{6.5.8}$$

Since D has only linear elementary divisors, it is clear that (6.5.8) has only linear elementary divisors by looking at the similarity transformation

$$\begin{pmatrix} I & 0 \\ WD^{-1}C & W \end{pmatrix}\begin{pmatrix} 0 & 0 \\ C & D \end{pmatrix}\begin{pmatrix} I & 0 \\ -D^{-1}C & W^{-1} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & WDW^{-1} \end{pmatrix}$$

where WDW^{-1} is a diagonal matrix. Applying Lemma 6.6 we obtain (6.5.6) and (6.5.7). This implies that

$$\mu\lambda_i = O(\mu), \quad i = 1,2,\ldots,m \tag{6.5.9}$$

and

$$\mu\lambda_i = r_i + O(\mu), \quad i = m+1, m+2, \ldots, m+n. \tag{6.5.10}$$

Proposition C follows immediately from (6.5.10) by dividing by μ.

Proposition D may be shown as follows. Using (6.5.9) we have that m eigenvalues remain finite as $\mu \to 0$. The eigenvalues of (6.5.4) may be obtained by solving the determinantal equation

$$\begin{vmatrix} A-\lambda I & B \\ \frac{C}{\mu} & \frac{D}{\mu} - \lambda I \end{vmatrix} = 0$$

or

$$\begin{vmatrix} A-\lambda I & B \\ C & D-\mu\lambda I \end{vmatrix} = 0. \tag{6.5.11}$$

This may be written as

$$\begin{vmatrix} A-\lambda I-B(D-\mu\lambda I)^{-1}C & 0 \\ C & D-\mu\lambda I \end{vmatrix} = 0 \qquad (6.5.12)$$

where we have added linear combinations of the last n rows of (6.5.11) to the first m rows. For the eigenvalues that remain finite as $\mu \to 0$, we may approximate them by solving the equation obtained by setting, $\mu = 0$ in (6.5.12) giving us

$$\begin{vmatrix} A-BD^{-1}C-\lambda I & 0 \\ C & D \end{vmatrix} = 0. \qquad (6.5.13)$$

Equation (6.5.13) has m roots and these are the eigenvalues of $A-BD^{-1}C$, proving Proposition D.

6. Comments on the Method

Dahlquist [2] examines the system

$$\frac{dx}{dt} = f(x,y,t) \qquad (6.6.1)$$

$$\frac{dy}{dt} = g(x,y,t) - Ay \qquad (6.6.2)$$

$$x(0) = \alpha, \quad y(0) = \beta,$$

in which the Lipschitz constants of f and g are much smaller than the moduli of the eigenvalues of the matrix A. In this paper he uses the SAPS (Smooth Approximate Particular Solution) technique on (6.6.2) and a conventional step by step technique to (6.6.1).

If we may isolate the parameter μ in (6.6.2) that causes the large eigenvalues of A then the asymptotic method described in this chapter is very applicable to (6.6.1) and (6.6.2). The two methods are quite similar with respect to the amount of work necessary in that both require the evaluation of partial derivatives with respect to x and y. The SAPS technique requires the solution of certain algebraic equations while the asymptotic method described here requires the evaluation of some second partial derivatives. The SAPS technique simplifies considerably when we are dealing with a linear system with constant coefficients and this is certainly true with the asymptotic method also. We should note that the SAPS technique is a second order method with respect to the step size used while the asymptotic method is a second order method with respect to the small parameter μ.

Examining (6.3.22) and (6.3.23) we should note that the asymptotic method described here would for arbitrary functions f and g be expensive as far as the calculation of the function h is concerned. However, in certain cases h simplifies considerably. We have already examined the case of linear systems with constant coefficients given by (6.5.2) and (6.5.3). Here we see that the quantity R in (6.5.3) may be calculated once and for all at the beginning and we need only calculate the quantity Ax + By and the multiplication R(Ax + By) for each function evaluation.

The case in which f is nonlinear and g is linear may also be handled rather inexpensively since the second partial derivatives in (6.3.22) disappear. If g is independent of t then g_y^{-1} may be evaluated once and for all at the beginning.

In the case where f is arbitrary and g is quadratic we certainly have partials of f and g to evaluate and g_y^{-1} to find at each stage. However, the second partials in (6.3.22) may be evaluated once and for all at the beginning. Hence, we obtain good approximations to (2.0.1) without any stiffness considerations and with a reasonable increase in the complexity of the functions.

7. Numerical Results for the Asymptotic Method

To emphasize the $O(\mu^2)$ error for this method we have used this method on Problem 5.1, Problem 5.2, Problem 5.3 and Problem 5.4 for various values of μ.

For Problem 5.1 we find the function h is given by

$$h(x,y,t,\mu) = y(1-\mu).$$

For this problem we may find the exact solutions to both (2.0.1) and (6.1.1) as functions of t and μ. Table 6.1 shows the results for various values t and $\mu = 10^{-6}$, 10^{-5} and 10^{-4}.

For Problem 5.2 we find the function h is given by

$$h(x,y,t,\mu) = y(1+ \frac{\mu}{2x}(\frac{y}{x} -1)).$$

Since we are interested only in the error due to the asymptotic nature of our approximation we solved (2.0.1) and (6.1.1) such that the errors due to the numerical integration of these systems was much smaller. We used Gear's subroutine package on (2.0.1) and the classical fourth order Runge-Kutta method on (6.1.1). Table 6.2 shows the results for various values of t and $\mu = 10^{-5}$, 10^{-4} and 10^{-3}.

For Problem 5.3 we find the function h is given by the following (we define X first):

$$X = \frac{2y^{(1)} - 1 - \frac{f(x,y,t)}{x}}{4x}$$

$$h^{(1)} = (-1+7\mu X)f(x,y,t)$$

$$h^{(2)} = (3-10\mu X)f(x,y,t).$$

Again we used Gear's package on (2.0.1) and the classical fourth order Runge-Kutta method on (6.1.1) such that the errors due to the numerical integrations were much smaller than the errors due to the asymptotic error. Table 6.3 shows the results for various values of t and $\mu = 10^{-6}$, 10^{-5} and 10^{-4}.

For problem 5.4 we calculate R in (6.5.3) in order to calculate h. We repeat the same procedure as in the last two examples. Table 6.4 shows the results for various values of t and $\mu = 10^{-6}$, 10^{-5} and 10^{-4}.

t	$\mu=10^{-6}$ error in x	y	$\mu=10^{-5}$ error in x	y	$\mu=10^{-4}$ error in x	y
1/16	-0.39D-14	-0.13D-12	-0.38D-12	-0.13D-10	-0.24D-09	-0.48D-08
2/16	-0.17D-13	-0.28D-12	-0.16D-11	-0.28D-10	-0.59D-09	-0.65D-08
3/16	-0.39D-13	-0.45D-12	-0.39D-11	-0.45D-10	-0.11D-08	-0.84D-08
4/16	-0.73D-13	-0.64D-12	-0.73D-11	-0.64D-10	-0.17D-08	-0.11D-07
5/16	-0.12D-12	-0.85D-12	-0.12D-10	-0.85D-10	-0.24D-08	-0.13D-07
6/16	-0.18D-12	-0.11D-11	-0.18D-10	-0.11D-09	-0.33D-08	-0.16D-07
7/16	-0.26D-12	-0.14D-11	-0.26D-10	-0.13D-09	-0.43D-08	-0.19D-07
8/16	-0.35D-12	-0.16D-11	-0.35D-10	-0.16D-09	-0.56D-08	-0.22D-07
9/16	-0.46D-12	-0.20D-11	-0.46D-10	-0.20D-09	-0.71D-08	-0.25D-07
10/16	-0.60D-12	-0.23D-11	-0.60D-10	-0.23D-09	-0.88D-08	-0.29D-07
11/16	-0.75D-12	-0.27D-11	-0.75D-10	-0.27D-09	-0.11D-07	-0.34D-07
12/16	-0.94D-12	-0.32D-11	-0.94D-10	-0.32D-09	-0.13D-07	-0.39D-07
13/16	-0.12D-11	-0.37D-11	-0.11D-09	-0.37D-09	-0.16D-07	-0.44D-07
14/16	-0.14D-11	-0.42D-11	-0.14D-09	-0.42D-09	-0.19D-07	-0.50D-07
15/16	-0.17D-11	-0.48D-11	-0.17D-09	-0.48D-09	-0.22D-07	-0.56D-07
16/16	-0.20D-11	-0.54D-11	-0.20D-09	-0.54D-09	-0.26D-07	-0.63D-07

Table 6.1

t	$\mu=10^{-5}$ error in x	y	$\mu=10^{-4}$ error in x	y	$\mu=10^{-3}$ error in x	y
1/16	-0.18D-13	-0.50D-12	-0.14D-11	-0.56D-10	-0.14D-09	-0.58D-08
2/16	-0.84D-13	-0.14D-11	-0.74D-11	-0.13D-09	-0.74D-09	-0.13D-07
3/16	-0.20D-12	-0.21D-11	-0.18D-10	-0.21D-09	-0.18D-08	-0.21D-07
4/16	-0.36D-12	-0.29D-11	-0.34D-10	-0.29D-09	-0.34D-08	-0.29D-07
5/16	-0.58D-12	-0.37D-11	-0.55D-10	-0.37D-09	-0.55D-08	-0.37D-07
6/16	-0.85D-12	-0.46D-11	-0.80D-10	-0.45D-09	-0.80D-08	-0.45D-07
7/16	-0.12D-11	-0.54D-11	-0.11D-09	-0.53D-09	-0.11D-07	-0.53D-07
8/16	-0.15D-11	-0.63D-11	-0.15D-09	-0.62D-09	-0.15D-07	-0.62D-07
9/16	-0.20D-11	-0.71D-11	-0.19D-09	-0.71D-09	-0.19D-07	-0.71D-07
10/16	-0.24D-11	-0.81D-11	-0.24D-09	-0.80D-09	-0.24D-07	-0.80D-07
11/16	-0.30D-11	-0.90D-11	-0.29D-09	-0.89D-09	-0.29D-07	-0.89D-07
12/16	-0.36D-11	-0.10D-10	-0.35D-09	-0.99D-09	-0.35D-07	-0.99D-07
13/16	-0.43D-11	-0.11D-10	-0.41D-09	-0.11D-08	-0.41D-07	-0.11D-06
14/16	-0.50D-11	-0.12D-10	-0.48D-09	-0.12D-08	-0.48D-07	-0.12D-06
15/16	-0.58D-11	-0.13D-10	-0.56D-09	-0.13D-08	-0.56D-07	-0.13D-06
16/16	-0.67D-11	-0.14D-10	-0.65D-09	-0.14D-08	-0.65D-07	-0.14D-06

Table 6.2

t	$\mu=10^{-6}$ error in			$\mu=10^{-5}$ error in			$\mu=10^{-4}$ error in		
	x	$y^{(1)}$	$y^{(2)}$	x	$y^{(1)}$	$y^{(2)}$	x	$y^{(1)}$	$y^{(2)}$
1/16	-0.10D-11	-0.43D-11	0.39D-11	-0.10D-09	-0.43D-09	0.39D-09	-0.10D-07	-0.43D-07	0.39D-07
2/16	-0.35D-11	-0.65D-11	0.27D-11	-0.35D-09	-0.64D-09	0.25D-09	-0.35D-07	-0.64D-07	0.25D-07
3/16	-0.69D-11	-0.71D-11	-0.23D-11	-0.68D-09	-0.71D-09	-0.21D-09	-0.68D-07	-0.71D-07	-0.21D-07
4/16	-0.11D-10	-0.68D-11	-0.68D-11	-0.11D-08	-0.68D-09	-0.87D-09	-0.11D-06	-0.68D-07	-0.68D-07
5/16	-0.13D-10	-0.56D-11	-0.17D-10	-0.14D-08	-0.59D-09	-0.18D-08	-0.14D-06	-0.59D-07	-0.17D-06
6/16	-0.18D-10	-0.44D-11	-0.25D-10	-0.18D-08	-0.45D-09	-0.26D-08	-0.18D-06	-0.45D-07	-0.25D-06
7/16	-0.22D-10	-0.27D-11	-0.34D-10	-0.22D-08	-0.28D-09	-0.33D-08	-0.22D-06	-0.28D-07	-0.34D-06
8/16	-0.26D-10	-0.68D-12	-0.42D-10	-0.26D-08	-0.98D-10	-0.42D-08	-0.26D-06	-0.98D-08	-0.42D-06
9/16	-0.29D-10	0.10D-11	-0.51D-10	-0.29D-08	0.94D-10	-0.50D-08	-0.29D-06	0.95D-08	-0.51D-06
10/16	-0.33D-10	0.29D-11	-0.59D-10	-0.33D-08	0.29D-09	-0.58D-08	-0.33D-06	0.29D-07	-0.59D-06
11/16	-0.36D-10	0.48D-11	-0.66D-10	-0.36D-08	0.48D-09	-0.66D-08	-0.36D-06	0.48D-07	-0.66D-06
12/16	-0.39D-10	0.66D-11	-0.74D-10	-0.39D-08	0.67D-09	-0.73D-08	-0.39D-06	0.67D-07	-0.73D-06
13/16	-0.41D-10	0.83D-11	-0.80D-10	-0.41D-08	0.84D-09	-0.80D-08	-0.41D-06	0.84D-07	-0.80D-06
14/16	-0.44D-10	0.10D-10	-0.86D-10	-0.45D-08	0.10D-08	-0.86D-08	-0.44D-06	0.10D-06	-0.86D-06
15/16	-0.46D-10	0.12D-10	-0.92D-10	-0.46D-08	0.12D-08	-0.92D-08	-0.46D-06	0.12D-06	-0.92D-06
16/16	-0.48D-10	0.13D-10	-0.97D-10	-0.47D-08	0.13D-08	-0.97D-08	-0.48D-06	0.13D-06	-0.97D-06

Table 6.3

	errors for $\mu=10^{-6}$			errors for $\mu=10^{-5}$			errors for $\mu=10^{-4}$		
1/16	0.59D-10 0.31D-09	0.26D-10 0.14D-09	0.15D-10	0.39D-08 0.31D-07	0.25D-08 0.14D-07	0.15D-08	0.39D-06 0.31D-05	0.25D-06 0.14D-05	0.15D-06
2/16	0.96D-10 0.25D-09	0.73D-10 0.18D-09	0.61D-10	0.96D-08 0.25D-07	0.73D-08 0.18D-07	0.61D-08	0.96D-06 0.25D-05	0.73D-06 0.18D-05	0.61D-06
3/16	0.10D-09 -0.27D-10	0.10D-09 0.15D-09	0.12D-09	0.10D-07 -0.25D-08	0.10D-07 0.10D-07	0.12D-07	0.10D-05 -0.25D-06	0.10D-05 0.10D-05	0.12D-05
4/16	0.45D-10 -0.26D-09	0.98D-10 0.10D-09	0.16D-09	0.45D-08 -0.26D-07	0.98D-08 0.10D-07	0.16D-07	0.45D-06 -0.28D-05	0.98D-06 0.10D-05	0.16D-05
5/16	-0.43D-10 -0.39D-09	0.65D-10 0.67D-10	0.17D-09	-0.43D-08 -0.39D-07	0.65D-08 0.67D-08	0.17D-07	-0.43D-06 -0.39D-05	0.65D-06 0.67D-06	0.17D-05
6/16	-0.12D-09 -0.32D-09	0.25D-10 0.57D-10	0.15D-09	-0.12D-07 -0.32D-07	0.25D-08 0.57D-08	0.15D-07	-0.12D-05 -0.32D-05	0.25D-06 0.57D-06	0.15D-05
7/16	-0.18D-09 -0.14D-09	-0.51D-11 0.69D-10	0.12D-09	-0.18D-07 -0.14D-07	-0.48D-09 0.69D-08	0.12D-07	-0.18D-05 -0.14D-05	-0.48D-07 0.69D-06	0.12D-05
8/16	-0.15D-09 0.48D-10	-0.14D-10 0.90D-10	0.78D-10	-0.15D-07 0.48D-08	-0.14D-08 0.90D-08	0.78D-08	-0.15D-05 0.48D-06	-0.14D-06 0.90D-06	0.78D-06
9/16	-0.10D-09 0.17D-09	-0.36D-11 0.11D-09	0.50D-10	-0.10D-07 0.17D-07	-0.34D-09 0.11D-07	0.51D-08	-0.10D-05 0.17D-05	-0.34D-07 0.11D-05	0.51D-06
10/16	-0.49D-10 0.21D-09	0.19D-10 0.11D-09	0.40D-10	-0.49D-08 0.21D-07	0.19D-08 0.11D-07	0.40D-08	-0.49D-06 0.21D-05	0.19D-06 0.11D-05	0.40D-06
11/16	-0.45D-11 0.17D-09	0.43D-10 0.11D-09	0.45D-10	-0.48D-09 0.18D-07	0.43D-08 0.11D-07	0.46D-08	-0.48D-07 0.18D-05	0.43D-06 0.11D-05	0.46D-06
12/16	0.19D-10 0.80D-10	0.61D-10 0.99D-10	0.60D-10	0.18D-08 0.80D-08	0.61D-08 0.99D-08	0.60D-08	0.19D-06 0.80D-06	0.61D-06 0.99D-06	0.60D-06
13/16	0.20D-10 -0.39D-11	0.70D-10 0.88D-10	0.76D-10	0.19D-08 -0.32D-09	0.69D-08 0.88D-08	0.76D-08	0.19D-06 -0.33D-07	0.70D-06 0.88D-06	0.77D-06
14/16	0.54D-11 -0.59D-10	0.69D-10 0.80D-10	0.89D-10	0.50D-09 -0.58D-08	0.69D-08 0.80D-08	0.89D-08	0.50D-07 -0.58D-06	0.69D-06 0.80D-06	0.89D-06
15/16	-0.15D-10 -0.77D-10	0.64D-10 0.77D-10	0.95D-10	-0.15D-08 -0.75D-08	0.63D-08 0.77D-08	0.94D-08	-0.15D-06 -0.76D-06	0.63D-06 0.77D-06	0.95D-06
16/16	-0.33D-10 -0.62D-10	0.56D-10 0.78D-10	0.94D-10	-0.32D-08 -0.61D-08	0.56D-08 0.79D-08	0.94D-08	-0.32D-06 -0.61D-06	0.57D-06 0.79D-06	0.94D-06

Table 6.4

CHAPTER VII

CONCLUSIONS

There are certain advantages to using the interpolation scheme. As opposed to using other asymptotic methods, the interpolation scheme does not require much analytic preparation before use. MacMillan's method, the asymptotic method presented in Chapter VI and especially Vasileva's method all require some analytic preparation. Another advantage is that it avoids handling large algebraic systems when m and/or n are large (except possibly in obtaining the degenerate system if $\phi(x,t)$ is not easily attainable). Most methods that handle stiff systems are implicit methods, requiring the solution of algebraic equations.

One of the main advantages of the interpolation scheme is that we may obtain answers outside the boundary layer on the time scale of the slowly varying components without having to integrate the full system (2.0.1) through the boundary layer. With other methods that have this characteristic we often are not sure whether we have the solution to our initial value problem or whether we have another solution with a slightly different initial value for t. It is clear from the error expression (3.2.5) that the interpolation scheme does not have this problem.

We have taken the point of view here, perhaps different than many cases, that we are always striving to provide answers with a relative error no greater than a given tolerance ε. Indeed it is quite possible to obtain highly accurate answers for values of t outside the boundary layer quite efficiently.

The error tolerance here is not just a local error tolerance but a global one and in examples run is very reliable. Even when it fails it often picks up again for larger t and is reliable at these larger values of t. This fail-safe aspect is possible because of the nature of the interpolation scheme.

We have found on examples run that the difference in function evaluations when the interpolation scheme is run versus the classical fourth order Runge-Kutta is about a factor of 100 or more.

In comparing the interpolation scheme to Gear's subroutine we find that the interpolation scheme compares favourably in many cases. For the values of t that we have used we find that the first stage is sometimes substantially better but for subsequent stages it levels off more slowly.

The routine that picks the auxiliary parameters does not pick the absolute optimum but does pick an efficient set in the examples tested.

A disadvantage of the interpolation scheme is that the boundary layer effects are somewhat extended as far as the effects on the numerical calculations are concerned.

As far as the asymptotic method presented in Chapter VI is concerned, we should note that it is necessary to integrate the full system through the boundary layer in order to apply the approximation. However, we certainly obtain a good advantage by letting the system (6.1.1) take over outside the boundary layer. The examples run in Chapter VI certainly illustrate the $O(\mu^2)$ accuracy of the approximation.

APPENDIX A

It is necessary to show the existence of the quantity $\frac{\partial z}{\partial \mu}(t,\mu)$ on the set D defined by the points (t,μ) such that $0 < t_1 \leq t \leq T$ for any $t_1 > 0$ and $0 \leq \mu \leq \bar{\mu}$ where for our purposes $\bar{\mu} = \max_{1 \leq i \leq q} \mu_i$. Vasileva [14] has shown the existence of $\frac{\partial^q z}{\partial \mu^q}(t,\mu)$ on the same interval but the techniques used are different and in my opinion more involved. We have not shown the results for $q > 1$ here but the method would be similar.

Theorem *Let* $z(t,\mu) = \begin{pmatrix} x(t,\mu) \\ y(t,\mu) \end{pmatrix}$ *denote the solution to the system*

$$\frac{dx}{dt} = f(x,y,t), \qquad x|_{t=0} = \alpha$$

$$\mu\frac{dy}{dt} = g(x,y,t), \qquad y|_{t=0} = \beta$$

and let Assumptions A to E of Chapter II be satisfied. Then the quantity $\frac{\partial^q z}{\partial \mu^q}(t,\mu)$ *exists on the interval* $0 \leq \mu \leq \bar{\mu}$ *and* $0 < t_1 \leq t \leq T$ *for any* $t_1 > 0$.

Proof It is clear that the theorem holds for $\mu > 0$. We must show that

$$\lim_{h \to 0}\{[\frac{x(t,\mu+h)-x(t,\mu)}{h}]_{\mu=0}\}$$

and

$$\lim_{h \to 0}\{[\frac{y(t,\mu+h)-y(t,\mu)}{h}]_{\mu=0}\}$$

exist on the interval $0 < t_1 \leq t \leq T$. This is equivalent to showing that

$$\lim_{\mu\to 0} \frac{x(t,\mu)-\tilde{x}_0(t)}{\mu}$$

and

$$\lim_{\mu\to 0} \frac{y(t,\mu)-\tilde{y}_0(t)}{\mu}$$

exist on the same interval.

Let

$$u(t,\mu) = \frac{x(t,\mu)-\tilde{x}_0(t)}{\mu}$$

and (A.1)

$$v(t,\mu) = \frac{y(t,\mu)-\tilde{y}_0(t)}{\mu} .$$

We will actually show that

$$\lim_{\mu\to 0} u(t,\mu) = \tilde{x}_1(t)$$

and (A.2)

$$\lim_{\mu\to 0} v(t,\mu) = \tilde{y}_1(t)$$

where $\tilde{x}_1$, $\tilde{y}_1$ are the solutions to the system (2.1.18) with initial condition $\tilde{x}_1(0)$ given by (2.1.29). We will also show that convergence in (A.2) is uniform on the expanding interval $\mu\tau_0 \le t \le T$.

We now obtain an expression for the quantity $u(t,\mu)$. We have using (2.0.1) and (2.1.2)

$$\frac{du}{dt} = \frac{d}{dt}\left[\frac{x(t,\mu)-\tilde{x}_0(t)}{\mu}\right]$$

$$= \frac{f(x(t,\mu),y(t,\mu),t)-f(\tilde{x}_0(t),\tilde{y}_0(t),t)}{\mu}$$

$$= f_x\left[\frac{x(t,\mu)-\tilde{x}_0(t)}{\mu}\right]$$

$$+ f_y\left[\frac{y(t,\mu)-\tilde{y}_0(t)}{\mu}\right]. \qquad (A.3)$$

The last step follows after applying Taylor's Theorem. Note that the x and y values of the components of f_x and f_y are in the interval spanned by $x(t,\mu)$ and $\tilde{x}_0(t)$, and $y(t,\mu)$ and $\tilde{y}_0(t)$ respectively. Simplifying (A.3) we have

$$\frac{du}{dt} = f_x u + f_y v. \qquad (A.4)$$

Applying Taylor's Theorem on $g(x,y,t)$ as we did to f in (A.3), we have

$$\frac{g(x(t,\mu),y(t,\mu),t)}{\mu} = g_x u + g_y v \qquad (A.5)$$

where we say the same for g_x and g_y as we did for f_x and f_y. Solving (A.5) for v and substituting in (A.4) we have

$$\frac{du}{dt} = a(t,\mu)u + b(t,\mu) \qquad (A.6)$$

where

$$a(t,\mu) = f_x - f_y g_y^{-1} g_x$$

and

$$b(t,\mu) = f_y g_y^{-1} \frac{g(x(t,\mu),y(t,\mu),t)}{\mu}$$

Vasileva [15] has shown

$$\lim_{\mu\to 0} x(t,\mu) = \tilde{x}_0(t) \text{ uniformly on } 0 \le t \le T$$

and

$$\lim_{\mu\to 0} y(t,\mu) = \tilde{y}_0(t) \text{ uniformly on the expanding interval } \mu\tau_0 \le t \le T$$

and therefore the limit as $\mu \to 0$ of the x values of f_x, f_y, g_x and g_y is $\tilde{x}_0(t)$ uniformly on $0 \le t \le T$, and the limit as $\mu \to 0$ of the y values of f_x, f_y, g_x and g_y is $\tilde{y}_0(t)$ uniformly on the expanding interval $\mu\tau_0 \le t \le T$.

We have therefore that

$$\lim_{\mu\to 0} a(t,\mu) = a_0(t)$$

and

$$\lim_{\mu\to 0} b(t,\mu) = b_0(t)$$

uniformly on the expanding interval $\mu\tau_0 \le t \le T$, where

$$a_0(t) = f_x(\tilde{x}_0,\tilde{y}_0,t) - f_y(\tilde{x}_0,\tilde{y}_0,t)g_y^{-1}(\tilde{x}_0,\tilde{y}_0,t)g_x(\tilde{x}_0,\tilde{y}_0,t) \qquad (A.7)$$

and

$$b_0(t) = f_y(\tilde{x}_0,\tilde{y}_0,t)g_y^{-1}(\tilde{x}_0,\tilde{y}_0,t)\frac{d\tilde{y}_0}{dt} .$$

In (A.7) we used the fact that

$$\lim_{\mu\to 0} \frac{g(x(t,\mu),y(t,\mu),t)}{\mu} = \frac{d\tilde{y}_0}{dt} .$$

We may show this by examining (6.3.22). We see that

$$\lim_{\mu\to 0} \frac{g(x(t,\mu),y(t,\mu),t)}{\mu} = \phi_x(\tilde{x}_0,t)f(\tilde{x}_0,\tilde{y}_0,t) + \phi_t(\tilde{x}_0,t).$$

But differentiating $\tilde{y}_0(t)$ in (2.1.2) we also see that

$$\frac{d\tilde{y}_0}{dt} = \phi_x(\tilde{x}_0,t)f(\tilde{x}_0,\tilde{y}_0,t) + \phi_t(\tilde{x}_0,t).$$

If we solve for $\tilde{y}_1$ in (2.1.18b) and substitute in (2.1.18a) we have

$$\frac{d\tilde{x}_1}{dt} = a_0(t)\tilde{x}_1 + b_0(t). \tag{A.8}$$

This implies that the right-hand side of (A.6) approaches the right-hand side of (A.8) uniformly on the interval $\mu\tau_0 \le t \le T$. Therefore, if the initial condition of $u(t,\mu)$ of (A.6) approaches the initial condition $\tilde{x}_1(0)$ as $\mu \to 0$ we have that

$$\lim_{\mu\to 0} u(t,\mu) = \tilde{x}_1(t)$$

by appealing to the theorem on the continuous dependence of the solutions of differential equations on the data. The result

$$\lim_{\mu\to 0} v(t,\mu) = \tilde{y}_1(t)$$

follows immediately by examining (A.5) and (2.1.18b).

It only remains to show that

$$\lim_{\mu\to 0}[u(t,\mu)]_{t=0} = \tilde{x}_1(0).$$

To do this we construct using (A.1)

$$u(\mu\tau,\mu) = \frac{x(\mu\tau,\mu)-\tilde{x}_0(\mu\tau)}{\mu}. \tag{A.9}$$

The quantity $x(\mu\tau,\mu)$ is the solution $\bar{x}(\tau,\mu)$ of the system

$$\begin{aligned} \frac{d\bar{x}}{d\tau} &= \mu f(\bar{x},\bar{y},\mu\tau), \quad \bar{x}(0) = \alpha \\ \frac{d\bar{y}}{d\tau} &= g(\bar{x},\bar{y},\mu\tau), \quad \bar{y}(0) = \beta \end{aligned} \tag{A.10}$$

obtained by making the change of variable $t = \mu\tau$ in the system (2.0.1).

The quantity $\tilde{x}_0(\mu\tau)$ is the solution $\hat{x}(\tau,\mu)$ of the system

$$\begin{aligned} \frac{d\hat{x}}{d\tau} &= f(\hat{x},\hat{y},\mu\tau), \quad x(0) = \alpha \\ 0 &= g(\hat{x},\hat{y},\mu\tau) \end{aligned} \tag{A.11}$$

obtained by making the change of variable $t = \mu\tau$ in the system (2.1.1).

Using (A.9), (A.10) and (A.11) we have

$$\begin{aligned} u(\mu\tau,\mu) &= \frac{\alpha+\mu\int_0^\tau f(\bar{x},\bar{y},\mu\sigma)d\sigma-\alpha-\mu\int_0^\tau f(\hat{x},\hat{y},\mu\sigma)d\sigma}{\mu} \\ &= \int_0^\tau [f(\bar{x},\bar{y},\mu\sigma)-f(\hat{x},\hat{y},\mu\sigma)]d\sigma. \end{aligned} \tag{A.12}$$

Since the right side of (A.10) depends continuously on μ, we have

$$\lim_{\mu\to 0} \bar{x}(\sigma,\mu) = \bar{x}_0(\sigma) = \alpha$$

and (A.13)

$$\lim_{\mu\to 0} \bar{y}(\sigma,\mu) = \bar{y}_0(\sigma)$$

uniformly on $0 \le \sigma \le \tau$. Here the quantities $\bar{x}_0(\tau)$ and $\bar{y}_0(\tau)$ are the solutions to (2.1.14) obtained by setting $\mu = 0$ in (A.10). We also have

$$\lim_{\mu\to 0} \hat{x}(\sigma,\mu) = \hat{x}_0(\sigma) = \alpha$$

and (A.14)

$$\lim_{\mu\to 0} y(\sigma,\mu) = \hat{y}_0(\sigma) = \phi(\alpha,0)$$

uniformly on $0 \le \sigma \le \tau$ since the right side of (A.13) depends continuously on μ. The quantities $\hat{x}_0(\tau)$ and $\hat{y}_0(\tau)$ are the solutions to (2.1.24) obtained by setting $\mu = 0$ in (A.11). Using (A.12), (A.13) and (A.14) we have

$$\lim_{\mu\to 0} u(\mu\tau,\mu) = \int_0^\tau [f(\alpha,\bar{y}_0(\sigma),0) - f(\alpha,\phi(\alpha,0),0)]d\sigma \qquad (A.15)$$

since f is continuous and we have uniform convergence in (A.13) and (A.14).

Let the integral in (A.15) be denoted by $R(\tau)$. Also let

$$\lim_{\tau\to\infty} R(\tau) \text{ be denoted by } R(\infty) \qquad (A.16)$$

and we are assured that this limit exists since from Wasow [16]

$$|\bar{y}(\sigma) - \phi(\alpha,0)| \le ce^{-k\sigma}$$

and f_y is bounded.

From the limit (A.16) we can choose τ_0 so large that the point $(0,R(\tau_0))$ in the (t,x) - plane has distance $\leq \varepsilon/2$ from $(0,R(\infty))$. From the limit (A.15) we can choose $\mu_0(\varepsilon)$ so small that for $\mu < \mu_0(\varepsilon)$, the point $(\mu\tau_0, u(\mu\tau_0,\mu))$ has distance $\leq \varepsilon/2$ from $(0,R(\tau_0))$. Hence $(\mu\tau_0, u(\mu\tau_0,\mu))$ has distance $\leq \varepsilon$ from $(0,R(\infty))$ and ε can be made as small as we please by taking τ_0 sufficiently large and then μ_0 sufficiently small. Note that $R(\infty) = \tilde{x}_1(0)$ in (2.1.29).

The interpretation of the limit taken in this way may be explained as follows.

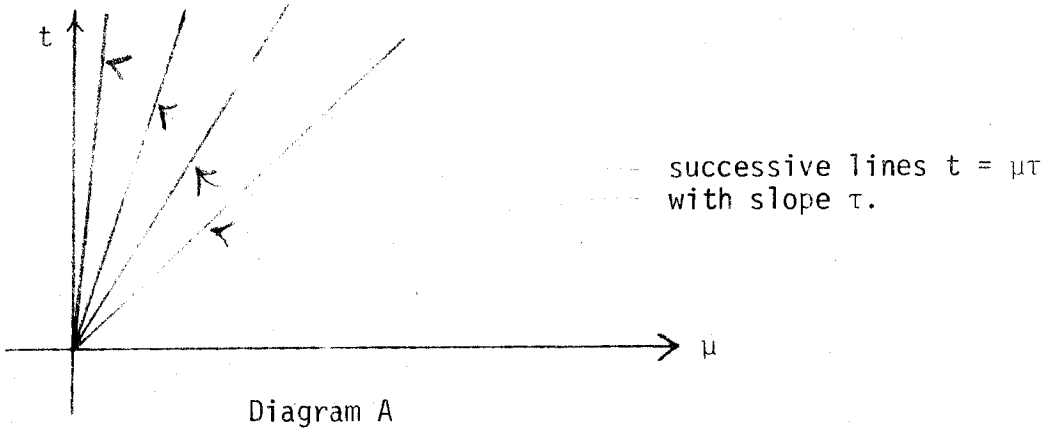

Diagram A

In Diagram A we evaluate $u(t,\mu)$ along successive lines $t = \mu\tau$ with τ increasing. With each of these lines we then allow $\mu \to 0$. The limiting case of this procedure is just the expression

$$\lim_{\mu\to 0} u(t,\mu)]_{t=0}.$$

This completes the proof.

APPENDIX B

Using the notation and assumptions of Chapters II and III we are interested in showing that the relation

$$|\bar{Z}_i^{(j)}(\tau) - \hat{Z}_i^{(j)}(\tau)| \leq ce^{-K\tau}$$

holds for c and κ positive constants independent of τ. Wasow [16] uses the description "of boundary layer type" to refer to a function $H(\tau)$ such that

$$|H(\tau)| \leq ce^{-K\tau} \quad .$$

Here and in what follows c and κ are used as generic constants only and no attempt will be made to distinguish different values.

We first prove the following lemma.

Lemma B *Given the function* $F(x,y,t)$ *where the partial derivative* F_y *is continuous in* y, *then using the notation of Chapter II we have the relation*

$$|F(\alpha,\bar{y}_0(\tau),0) - F(\alpha,\phi(\alpha,0),0)| \leq ce^{-K\tau}. \tag{B.1}$$

Proof Using Taylor's Theorem we have

$$F(\alpha,\bar{y}_0,0) - F(\alpha,\phi(\alpha,0),0) = F_y(\bar{y}_0-\phi(\alpha,0)) \tag{B.2}$$

where the y values in each of the components of F_y are in the interval spanned by $\bar{y}_0(\tau)$ and $\phi(\alpha,0)$. Since $\bar{y}_0(\tau)$ is bounded and F_y is continuous

$$|F_y| \leq K \tag{B.3}$$

for some positive constant K. Also from Wasow [16] we have

$$|\bar{y}_0(\tau) - \phi(\alpha,0)| \leq ce^{-K\tau} . \tag{B.4}$$

The relations (B.2), (B.3) and (B.4) establish (B.1).

We now move on to the main result here.

Theorem B *The relations*

$$|\bar{x}_i^{(j)}(\tau) - x_i^{(j)}(\tau)| \leq ce^{-K\tau} \tag{B.5}$$

and

$$|\bar{y}_i^{(j)}(\tau) - y_i^{(j)}(\tau)| \leq ce^{-K\tau} \tag{B.6}$$

hold for c *and* κ *positive constants independent of* τ *and for all* i *and* j. *The constants* c *and* κ *may depend on* i *and* j. *As in Chapter III the notation* $\bar{Z}_i^{(j)}(\tau)$ *means* $\dfrac{d^j\bar{Z}_i}{d\tau^j}$ *and* $\hat{Z}_i^{(j)}(\tau)$ *means* $\dfrac{d^j\hat{Z}_i}{d\tau^j}$.

Remark Wasow [16] has shown (B.5) and (B.6) for j = 0,1 and all i. The first step is to show (B.5), and (B.6) for all j and i = 0,1 (since these are special cases).

Proof For i = 0 we have $\bar{x}_0(\tau) = \hat{x}_0(\tau) = \alpha$ from (2.1.15) and (2.1.25) and therefore (B.5) follows trivially for all j. Also from (2.1.15) we have

$$\bar{y}_0^{(1)} = g(\alpha,\bar{y}_0,0) \tag{B.7}$$

and using Taylor's Thoerem

$$g(\alpha,\bar{y}_0,0) = g(\alpha,\phi(\alpha,0),0) + g_y(\bar{y}_0-\phi(\alpha,0))$$

$$= g_y(\bar{y}_0-\phi(\alpha,0))$$

where the y values of each of the components in g_y are in the interval spanned by $\bar{y}_0(\tau)$ and $\phi(\alpha,0)$. Since $\bar{y}_0(\tau)$ is bounded and g_y is continuous we may assume g_y is bounded by K. Therefore

$$\begin{aligned} |\bar{y}_0^{(1)}| &= |g(\alpha,\bar{y}_0,0)| \\ &\leq K|\bar{y}_0-\phi(\alpha,0)| \\ &\leq ce^{-K\tau}. \end{aligned} \tag{B.8}$$

From (2.1.25) we have

$$\frac{d\hat{y}_0}{d\tau} = 0$$

and therefore

$$|\bar{y}_0^{(j)}-\hat{y}_0^{(j)}| = |\bar{y}_0^{(j)}|. \tag{B.9}$$

Using (B.7), (B.8) and (B.9) we have

$$\begin{aligned} |\bar{y}_0^{(2)}-\hat{y}_0^{(2)}| &= |g_y(\alpha,\bar{y}_0 0)\bar{y}_0^{(1)}| \\ &\leq |g_y(\alpha,\bar{y}_0,0)||\bar{y}_0^{(1)}| \\ &\leq ce^{-K\tau}. \end{aligned}$$

Since higher derivatives of $\bar{y}_0$ all contain factors of $\bar{y}_0^{(1)}$ (from (B.7)), we have (B.6) with i = 0 and all j.

For i = 1 we have been using (2.1.16) and (2.1.26)

$$\bar{x}_1^{(1)} - \hat{x}_1^{(1)} = f(\alpha,\bar{y}_0,0) - f(\alpha,\phi(\alpha,0),0).$$

Therefore using arguments as before

$$|\bar{x}_1^{(2)} - \bar{x}_1^{(2)}| = |f_y(\alpha,\bar{y}_0,0)\bar{y}_0^{(1)}|$$

$$\leq |f_y(\alpha,\bar{y}_0,0)||\bar{y}_0^{(1)}|$$

$$\leq ce^{-K\tau}.$$

Again, higher derivatives all contain factors of $\bar{y}_0^{(1)}$ and we have (B.5) for $i = 1$ and all j.

In what follows f and g and their partials when evaluated at $(\alpha,\bar{y}_0,0)$ will be denoted by $\bar{f}$, $\bar{g}$ etc. and when evaluated at $(\alpha,\phi(\alpha,0),0)$ will be denoted by $\hat{f}$, $\hat{g}$ etc. Then using (2.1.16) and (2.1.26) again, we have

$$\bar{y}_1^{(1)} - \hat{y}_1^{(1)} = \bar{g}_x\bar{x}_1 + \bar{g}_y\bar{y}_1 + \bar{g}_t\tau - \hat{g}_x\hat{x}_1 - \hat{g}_y\hat{y}_1 - \hat{g}_t\tau.$$

Therefore, we have

$$\bar{y}_1^{(2)} - \hat{y}_1^{(2)} = \bar{g}_{xy}\bar{y}_0^{(1)}\bar{x}_1 + \bar{g}_{yy}\bar{y}_0^{(1)}\bar{y}_1 + \bar{g}_{ty}\bar{y}_0^{(1)}\tau + \bar{g}_x\bar{x}_1^{(1)}$$
$$- \hat{g}_x\hat{x}_1^{(1)} + \bar{g}_y\bar{y}_1 - \hat{g}_y\hat{y}_1^{(1)} + \bar{g}_t - \hat{g}_t . \qquad (B.10)$$

Adding and subtracting appropriate terms we have

$$y_1^{(2)} - \hat{y}_1^{(2)} = \bar{g}_{xy}\bar{y}_0^{(1)}\bar{x}_1 + g_{yy}\bar{y}_0^{(1)}\bar{y}_1 + \bar{g}_{ty}\bar{y}_0^{(1)}\tau + \bar{g}_x\bar{x}_1^{(1)}$$
$$- \bar{g}_x\hat{x}_1^{(1)} + \bar{g}_x\hat{x}_1^{(1)} - \hat{g}_x\hat{x}_1^{(1)} + \bar{g}_y\bar{y}_1^{(1)}$$
$$- \bar{g}_y\hat{y}_1^{(1)} + \bar{g}_y\hat{y}_1^{(1)} - \hat{g}_y\hat{y}_1^{(1)} + \bar{g}_t - \hat{g}_t$$

giving

$$|\bar{y}_1^{(2)} - \hat{y}_1^{(2)}| \leq |\bar{g}_{xy}||\bar{x}_1||\bar{y}_o^{(1)}| + |\bar{g}_{yy}||\bar{y}_1||\bar{y}_o^{(1)}| + |\bar{g}_{ty}||\tau\bar{y}_o^{(1)}|$$

$$+ |\bar{g}_x||\bar{x}_1^{(1)} - \hat{x}_1^{(1)}| + |x_1^{(1)}||\bar{g}_x - \hat{g}_x| + |\bar{g}_y||\bar{y}_1^{(1)} - \hat{y}_1^{(1)}|$$

$$+ |y_1^{(1)}||\bar{g}_y - \hat{g}_y| + |\bar{g}_t - \hat{g}_t|. \qquad \text{(B.11)}$$

Now $\hat{x}_i$ and $\hat{y}_i$ are polynomials in τ (see (2.1.23)) and $|\bar{g}_x - \hat{g}_x|$, $|\bar{g}_y - \hat{g}_y|$ and $|\bar{g}_t - \hat{g}_t|$ are of boundary layer type (using Lemma B). Therefore, the fifth, seventh and eighth terms of (B.11) are of boundary layer type. For the quantities $|\bar{x}_1|$ and $|\bar{y}_1|$ we may write

$$|\bar{x}_1| \leq |\hat{x}_1| + ce^{-K\tau}$$

and (B.12)

$$|\bar{y}_1| \leq |\hat{y}_1| + ce^{-K\tau}$$

using the inequalities

$$|\bar{x}_1| - |\hat{x}_1| \leq |\bar{x}_1 - \hat{x}_1| \leq ce^{-K\tau}$$

and

$$|\bar{y}_1| - |\hat{y}_1| \leq |\bar{y}_1 - \hat{y}_1| \leq ce^{-K\tau}.$$

This combined with the fact that $|\bar{g}_{xy}|$, $|\bar{g}_{yy}|$ and $|\bar{g}_{ty}|$ are bounded and $|\bar{y}_o^{(1)}|$ is of boundary layer type implies that the first, second and third terms of (B.11) are of boundary layer type. Since $|\bar{g}_x|$ and $|\bar{g}_y|$ are

bounded and we know $|\bar{x}_1^{(1)}-\hat{x}_1^{(1)}|$ and $|\bar{y}_1^{(1)}-\hat{y}_1^{(1)}|$ are of boundary layer type we have the fourth and sixth terms of boundary layer type. Hence $|\bar{y}_1^{(2)}-\hat{y}_1^{(2)}|$ is of boundary type.

Looking again at (B.10) we may use induction along with similar arguments to show (B.6) with i = 1 and all j.

We now show (B.5) and (B.6) for i > 2 and j > 2. We use an induction argument by assuming

$$|\bar{z}_r^{(s)}-\hat{z}_r^{(s)}| \le ce^{-K\tau}, \quad r \le i, \; s \le j-1 \tag{B.13}$$

and then show (B.5) and (B.6) for r = i and s = j.

Using (2.1.17) and (2.1.27) we have

$$\bar{x}_i^{(1)}-\hat{x}_i^{(1)} = \bar{f}_x\bar{x}_{i-1} + \bar{f}_y\bar{y}_{i-1} + \bar{p}_{i-2}$$
$$- \hat{f}_x\hat{x}_{i-1} - \hat{f}_y\hat{y}_{i-1} - \hat{p}_{i-2} \tag{B.14}$$

and

$$\bar{y}_i^{(1)} - \hat{y}_i^{(1)} = \bar{g}_x\bar{x}_i + \bar{g}_y\bar{y}_i + \bar{q}_{i-1}$$
$$- \hat{g}_x\hat{x}_i - \hat{g}_y\hat{y}_i - \hat{q}_{i-1}. \tag{B.15}$$

Here $\bar{p}_{i-2}$ and $\hat{p}_{i-2}$ are polynomial functions of τ and of $\bar{x}_k$, $\bar{y}_k$, $k \le i-2$ and $\hat{x}_k$, $\hat{y}_k$, $k \le i-2$ respectively. Also $\bar{q}_{i-1}$ and $\hat{q}_{i-1}$ are polynomial functions of τ and of $\bar{x}_k$, $\bar{y}_k$, $k \le i-1$ and $\hat{x}_k$, $\hat{y}_k$, $k \le i-1$ respectively.

Using (B.14) we have

$$\bar{x}_i^{(2)} - \hat{x}_i^{(2)} = \bar{f}_{xy}\bar{y}_o^{(1)}\bar{x}_{i-1} + \bar{f}_{yy}\bar{y}_o^{(1)}\bar{y}_{i-1}$$

$$+ \bar{f}_x\bar{x}_{i-1}^{(1)} - \hat{f}_x\hat{x}_{i-1}^{(1)} + \bar{f}_y\bar{y}_{i-1}^{(1)} - \hat{f}_y\hat{y}_{i-1}^{(1)}$$

$$+ \sum_{k=1}^{i-2} \frac{\partial \bar{p}_{i-2}}{\partial \bar{x}_k}\bar{x}_k^{(1)} - \sum_{k=1}^{i-2} \frac{\partial \hat{p}_{i-2}}{\partial \hat{x}_k}\hat{x}_k^{(1)}$$

$$+ \sum_{k=1}^{i-2} \frac{\partial \bar{p}_{i-2}}{\partial \bar{y}_k}\bar{y}_k^{(1)} - \sum_{k=1}^{i-2} \frac{\partial \hat{p}_{i-2}}{\partial \hat{y}_k}\hat{y}_k^{(1)} . \qquad \text{(B.16)}$$

Adding and subtracting appropriate terms in (B.16) and using the triangle inequality we have

$$|\bar{x}_i^{(2)} - \hat{x}_i^{(2)}| \le |\bar{f}_{xy}||\bar{x}_{i-1}||\bar{y}_o^{(1)}| + |\bar{f}_{yy}||\bar{y}_{i-1}||\bar{y}_o^{(1)}|$$

$$+ |\bar{f}_x||\bar{x}_{i-1}^{(1)} - \hat{x}_{i-1}^{(1)}| + |\hat{x}_{i-1}^{(1)}||\bar{f}_x - \hat{f}_x|$$

$$+ |\bar{f}_y||\bar{y}_{i-1}^{(1)} - \hat{y}_{i-1}^{(1)}| + |\hat{y}_{i-1}^{(1)}||\bar{f}_y - \hat{f}_y|$$

$$+ \sum_{k=1}^{i-2} |\bar{x}_k^{(1)}||\frac{\partial \bar{p}_{i-2}}{\partial \bar{x}_k} - \frac{\partial \hat{p}_{i-2}}{\partial \hat{x}_k}| + \sum_{k=1}^{i-2} |\frac{\partial \hat{p}_{i-2}}{\partial \hat{x}_k}||\bar{x}_k^{(1)} - \hat{x}_k^{(1)}|$$

$$+ \sum_{k=1}^{i-2} |\bar{y}_k^{(1)}||\frac{\partial \bar{p}_{i-2}}{\partial \bar{y}_k} - \frac{\partial \hat{p}_{i-2}}{\partial \hat{y}_k}| + \sum_{k=1}^{i-2} |\frac{\partial \hat{p}_{i-2}}{\partial \hat{y}_k}||\bar{y}_k^{(1)} - \hat{y}_k^{(1)}| .$$

The quantities $|\bar{f}_{xy}|, |\bar{f}_{yy}|, |\bar{f}_x|$ and $|\bar{f}_y|$ are all bounded. The quantities $\hat{x}_{i-1}^{(1)}$, $\hat{y}_{i-1}^{(1)}$, $\frac{\partial \hat{p}_{i-2}}{\partial \hat{x}_k}$ and $\frac{\partial \hat{p}_{i-2}}{\partial \hat{y}_k}$ are all polynomials in τ (see (2.1.23)).

Also we have the relations.

$$|\bar{x}_{i-1}| \le |\hat{x}_{i-1}| + ce^{-K\tau}$$

$$|\bar{y}_{i-1}| \le |\hat{y}_{i-1}| + ce^{-K\tau}$$

$$|\bar{x}_k^{(1)}| \le |\hat{x}_k^{(1)}| + ce^{-K\tau}, \quad k \le i-2$$

$$|\bar{y}_k^{(1)}| \le |\hat{y}_k^{(1)}| + ce^{-K\tau}, \quad k \le i-2.$$

These can be shown as (B.12) was shown and by assuming (B.13) holds. The quantities $\hat{x}_{i-1}$, $\hat{y}_{i-1}$, $\hat{x}_k^{(1)}$, and $\hat{y}_k^{(1)}$ are all polynomials in τ.

Using these facts we can show that the first two terms are of boundary layer type since $|\bar{y}_0^{(1)}|$ is of boundary layer type. Also the third and fifth terms and the second and fourth sums are of boundary layer type since we are assuming (B.13). The fourth and sixth terms are of boundary layer type using Lemma B. The first and third sums are of boundary layer type since $|\frac{\partial \bar{p}_{i-2}}{\partial \bar{z}_k} - \frac{\partial \hat{p}_{i-2}}{\partial \hat{z}_k}|$ is of boundary layer type. This last statement is true since $\bar{p}_{i-2}$ is made up of the same combinations of $\bar{x}_k$, $\bar{y}_k$, $k \le i-2$ as $\hat{p}_{i-2}$ is of $\hat{x}_k$, $\hat{y}_k$, $k \le i-2$ and we are assuming (B.13) holds.

Using (B.15) and assuming (B.13) we may show

$$|\bar{y}_i^{(2)} - \hat{y}_i^{(2)}| \le ce^{-K\tau}$$

by similar arguments.

Again using (B.14) and (B.15) and assuming (B.13) we may use similar arguments to show (B.5) and (B.6). This completes the proof.

APPENDIX C

Listing of Subroutine for the Interpolation Scheme

```
      SUBROUTINE INTE(X,M,N,MPN,T,TFINAL,U,UU,NP,K,H,TA,XDEG,XA,C,D,
     &  CC,DD,HZ,TOL,L,Z)
C*************************************************************************
C*     This subroutine integrates a system of O.D.E.'s of the form      *
C*                          dx/dt=f(x,y,t)                              *
C*                        udy/dt=g(x,y,t)                               *
C* where u is a small parameter. Starting values of X at T are given    *
C* and the answer at TFINAL is given back in X. Each succeeding call    *
C* should have only a new TFINAL. A method of interpolation on a set    *
C* of auxiliary solutions is used. An explanation of the                *
C* parameters follows.                                                  *
C*  X -A vector of size M+N that contains the starting values for       *
C*     the system. The first M positions correspond to x, the last      *
C*     N positions correspond to y.                                     *
C*  M -The dimension of the vector x(at least 1).                       *
C*  N -The dimension of the vector y(at least 1).                       *
C*  MPN -The quantity M+N.                                              *
C*  T -The starting value of t for this step. It contains TFINAL at     *
C*     the exit of the subroutine.                                      *
C*  TFINAL  -The value of t at which an answer is desired.              *
C*  U -The small parameter.                                             *
C*  UU -An array of size NP. The subroutine requires a set of           *
C*    auxiliary parameters much larger than U to avoid severe           *
C*    stiffness but small enough to provide accurate approximations.    *
C*  NP -The maximum number of auxiliary parameters allowed (5 is        *
C*    usually an upper bound but at least 1 is needed).                 *
C*  K -A parameter that can have 3 possible values:                     *
C*    (a) 1 if the user wants to provide the array UU and does not      *
C*        want the subroutine to try to find it.                        *
C*    (b) 2 if the user wants the subroutine to try to find             *
C*        a good  set of UU on its own. If this value is used, then NP  *
C*        should be set at 4 or 5.                                      *
C*    (c) 3 if this is the second or greater reference to this          *
C*        subroutine during this run. During the first reference        *
C*        the subroutine will automatically set K to 3.                 *
C* The parameter K and the array UU should not be changed after         *
C* the first reference to this subroutine unless it is desired to set a*
C* new set of auxiliary parameters. If this is done K should be set at  *
C* 1 or 2 again depending on the option desired.                        *
C*  H,TA,XDEG,XA -Arrays of size NP,NP,MPN, and NPxMPN respectively     *
C*    used as working storage. They should not be altered by the        *
C*    main program.                                                     *
C*  C,D,CC,DD,HZ -Variables used as working storage. They should        *
C*    not be altered by the main program.                               *
C*  TOL -Maximum error allowed in the answer.                           *
C*  L -A parameter indicating the success of this stage. The            *
C*    possible values are as follows:                                   *
C*    (a) 1 if the stage was successful within the maximum error.       *
C*    (b) 2 if the stage was not successful within the maximum error    *
C*        given but successful within a larger error. The larger        *
C*        error may be found by examining TOL. Note that TOL in this    *
```

```
C*        case should be reset by the main program because it is still *
C*        quite possible to obtain better answers in later steps if    *
C*        desired. The user should always use the same value of TOL.   *
C*     (c) 3 if K has an impossible value.                             *
C*     (d) 4 if the array UU has an element with an unexpected value   *
C*        less than or equal to 0.                                     *
C*     (e) 5 if the small parameter U is less than or equal to 0.      *
C*     (f) 6 if NP is less than 1 or greater than 5.                   *
C*     (g) 7 if the subroutine is not able to find a set of            *
C*        parameters UU.                                               *
C*     (h) 8 if K is 2 and NP is not 4 or 5.                           *
C*   Z -A variable used by the subroutine in checking error tolerances.*
C*                                                                     *
C*     Note that all real variables should be declared as DOUBLE       *
C* PRECISION. In addition the following 3 subroutines should be        *
C* provided by the user and all real variables should be declared as   *
C* DOUBLE PRECISION here also.                                         *
C*                                                                     *
C*     FNEVAL(X,T,U)                                                   *
C*     This subroutine should evaluate the functions f and g/u placing *
C* the results in an array F declared as COMMON /FUNCT/F(10). The first*
C* M elements of F correspond to f and the last N elements correspond  *
C* to g/u.X should be an array of size 10 whose first M elements       *
C* correspond to x and whose last N elements correspond to y. U is the *
C* small parameter. This subroutine is used by INTF to calculate the   *
C* auxillary solutions.                                                *
C*                                                                     *
C*     PHI(X,M,N,T)                                                    *
C*     This subroutine should use some procedure to solve the function *
C* g(x,y,t)=0 for y=p(x,t). This solution p should be a stable solution*
C* such that the initial conditions of our system are in its domain of *
C* influence. X is an array of size MPN=M+N whose first M positions    *
C* correspond to x and whose last N positions correspond to y. T is the*
C* value  of t.                                                        *
C*                                                                     *
C*     EVAL(X,M,N,T)                                                   *
C*     This subroutine is used by INTE to evaluate the degenerate.     *
C* system. It should evaluate the function f(x,p(x,t),t) where p(x,t)  *
C* is described in subroutine PHI. The results  should be placed in    *
C* an array F declared as COMMON /FUNCT/F(10). Only the first M        *
C* elements of F need be calculated. All arguments here are as in the  *
C* subroutine PHI. The values of p(x,t) should be obtained by the      *
C* statement CALL PHI(X,M,N,T).                                        *
C*********************************************************************
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION X(MPN),UU(NP),H(NP),TA(NP),XDEG(MPN),XA(NP,MPN)
      DIMENSION ER(5,10),E(5),TAB(5,5,10),XAA(5,10)
      DIMENSION XO(10),XP(10),UV(5),HA(5),TAA(5),XD(10)
C***Calculate Z used for checking error tolerances.
      Z=1.D0
      DO 90 I=1,M
   90 XP(I)=X(I)
```

```
      CALL PUT(XR,(M),(N),T)
      DO 91 I=1,MPN
      IF(DABS(XR(I)).GT.Z)Z=DABS(XR(I))
   91 CONTINUE
C***Store the starting values of X.
      DO 1 I=1,MPN
    1 XO(I)=X(I)
C***Check the parameters for possible errors.
      IF(K.NE. 2) GO TO 45
      IF(NP.EQ. 5 .OR. NP.EQ. 4) GO TO 45
      L=8
      RETURN
   45 IF(K.GE. 1 .AND.K.LE. 3) GO TO 2
      L=3
      RETURN
    2 IF(NP.LE. 5 .AND. NP.GE. 1) GO TO 40
      L=6
      RETURN
   40 IF(K.EQ. 2) GO TO 38
      DO 37 J=1,NP
      IF(UU(J).GT. 0.D0) GO TO 37
      L=4
      RETURN
   37 CONTINUE
   38 IF(U.GT. 0.D0) GO TO 39
      L=5
      RETURN
   39 GO TO (1000,2000,3000),K
C***Calculate the error tolerances for the auxiliary solutions.
 1000 DO 77 J=1,NP
   77 UV(J)=UU(J)
      NN=np
      CALL ERRTOL(UV,U,NN,C,D,TOL)
C***Assign upper and lower tolerances for the degenerate solution
      CC=TOL*.1D0
      DD=CC*.01D0
C***Calculate the starting step sizes for the auxiliary solutions
C***if necessary.
      IF(K.EQ. 2)GO TO 3000
      CALL AUXSTP(XO,M,N,T,UV,NN,C,HA,Z)
      DO 78 J=1,NP
   78 H(J)=HA(J)
C***Calculate the starting step size for the degenerate system.
      CALL DEGSTP(XO,M,N,T,CC,HZ,Z)
C***Store the values of XO in XDEG and XA and the value of T in TA.
   25 DO 24 I=1,MPN
      XDEG(I)=XO(I)
      DO 24 J=1,NP
   24 XA(J,I)=XO(I)
      DO 23 J=1,NP
   23 TA(J)=T
C***Advance the degenerate solution to T=TFINAL if necessary.
```

```
 3000 IF(T.GE.TFINAL) GO TO 250
      CALL DEGEN(XDEG,M,N,T,HZ,TFINAL,CC,DD,Z)
C***Advance the first auxiliary solution to T=TFINAL if necessary.
  250 IF(TA(1).GE.TFINAL)GO TO 261
      DO 22 I=1,MPN
   22 XR(I)=XA(1,I)
      CALL AUXSOL(XR,M,N,TA(1),UU(1),H(1),TFINAL,C,D,Z)
      DO 260 I=1,MPN
  260 XA(1,I)=XR(I)
C***Start the table for Aitken's Interpolation.
  261 DO 26 I=1,MPN
   26 TAB(1,1,I)=((UU(1)-U)*XDEG(I)+U*XA(1,I))/UU(1)
C***Check whether we have agreement within error tolerance.
      JS=1
      IND=0
      DO 60 I=1,MPN
      ER(1,I)=DABS(TAB(1,1,I)-XDEG(I))/Z
      IF(ER(1,I).GT.TOL*.8D0)IND=1
   60 CONTINUE
      IF(IND.EQ. 0)GO TO 31
      IF(NP.EQ. 1)GO TO 61
C***Keep using more auxiliary solutions until agreement within
C***error tolerance is reached.
      DO 27 J=2,NP
      IND=0
      JS=J
C***Advance each succeeding auxiliary solution when needed
C***if necessary.
      IF(TA(J).GE.TFINAL)GO TO 291
      DO 28 I=1,MPN
   28 XR(I)=XA(J,I)
      CALL AUXSOL(XR,M,N,TA(J),UU(J),H(J),TFINAL,C,D,Z)
      DO 290 I=1,MPN
  290 XA(J,I)=XR(I)
C***Make the next row in the table for Aitken's Interpolation.
  291 DO 29 I=1,MPN
      TAB(J,1,I)=((UU(J)-U)*XDEG(I)+U*XA(J,I))/UU(J)
      DO 29 JJ=2,J
   29 TAB(J,JJ,I)=((UU(J)-U)*TAB(JJ-1,JJ-1,I)-(UU(JJ-1)-U)
     &    *TAB(J,JJ-1,I))/(UU(J)-UU(JJ-1))
      DO 30 I=1,MPN
C***Make the test to see if we have agreement within error
C***tolerance
      ER(J,I)=DABS(TAB(J,J,I)-TAB(J-1,J-1,I))/Z
      IF(ER(J,I).GT.TOL*.8D0) IND=1
   30 CONTINUE
      IF(IND.EQ. 0) GO TO 31
   27 CONTINUE
C***We have not obtained the requested accuracy after using
C***NP values, so we determine what accuracy we do have,
C***set the parameter L and return with our best answers.
   61 DO 32 J=1,NP
```

```
      E(J)=ER(J,I)
      DO 32 I=2,MPN
      IF(ER(J,I).GT.E(J)) E(J)=ER(J,I)
   32 CONTINUE
      JS=1
      EMIN=E(1)
      IF(NP.LE. 1)GO TO 33
      DO 35 J=2,NP
      IF(E(J).GE.EMIN) GO TO 35
      EMIN=E(J)
      JS=J
   35 CONTINUE
   33 L=2
      TOL=EMIN+.2D0*TOL
      DO 34 I=1,MPN
   34 X(I)=TAB(JS,JS,I)
      K=3
      RETURN
C***We are within the specified error tolerance, so we
C***set the parameter L and return with the proper values.
   31 L=1
      DO 36 I=1,MPN
   36 X(I)=TAB(JS,JS,I)
      K=3
      RETURN
 2000 NS=NP
      CALL PARAM(XO,M,N,T,TFINAL,U,UV,NS,TOL,LL,
     &  XAA,HA,TAA,XD,TD,HZ,Z)
      IF(LL.EQ. 0) GO TO 2002
      L=7
      RETURN
 2002 DO 2001 J=1,NP
      DO 2003 I=1,MPN
 2003 XA(J,I)=XAA(J,I)
      H(J)=HA(J)
      TA(J)=TAA(J)
 2001 UU(J)=UV(J)
      DO 2004 I=1,MPN
 2004 XDEG(I)=XD(I)
      T=TD
      GO TO 1000
      END
C
C
C***PARAM is a subroutine that chooses a set of auxiliary parameters
C***based on the first value of t for which answers are desired and
C***the error tolerance.
      SUBROUTINE PARAM(XO,M,N,TO,TLIM,U,UU,NP,TOL,K,
     &  XA,HA,TA,XD,TFINAL,HD,Z)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION XO(10),XD(10),XA(5,10),XR(10),XI(10),XJ(10),XK(10)
      DIMENSION UU(5),HA(5),TAB(5,5,10),E2(10)
```

```
      DIMENSION TA(5),VV(5),UTEMP(5),HTEMP(5)
      MPN=M+N
C***K is used to denote the success or failure of the subroutine.
C***K=0 denotes success, K=1 denotes failure.
      K=0
      IJ=0
      Q=.1D0
      RR=2.D0
      A=100.D0
      S=2.D0
C***If the value of t is too large for the efficiency of this
C***subroutine we run it at a value of .3.
      CHK=.3D0
      TFINAL=TLIM
      TDEL=TFINAL-T0
      IF(TDEL.LE.CHK)GO TO 25
      TDEL=CHK
      TFINAL=T0+TDEL
C***Pick a starting auxiliary parameter.
   25 CALL SOLVE(TDEL,2,S,A,UU(NP))
      NPM1=NP-1
C***Calculate the other starting parameters.
      DO 1 J=1,NPM1
    1 UU(NP-J)=UU(NP-J+1)*.75D0
      CC=TOL*.1D0
      DD=CC*.01D0
C***Calculate the starting step size for the degenerate system.
      CALL DEGSTP(X0,M,N,T0,CC,HD,Z)
      DO 2 I=1,MPN
    2 XD(I)=X0(I)
      TT=T0
C***Calculate the degenerate solution.
      CALL DEGEN(XD,M,N,TT,HD,TFINAL,CC,DD,Z)
C***Calculate the error tolerances for the auxiliary systems.
      CALL ERRTOL(UU,U,NP,C,D,TOL)
C***Calculate the starting step sizes for the auxiliary systems.
      CALL AUXSTP(X0,M,N,T0,UU,NP,C,HA,Z)
C***Calculate NP auxiliary solutions.
      DO 3 J=1,NP
      DO 4 I=1,MPN
    4 XR(I)=X0(I)
      TA(J)=T0
      CALL AUXSOL(XR,M,N,TA(J),UU(J),HA(J),TFINAL,C,D,Z)
      DO 5 I=1,MPN
    5 XA(J,I)=XR(I)
    3 CONTINUE
C***Check that the effects of EXP(-S*TDEL/UU(NP)) in the
C***third derivative with respect to u have been eliminated.
C***This is done by forming the approximations XI,XJ and XK
C***in order to obtain two estimates for the error involved
C***so that an estimate for the third derivative with respect
C***to u may be roughly approximated.
```

```
      C0=(U-UU(NP-1))*(U-UU(NP))/UU(NP-1)/UU(NP)
      C1=U*(U-UU(NP))/UU(NP-1)/(UU(NP-1)-UU(NP))
      C2=U*(U-UU(NP-1))/UU(NP)/(UU(NP)-UU(NP-1))
      DO 6 I=1,MPN
    6 XI(I)=C0*XD(I)+C1*XA(NP-1,I)+C2*XA(NP,I)
      C0=(U-UU(NP-2))*(U-UU(NP-1))/UU(NP-2)/UU(NP-1)
      C1=U*(U-UU(NP-1))/UU(NP-2)/(UU(NP-2)-UU(NP-1))
      C2=U*(U-UU(NP-2))/UU(NP-1)/(UU(NP-1)-UU(NP-2))
      DO 7 I=1,MPN
    7 XJ(I)=C0*XD(I)+C1*XA(NP-2,I)+C2*XA(NP-1,I)
   16 C0=(U-UU(NP-3))*(U-UU(NP-2))/UU(NP-3)/UU(NP-2)
      C1=U*(U-UU(NP-2))/UU(NP-3)/(UU(NP-3)-UU(NP-2))
      C2=U*(U-UU(NP-3))/UU(NP-2)/(UU(NP-2)-UU(NP-3))
      DO 8 I=1,MPN
    8 XK(I)=C0*XD(I)+C1*XA(NP-3,I)+C2*XA(NP-2,I)
      DO 9 I=1,MPN
    9 E2(I)=DABS(XI(I)-XJ(I))/Z
      L=1
      EMAX=E2(1)
      DO 10 I=2,MPN
      IF(E2(I).LE.EMAX) GO TO 10
      L=I
      EMAX=E2(I)
   10 CONTINUE
      YB=6*E2(L)/U/(U-UU(NP-1))/(U-UU(NP))
      E1=DABS(XJ(L)-XK(L))/Z
      YA=6*E1/U/(U-UU(NP-2))/(U-UU(NP-1))
      YMIN=YA
      IF(YB.LT.YA) YMIN=YB
C***If the effects have not been eliminated we multiply the
C***auxiliary parameters by .75, store the necessary values
C***and repeat the process.
      IF(DABS(YB-YA)/YMIN.LT.RR) GO TO 12
      YB=YA
      DO 13 J=1,NPM1
      NPMJ=NP-J
      UU(NPMJ+1)=UU(NPMJ)
      DO 14 I=1,MPN
   14 XA(NPMJ+1,I)=XA(NPMJ,I)
   13 CONTINUE
      UU(1)=UU(2)*.75D0
      CALL ERRTOL(UU,U,NP,C,D,TOL)
      CALL AUXSTP(X0,M,N,T0,UU,1,C,HA,Z)
      DO 17 I=1,MPN
   17 XR(I)=X0(I)
      TA(1)=T0
      CALL AUXSOL(XR,M,N,TA(1),UU(1),HA(1),TFINAL,C,D,Z)
      DO 15 I=1,MPN
      XA(1,I)=XR(I)
      XI(I)=XJ(I)
   15 XJ(I)=XK(I)
      IJ=IJ+1
```

```
C***If we are not successful after 5 tries we return indicating
C***the failure.
      IF(IJ.LE. 5) GO TO 16
      K=1
      RETURN
C***Set up Aitken's table.
   12 DO 11 I=1,MPN
      DO 18 J=1,NP
   18 TAB(J,1,I)=((UU(J)-U)*XD(I)-U*XA(J,I))/UU(J)
      DO 19 J=2,NP
      DO 20 JJ=2,J
      TAB(J,JJ,I)=((UU(J)-U)*TAB(JJ-1,JJ-1,I)-(UU(JJ-1)-U)*TAB(J,JJ-1,I)
     &    )/(UU(J)-UU(JJ-1))
   20 CONTINUE
   19 CONTINUE
C***Calculate the error in using NP-1 auxiliary parameters.
      E2(I)=DABS(TAB(NP,NP,I)-TAB(NP-1,NP-1,I))/Z
   11 CONTINUE
      EMAX=E2(1)
      DO 21 I=2,MPN
      IF(E2(I).GT.EMAX)EMAX=E2(I)
   21 CONTINUE
      FACT=1.D0
      DO 22 J=1,NPM1
   22 FACT=FACT*(U-UU(J))/J
      NN=NPM1
C***Use the error to calculate a set of auxiliary parameters
C***based on the error tolerance and the error involved in using
C***NP-1 auxiliary parameters.
      UMIN=(TOL*Q*DABS(FACT)/EMAX)**(1.D0/NN)
      UMAX=UMIN*(NP-1)
C***Keep only the auxiliary parameters that are in the range.
C***For those out of the range new auxiliary parameters are
C***obtained using EMAX in the error expression for Lagrange
C***interpolation. For each of the new auxiliary parameters
C***its values of X,T,H,C and D are set. The auxiliary
C***parameters that are kept and the degenerate solution keep
C***the current values of these quantities.
      IS=0
      DO 100 I=1,NP
      IF(UU(I).LT.UMIN.OR.UU(I).GT.UMAX)GO TO 100
      IS=IS+1
      VV(IS)=UU(I)
      DO 105 II=1,MPN
  105 XA(IS,II)=XA(I,II)
      TA(IS)=TA(I)
      HA(IS)=HA(I)
  100 CONTINUE
      IF(IS.GT. 0)GO TO 101
      DO 102 I=1,NP
      DO 150 II=1,MPN
  150 XA(I,II)=XO(II)
```

```
      TA(I)=TO
  102 VV(I)=I*UMIN
      CALL ERRTOL(VV,U,NP,C,D,TOL)
      CALL AUXSTP(XO,M,N,TO,VV,NP,C,HA,Z)
      GO TO 200
  101 IF(IS.NE.NP)GO TO 103
      IF(EMAX.LE.TOL*O)GO TO 200
      VV(1)=TOL*O/EMAX*VV(1)
      DO 106 II=1,MPN
  106 XA(1,II)=XO(I)
      TA(1)=TO
      CALL ERRTOL(VV,U,NP,C,D,TOL)
      CALL AUXSTP(XO,M,N,TO,VV,1,C,HA,Z)
      GO TO 200
  103 IF(IS.NE.NP-1)GO TO 104
      F1=1.DO
      F2=1.DO
      DO 107 I=1,NPM1
      F1=F1*VV(I)
  107 F2=F2*UU(I)
      IF(F1.GT.F2*TOL*O/EMAX)GO TO 108
      VV(NP)=4.DO/3.DO*VV(NP-1)
      DO 109 II=1,MPN
  109 XA(NP,II)=XO(II)
      TA(NP)=TO
      CALL ERRTOL(VV,U,NP,C,D,TOL)
      UTEMP(1)=VV(NP)
      CALL AUXSTP(XO,M,N,TO,UTEMP,1,C,HTEMP,Z)
      HA(NP)=HTEMP(1)
      GO TO 200
  108 VV(NP-1)=TOL*O/EMAX*F2/F1*VV(NP-1)
      III=1
  111 IF(DABS(VV(NP-1)-VV(III)).LT. 1.D-4)GO TO 110
      III=III+1
      IF(III.LE.NP-2)GO TO 111
      GO TO 112
  110 IF(VV(NP-1).GT.VV(III))GO TO 113
      VV(NP-1)=VV(NP-1)-1.D-4
      GO TO 112
  113 VV(NP-1)=VV(NP-1)+1.D-4
  112 VV(NP)=16.DO/9.DO*VV(NP-2)
      DO 114 I=1,MPN
      XA(NP-1,I)=XO(I)
  114 XA(NP,I)=XO(I)
      TA(NP-1)=TO
      TA(NP)=TO
      CALL ERRTOL(VV,U,NP,C,D,TOL)
      UTEMP(1)=VV(NP-1)
      UTEMP(2)=VV(NP)
      CALL AUXSTP(XO,M,N,TO,UTEMP,2,C,HTEMP,Z)
      HA(NP-1)=HTEMP(1)
      HA(NP)=HTEMP(2)
```

```
      GO TO 200
  104 F2=1.D0
      DO 115 I=1,NPM1
  115 F2=F2*UU(I)
      F1=1.D0
      DO 116 I=1,IS
  116 F1=F1*VV(I)
      IFACT=1
      IJK=NP-IS-1
      DO 117 I=1,IJK
  117 IFACT=IFACT*I
      VV(IS+1)=(TOL*0/EMAX*F2/F1/IFACT)**(1.D0/IJK)
      IJKP1=IJK+1
      DO 118 I=1,IJKP1
  118 VV(IS+I)=VV(IS+1)*I
      JJ=1
  123 III=1
  120 IF(DABS(VV(IS+JJ)-VV(III)).LT. 1.D-4)GO TO 119
      III=III+1
      IF(III.LE.IS)GO TO 120
      GO TO 121
  119 IF(VV(IS+JJ).GT.VV(III))GO TO 122
      VV(IS+JJ)=VV(IS+JJ)-1.D-4
      GO TO 121
  122 VV(IS+JJ)=VV(IS+JJ)+1.D-4
  121 JJ=JJ+1
      IF(IS+JJ.LE.NP)GO TO 123
      ISP1=IS+1
      DO 124 J=ISP1,NP
      DO 125 I=1,MPN
  125 XA(J,I)=XO(I)
      TA(J)=TO
  124 UTEMP(J-IS)=VV(J)
      CALL ERRTOL(VV,U,NP,C,D,TOL)
      CALL AUXSTP(XO,M,N,TO,UTEMP,IJKP1,C,HTEMP,Z)
      DO 127 J=ISP1,NP
  127 HA(J)=HTEMP(J-IS)
  200 DO 126 J=1,NP
  126 UU(J)=VV(J)
      RETURN
      END
C
C
C***SOLVE is a subroutine that finds the smallest root of
C***u*ln(u)+Cu+D=0 if it exists where C=ln(A)/2(K+1) and
C***D=S*T/2(K+1).
      SUBROUTINE SOLVE(T,K,S,A,W)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      C=DLOG(A)/(2*(K+1))
      D=S*T/(2*(K+1))
C***Choose the starting value at the minimum value of the function.
      W=DEXP(-(1.D0+C))
```

```
      IF(W*DLOG(W)+C*W+D .GT. 0.D0) GO TO 7
C***Find V and W such that we have a change in sign when the function
C***is evaluated here.
    1 V=W*.1D0
      FV=V*DLOG(V)+C*V+D
      IF(FV.GE. 0.D0)GO TO 2
      W=V
      GO TO 1
C***Test for convergence of Newton's method.
    2 IF(FV/(V*(DLOG(V)+1.D0+C)**2).LT. 1.D0) GO TO 5
C***Continually refine V and W until the test for convergence of
C***Newton's method is satisfied.
    3 VW=(V+W)/2.D0
      FV=VW*DLOG(VW)+C*VW+D
      IF(FV.GE. 0.D0) GO TO 4
      W=VW
      GO TO 3
    4 V=VW
      GO TO 2
C***Apply Newton's method to convergence.
    5 W=(V-D)/(DLOG(V)+1.D0+C)
      IF(DABS(W-V)/DABS(W).LE. .5D-4)GO TO 6
      V=W
      GO TO 5
C***The maximum value we allow is .01.
    6 IF(W.GT. .01D0)W=.01D0
      RETURN
    7 W=.01D0
      RETURN
      END
C
C
C***ERRTOL is a subroutine that calculates the error tolerances
C***necessary to integrate the auxiliary systems so that the over
C***all error stays within TOL.
      SUBROUTINE ERRTOL(UU,U,NP,C,D,TOL)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION B(5),UU(5)
      SUM=0.D0
      DO 77 I=1,NP
      B(I)=1.D0
      DO 78 J=1,NP
      IF(I.EQ.J) GO TO 78
      B(I)=B(I)*(U-UU(J))/(UU(I)-UU(J))
   78 CONTINUE
      B(I)=B(I)*U/UU(I)
   77 SUM=SUM+B(I)
      C=TOL/SUM*.1D0
      D=C*.01D0
      RETURN
      END
C
```

```
C
C***AUXSTP is a subroutine that calculates the starting
C***step sizes for a set of auxiliary systems by using a
C***1-step, 2-step method.
      SUBROUTINE AUXSTP(XO,M,N,T,UU,NP,C,H,Z)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION XR(10),XS(10),XT(10),XO(10),UU(5),H(5)
      MPN=M+N
      DO 3 J=1,NP
      H(J)=UU(J)
      DO 4 I=1,MPN
    4 XR(I)=XO(I)
      TT=T
      CALL RK(XR,M,N,TT,UU(J),H(J),1)
   10 H(J)=H(J)/2.D0
      DO 5 I=1,MPN
    5 XS(I)=XO(I)
      TT=T
      CALL RK(XS,M,N,TT,UU(J),H(J),1)
      DO 6 I=1,MPN
    6 XT(I)=XS(I)
      CALL RK(XT,M,N,TT,UU(J),H(J),1)
      DO 7 I=1,MPN
      IF(DABS(XT(I)-XR(I))/Z.GT.C)GO TO 8
    7 CONTINUE
      H(J)=H(J)*2.D0
      GO TO 3
    8 DO 9 I=1,MPN
    9 XR(I)=XS(I)
      GO TO 10
    3 CONTINUE
      RETURN
      END
C
C
C***DEGSTP is a subroutine that calculates the starting
C***step size for the degenerate system by using a
C***1-step, 2-step method.
      SUBROUTINE DEGSTP(XO,M,N,T,CC,HZ,Z)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION XR(10),XS(10),XT(10),XO(10)
      MPN=M+N
      HZ=CC**.25D0
      DO 11 I=1,M
   11 XR(I)=XO(I)
      TT=T
      CALL RUNGE(XR,M,N,TT,HZ,1)
   17 HZ=HZ/2.D0
      DO 12 I=1,M
   12 XS(I)=XO(I)
      TT=T
      CALL RUNGE(XS,M,N,TT,HZ,1)
```

```
      DO 13 I=1,M
   13 XT(I)=XS(I)
                     CALL RUNGE(XT,M,N,TT,HZ,1)
      DO 14 I=1,MPN
      IF(DABS(XT(I)-XR(I))/Z.GT.CC)GO TO 15
   14 CONTINUE
      HZ=HZ*2.D0
      RETURN
   15 DO 16 I=1,MPN
   16 XR(I)=XS(I)
      GO TO 17
      END
C
C
C***DEGEN is a subroutine that integrates the degenerate system
C***using a fourth order Runge-Kutta method with a modified
C***1-step, 2-step method.
      SUBROUTINE DEGEN(X,M,N,T,H,TFINAL,C,D,Z)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION X(10),XX(10),XXX(10),E(10)
      MPN=M+N
C***K is the number of steps we take before checking accuracy.
      K=8
    7 DELTA=TFINAL-T
      IF(K*H.GT. DELTA) GO TO 1
      NS=K
   18 TO=T
      DO 2 I=1,M
      XX(I)=X(I)
    2 XXX(I)=X(I)
C***Apply Runge-Kutta with a step size H.
      CALL RUNGE(XX,M,N,TO,H,NS)
      TO=T
    9 H=H/2.D0
      NS=NS*2
C***Apply Runge-Kutta with a step size H/2.
      CALL RUNGE(XXX,M,N,TO,H,NS)
C***Check for agreement within the error tolerance and refine
C***H and repeat if the error tolerance is not satisfied.
      DO 3 I=1,MPN
      E(I)=DABS(XX(I)-XXX(I))/Z
      IF(E(I).GT.C) GO TO 4
    3 CONTINUE
      H=H*2.D0
      DO 5 I=1,MPN
    5 X(I)=XXX(I)
      T=TO
C***If the error is smaller than the minimum tolerance we
C***double H for the next application.
      DO 6 I=1,MPN
      IF(E(I).GT.D) GO TO 7
    6 CONTINUE
```

```
      H=H*2.D0
      GO TO 7
    4 DO 8 I=1,MPN
      XX(I)=XXX(I)
    8 XXX(I)=X(I)
      TO=T
      GO TO 9
    1 IF(DELTA.GT. 0.D0) GO TO 10
      RETURN
   10 IF(DELTA.GT.H) GO TO 11
C***If the step size H is too large for the value of t needed we
C***define HH to fit exactly and perform the 1-step,2-step with
C***this value of HH.
      HH=DELTA
      NS=1
      TO=T
      DO 12 I=1,M
      XX(I)=X(I)
   12 XXX(I)=X(I)
      CALL RUNGE(XX,M,N,TO,HH,NS)
      TO=T
   17 HH=HH/2.D0
      NS=NS*2
      CALL RUNGE(XXX,M,N,TO,HH,NS)
      DO 13 I=1,MPN
      E(I)=DABS(XX(I)-XXX(I))/Z
      IF(E(I).GT.C)GO TO 14
   13 CONTINUE
      DO 15 I=1,MPN
   15 X(I)=XXX(I)
      T=TO
      RETURN
   14 DO 16 I=1,MPN
      XX(I)=XXX(I)
   16 XXX(I)=X(I)
      TO=T
      GO TO 17
   11 NS=DELTA/H
      GO TO 18
      END
C
C
C***AUXSOL is a subroutine that integrates the auxiliary systems
C***using a fourth order Runge-Kutta method with a modified
C***1-step, 2-step method.
      SUBROUTINE AUXSOL(X,M,N,T,U,H,TFINAL,C,D,Z)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION X(10),XX(10),XXX(10),E(10)
      MPN=M+N
C***K is the number of steps we take before checking accuracy.
      K=1
    7 DELTA=TFINAL-T
```

```
      IF(K*H.GT. DELTA) GO TO 1
      NS=K
   18 TO=T
      DO 2 I=1,MPN
      XX(I)=X(I)
    2 XXX(I)=X(I)
C***Apply Runge-Kutta with a step size H.
      CALL RK(XX,M,N,TO,U,H,NS)
      TO=T
    9 H=H/2.DO
      NS=NS*2
C***Apply Runge-Kutta with a step size H/2.
      CALL RK(XXX,M,N,TO,U,H,NS)
C***Check for agreement within the error tolerance and refine
C***H and repeat if the error tolerance is not satisfied.
      DO 3 I=1,MPN
      E(I)=DABS(XX(I)-XXX(I))/Z
      IF(E(I).GT.C) GO TO 4
    3 CONTINUE
      H=H*2.DO
      DO 5 I=1,MPN
    5 X(I)=XXX(I)
      T=TO
C***If the error is smaller than the minimum tolerance we
C***double H for the next application.
      DO 6 I=1,MPN
      IF(E(I).GT.D) GO TO 7
    6 CONTINUE
      H=H*2.DO
      GO TO 7
    4 DO 8 I=1,MPN
      XX(I)=XXX(I)
    8 XXX(I)=X(I)
      TO=T
      GO TO 9
    1 IF(DELTA.GT. 0.DO) GO TO 10
      RETURN
   10 IF(DELTA.GT.H) GO TO 11
C***If the step size H is too large for the value of t needed we
C***define HH to fit exactly and perform the 1-step,2-step with
C***this value of HH.
      HH=DELTA
      NS=1
      TO=T
      DO 12 I=1,MPN
      XX(I)=X(I)
   12 XXX(I)=X(I)
      CALL RK(XX,M,N,TO,U,HH,NS)
      TO=T
   17 HH=HH/2.DO
      NS=NS*2
      CALL RK(XXX,M,N,TO,U,HH,NS)
```

```
      DO 13 I=1,MPN
      E(I)=DABS(XX(I)-XXX(I))/Z
      IF(E(I).GT.C)GO TO 14
   13 CONTINUE
      DO 15 I=1,MPN
   15 X(I)=XXX(I)
      T=TO
      RETURN
   14 DO 16 I=1,MPN
      XX(I)=XXX(I)
   16 XXX(I)=X(I)
      TO=T
      GO TO 17
   11 NS=DELTA/H
      GO TO 18
      END
C
C
C***RUNGE is a subroutine that actually implements a fourth
C***order Runge-Kutta method for DEGEN.
      SUBROUTINE RUNGE(X,M,N,T,H,NS)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /FUNCT/F(10)
      DIMENSION X(10),XT(10),A(10),B(10),C(10)
      DO 5 J=1,NS
      DO 8 I=1,M
    8 XT(I)=X(I)
      CALL EVAL(XT,M,N,T)
      DO 2 I=1,M
      A(I)=F(I)
    2 XT(I)=X(I)+H*A(I)/2.D0
      T=T+H/2.D0
      CALL EVAL(XT,M,N,T)
      DO 3 I=1,M
      B(I)=F(I)
    3 XT(I)=X(I)+H*B(I)/2.D0
      CALL EVAL(XT,M,N,T)
      DO 4 I=1,M
      C(I)=F(I)
    4 XT(I)=X(I)+H*C(I)
      T=T+H/2.D0
      CALL EVAL(XT,M,N,T)
      DO 5 I=1,M
    5 X(I)=X(I)+H*(A(I)+2.D0*B(I)+2.D0*C(I)+F(I))/6.D0
      CALL PHI(X,M,N,T)
      RETURN
      END
C
C
C***RK is a subroutine that actually implements a fourth
C***order Runge-Kutta method for AUXSOL.
      SUBROUTINE RK(X,M,N,T,U,H,NS)
```

```
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /FUNCT/F(10)
      DIMENSION X(10),XT(10),A(10),B(10),C(10)
      MPN=M+N
      DO 5 J=1,NS
      DO 8 I=1,MPN
    8 XT(I)=X(I)
      CALL FNEVAL(XT,T,U)
      DO 2 I=1,MPN
      A(I)=F(I)
    2 XT(I)=X(I)+H*A(I)/2.D0
      T=T+H/2.D0
      CALL FNEVAL(XT,T,U)
      DO 3 I=1,MPN
      B(I)=F(I)
    3 XT(I)=X(I)+H*B(I)/2.D0
      CALL FNEVAL(XT,T,U)
      DO 4 I=1,MPN
      C(I)=F(I)
    4 XT(I)=X(I)+H*C(I)
      T=T+H/2.D0
      CALL FNEVAL(XT,T,U)
      DO 5 I=1,MPN
    5 X(I)=X(I)+H*(A(I)+2.D0*B(I)+2.D0*C(I)+F(I))/6.D0
      RETURN
      END
C
C
C***FNEVAL is a subroutine that evaluates the functions f and g/u.
C***This subroutine is used by RK and must be supplied by the user.
      SUBROUTINE FNEVAL(X,T,U)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /FUNCT/F(10)
      DIMENSION X(10)
      F(1)=X(2)
      F(2)=(X(1)**2-X(2)**2)/U
      RETURN
      END
C
C
C***PHI is a subroutine that solves g(x,y,t)=0 as y=p(x,t).
C***This subroutine is used by EVAL and RUNGE and must be supplied
C***by the user.
      SUBROUTINE PHI(X,M,N,T)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DIMENSION X(10)
      X(2)=X(1)
      RETURN
      END
C
C
C***EVAL is a subroutine that evaluates the function f(x,p(x,t),t)
```

```
C***where p is explained in PHI.This subroutine is used by DEGEN
C***and must be supplied by the user.
      SUBROUTINE EVAL(X,M,N,T)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /FUNCT/F(10)
      DIMENSION X(10)
      CALL PHI(X,M,N,T)
      F(1)=X(2)
      RETURN
      END
```

BIBLIOGRAPHY

[1] Bjurel, G., Dahlquist, G., Lindberg, G., Linde, S., and Oden, L., Survey of stiff ordinary differential equations, Report NA70.11 (1970), Dept. of Inf. Proc., The Royal Institute of Tech., Stockholm.

[2] Dahlquist, G., A numerical method for some ordinary differential equations with large Lipschitz constants, IFIP Congress (1968).

[3] Finden, W., An interpolation scheme to solve systems of ordinary differential equations in which a small parameter multiplies some of the derivatives, Proceedings of the Third Manitoba Conference on Numerical Mathematics, 1973.

[4] Flatto, L. and Levinson, N., Periodic solutions of singularly perturbed systems, J. Rat. Mech. Analysis 4 (1955), 943-950.

[5] Gear, C.W., Numerical initial value problems in ordinary differential equations, Prentice-Hall, 1971.

[6] Gear, C.W., The automatic integration of stiff ordinary differential equations, IFIP Report (1968c).

[7] Hull, T.E., Enright, W.H., Fellen, B.M., and Sedgwick, A.E., Comparing numerical methods for ordinary differential equations, SIAM J. Numerical Analysis, Vol. 9, Dec. 1972, 603-637.

[8] Lawson, J.D., An order five Runge-Kutta process with extended region of stability, Siam J. Numer. Anal., Vol. 3, No. 4 (1966), 593-597.

[9] Lawson, J.D., Generalized Runge-Kutta processes for stable systems with large Lipschitz constants, SIAM J. Numerical Analysis, Vol.4, No.3 (1967).

[10] MacMillan, D.B., Asymptotic methods for systems of differential equations in which some variables have very short response times, SIAM J. Appl. Math., Vol.16, No.4 (1968), 704-721.

[11] Miranker, W.L., Numerical methods of boundary layer type for stiff systems of differential equations, IBM Res. Rep. RC3881 (1972).

[12] Miranker, W.L., Matricial difference schemes for integrating stiff systems of ordinary differential equations, Math. Comp., 25(1971),717-728.

[13] Ortega and Rheinboldt, Iterative solution of nonlinear equations in several variables, Academic Press, New York, 1970.

[14] Vasileva, A.B. [1959a], On repeated differentiation with respect to the parameter of solutions of systems of ordinary differential equations with a small parameter multiplying the derivative, Mat. Sb. (Russian), 48 (90), 311-334.

[15] Vasileva, A.B. [1963], Asymptotic behaviour of solutions of certain problems for ordinary nonlinear differential equations with a small parameter multiplying the highest derivatives, Usp. Mat. Nauk (Russian), 18, 15-86.

[16] Wasow, W., Asymptotic Expansions for ordinary differential equations, Interscience, New York, 1965.

[17] Wilkinson, J.H., The algebraic eigenvalue problem, Oxford University Press, London.