stichting
mathematisch
centrum

MC

AFDELING NUMERIEKE WISKUNDE
(DEPARTMENT OF NUMERICAL MATHEMATICS)

NW 38/77 JUNI

J.G. VERWER

AN IMPLEMENTATION OF A CLASS OF STABILIZED, EXPLICIT METHODS FOR THE TIME INTEGRATION OF PARABOLIC EQUATIONS

Preprint

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O).

AMS(MOS) subject classification scheme (1970): 65L05, M10, M20

An implementation of a class of stabilized, explicit methods for the time integration of parabolic equations *)

by

J.G. Verwer

ABSTRACT

The paper deals with an inplementation of a class of explicit three-step Runge-Kutta methods for the numerical solution of initial value problems for systems of ordinary differential equations. The systems we have in mind do originate from parabolic partial differential equations by applying the method of semi-discretization. The underlying schemes are stabilized and of first and second order. The number of function evaluations per step varies between two and twelve. The implementation is provided with steplength, error and order control. A FORTRAN version of the implementation is available. Numerical results of this FORTRAN program, applied to two semi-discretized problems, are reported.

KEYWORDS & PHRASES: *Numerical analysis, Parabolic partial differential equations, Semi-discretization, Implementation of time integrators.*

---

*) This report will be submitted for publication elsewhere

1. INTRODUCTION

In this paper we discuss the implementation of a class of explicit methods to be used for the time integration of semi-discretized parabolic partial differential equations. The semi-discretized system of ordinary differential equations is supposed to be in the autonomous form

$$y' = f(y). \tag{1.1}$$

An important property, possessed by the major part of semi-discretized parabolic systems, is that the spectrum of the Jacobian matrix, say J(y), is almost real, i.e. the eigenvalues are situated in a long narrow strip around the negative axis of the complex plane. This property is essential for the implemented class of methods. Therefore it must be assumed that the problems, to which our time integrator is applied, possess this property.

At the present time many numerical methods exist for solving time depent partial differential equations (see RICHTMYER & MORTON [8]). When dealing with more than one dimension, the greater part of these methods are not so easy to apply and can only efficiently be implemented for narrow classes of problems. As a consequence, the development of mathematical software for wide classes of linear, and also non-linear, partial differential equations is still in a very early state (see SINCOVEC & MADSEN [11] for a list of references). This is in direct contrast to the development in the field of ordinary differential equations (see e.g. SHAMPINE & GORDON [10]). Very capable software exists for wide classes of non-linear ordinary differential equations. By way of the method of semi-discretization, we can make use of the developments in this field (e.g. steplength and error control) for the implementation of a time integrator.

In this connection stabilized, explicit integration formulas are suited, because of the fact that these formulas, when used in conjunction with semi-discretization, are easy to apply, and can be implemented for wide classes of linear and non-linear problems in one and more dimensions. The only mathematical restriction, to be posed for parabolic problems, is that the

spectrum of the Jacobian of the semi-discretized system is almost real. A practical restriction, with respect to the application of such methods, may arise when the spectral radius of J(y) is extremely large. In spite of the relatively large stability boundaries, the methods are then forced to integrate with very small steps, which may result in an excessive runtime. In such a situation it may be preferable to use an implicit method, which is unconditionally stable (see RICHTMYER & MORTON [8]).

The integrator discussed is based on three-step Runge-Kutta formulas of order one and two. These formulas are stabilized with respect to the real boundary of absolute stability. In fact, the stability regions are long narrow strips around the negative axis of the complex plane. The stabilization of the formulas is achieved by using extra evaluations of the function f per integration step. The number of function evaluations may vary between two and twelve. The analysis and construction of the formulas is discussed in VERWER [14]. In section 2 of this paper we shall give a short review of the theoretical aspects. Section 3 is devoted to the actual implementation. In this section simple mechanisms for the steplength, error and order control are discussed. In section 4 we discuss M3RK which is a FORTRAN program based on the implementation discussed in section 3. The last section of this paper is devoted to a discussion of some numerical results obtained with M3RK.

Finally it should be noted that an ALGOL 60 version of an implementation of stabilized, explicit one-step Runge-Kutta methods (cf. VAN DER HOUWEN [13]) has already been given by BEENTJES [6].

2. THE CLASS OF INTEGRATION FORMULAS

In this section we shortly discuss the underlying class of methods. The analysis and construction of the formulas is extensively discussed in VERWER [14,15,16], where one can also find further references.

Our class of three-step Runge-Kutta formulas may be represented as follows:

$$y_{n+1}^{(0)} = y_n,$$

$$(2.1)\qquad y_{n+1}^{(j)} = (1-b_j)y_n + b_j y_{n-1} + c_j hf(y_{n-1}) + \lambda_j hf(y_{n+1}^{(j-1)}), \quad j = 1,\ldots,m,$$

$$y_{n+1} = dy_{n+1}^{(m)} + (1-d)y_{n-2}, \quad m \geq 2, \quad n = 2,3,\ldots.$$

The vector y_n denotes the approximation to the analytical solution $y(x)$ at $x = x_n$. The points x_j, $j = n-2,\ldots,n+1$, denote the reference points of the three-step formula and h denotes the steplength, i.e. $h = x_{n+1} - x_n$. In the present section h is supposed to be constant. For the application of (2.1) the additional starting vectors y_1 and y_2 must be given. Formula (2.1) is called a three-step formula of degree m, m being the number of function evaluations per integration step.

When applied to the scalar test-model

$$(2.2)\qquad y' = \delta y,$$

scheme (2.1) yields the linear recurrence relation

$$(2.3)\qquad y_{n+1} = dS(z)y_n + dP(z)y_{n-1} + (1-d)y_{n-2},$$

where $S(z)$ and $P(z)$ are polynomials of degree m in $z = h\delta$. The stability of method (2.1) depends on the parameter d and the coefficients of the stability polynomials S and P. We have implemented schemes of order $p = 1$ and $p = 2$, with degree m satisfying $2 \leq m \leq 12$. The corresponding polynomials S and P are such that the absolute stability regions contain a long narrow strip around the negative axis. We have

$$(2.4)\qquad \beta_1(m) \simeq 5.15\ m^2, \quad \beta_2(m) \simeq 2.29\ m^2,$$

where $\beta_1(m)$ denotes the real boundary of absolute stability for the p-th order scheme of degree m. For real eigenvalues, the extrema of the amplification factors of (2.3) are bounded by about 0.9 in the stability interval. Because of this we have a strong damping for the higher harmonics.

The integration parameters of (2.1) are expressed in the parameter d and the coefficients of S and P.

They are determined in such a way that the principal local truncation error, LTE_p say, is given by (see VERWER [14])

$$
\begin{aligned}
&LTE_1 \simeq C_2 h^2 y^{(2)}(x_n), \qquad C_2 \simeq 1.27, \\
&LTE_2 \simeq C_3 h^3 y^{(3)}(x_n), \qquad C_3 \simeq 0.44.
\end{aligned} \tag{2.5}
$$

Observe that LTE_p does not depend on m. For p = 1,2 and m = 2,...,12, the coefficients s_i of S and p_i of P are given in VERWER [14]. The parameters b_j, c_j and λ_j are given by

$$
\begin{aligned}
b_m &= p_0, \\
c_m &= \frac{(1 - \tfrac{1}{2}p_0)(p_1 - 2p_2 + 2p_3 + 2s_3) - (\tfrac{1}{2} + \tfrac{1}{4}p_0)^2}{2 + p_1 - 2p_2 + 2p_3 + 2s_3}, \\
\lambda_m &= 1 - \tfrac{1}{2}p_0 - c_m, \\
b_j &= 0, \qquad j = 1,\ldots,m-2, \\
b_{m-1} &= \frac{p_1 - c_m}{\lambda_m}, \\
c_j &= \frac{p_{m+1-j}}{s_{m+1-j}}, \qquad j = 1,\ldots,m-2, \\
c_{m-1} &= \frac{p_2}{\lambda_m}, \\
\lambda_j &= \frac{s_{m+1-j}}{s_{m-j}}, \qquad j = 1,\ldots,m-2, \\
\lambda_{m-1} &= \frac{s_2}{\lambda_m}.
\end{aligned} \tag{2.6}
$$

The parameter d is independent of m and is given by

$$
d = 1.375, \quad p = 1, \qquad d = 0.775, \quad p = 2. \tag{2.7}
$$

Finally we mention the concept of internal stability. Because of the relatively large degree and relatively large stability boundaries, we have to deal with an accumulation of rounding errors which appears per integration step. Especially for the higher degree formulas it can easily reduce the local accuracy. For a formula of degree m this accumulation is approximately governed by a so-called internal stability function, say $Q_{m-1}^{[p]}(z)$, which is a strongly increasing polynomial of degree m - 1 Let σ denote the spectral radius. Then the accumulation is generally under control if we adjust the steplength h and the degree m to the so-called internal stability condition

$$(2.8) \qquad Q_{m-1}^{[p]}(h\sigma(J(y_n))) \leq \frac{\text{maximal local truncation}}{\text{arithmetic precision}} .$$

In the program we shall use the values $Q_{m-1}^{[p]}(\beta_p(m))$. For future reference, the values $Q_{m-1}^{[1]}(\beta_1(m))$ are listed in table 2.1. The values for the second order schemes, which are used in the program, are defined by $Q_{m-1}^{[2]}(\beta_2(m)) = 10^2\, Q_{m-1}^{[1]}(\beta_1(m))$.

m	2	3	4	5	6	7	8	9	10	11	12
	$3_{10}{}^{1}$	$1_{10}{}^{2}$	$7_{10}{}^{2}$	$4_{10}{}^{3}$	$3_{10}{}^{4}$	$2_{10}{}^{5}$	$9_{10}{}^{5}$	$5_{10}{}^{6}$	$3_{10}{}^{7}$	$2_{10}{}^{8}$	$1_{10}{}^{9}$

Table 2.1 The values $Q_{m-1}^{[1]}(\beta_1(m))$.

Finally we note that scheme (2.1) needs six arrays of storage. For the actual implementation discussed in the next section we also use six arrays.

3. THE IMPLEMENTATION OF THREE-STEP RUNGE-KUTTA FORMULAS

When integrating time dependent partial differential equations by using the method of semi-discretization there arise two types of discretization errors, viz. the error due to the spatial discretization and the error due to the time integration. In general the first error can not be controlled. To our opinion it is nevertheless useful to supply a method for the time

integration with various control mechanisms, if possible. By doing this one relieves the task of the user of such an integrator. To support this opinion we make the following observation. Our methods are conditionally stable. When applied to a non-linear system, it may then happen that a sudden instability arises because of an increase of the spectral radius. When a method is supplied with error and steplength control, such a sudden instability is immediately detected and the steplength is decreased.

The most widely applied implementation technique for linear multistep methods is nowadays the Nordsieck technique (see GEAR [5]). This technique makes it is very easy to realize error, steplength and also order control. Compared with a Lagrange implementation, i.e. an implementation where the y- and y'-values are stored, a Nordsieck implementation is less efficient for large systems because of the higher overhead costs. Therefore we prefer the Lagrange implementation for our formulas. As we have to deal with a low order and with three-step formulas, this yields no particular problems.

The greater part of the ideas we apply are well known and extensively discussed in the literature (see e.g. GEAR [5] and SHAMPINE & GORDON [10]). We shall therefore omit details where possible, but still observe that (as usual) most of the ideas we apply are based partly on theoretical arguments, and partly on heuristics.

3.1. THE START OF THE PROCESS

The two additional starting vectors y_1 and y_2 are computed by means of a one-step Runge-Kutta scheme of order $p = 2$, which is also formulated as a three-step scheme by introducing zero-parameters. It is obtained from (2.1) by putting

$$(3.1) \qquad d = 1,\ b_j = c_j = 0,\ \lambda_j = r_{m+1-j}/r_{m-j}, \qquad j = 1,\ldots,m;\quad 2 \le m \le 12,$$

where r_j, $j = 0,\ldots,m$, denote the coefficients of the corresponding m-th degree stability polynomial, say R_m. For $m = 2$ this polynomial is given by $R_2(z) = 1 + z + \frac{1}{2}z^2$. For $3 \le m \le 12$ this polynomial is chosen equal to the stabilized polynomial $\tilde{R}_m^{(2)}$ given by VAN DER HOUWEN [13, table 2.6.7']. The extrema of $\tilde{R}_m^{(2)}$ in its real interval of absolute stability, say $(-\tilde{\beta}_2(m),0)$, are bounded by 0.95. Observe that for $m = 2$ no specific damping properties are imposed. For $m = 2$ the absolute stability boundary is 2. For convenience we approximate the boundaries $\tilde{\beta}_2(m)$ with

$$\tilde{\beta}_2(m) \simeq 0.44\ m^2 + 0.03\ m^3. \tag{3.2}$$

If $m \neq 12$, these approximations are slightly smaller than the true boundaries of $\tilde{R}_m^{(2)}$.

The internal stability behaviour of the starting schemes is roughly the same as that of the three-step schemes. In particular, the values given in table (2.1) hold for the starting schemes.

In order to start the process we need an initial steplength, say h_{start}. This initial steplength should be related to the local tolerance, say TOL, which is specified by the user. We estimate h_{start} as follows. Let $\sigma_0 = \sigma(J(y_0))$, σ denoting the spectral radius, be given (if σ_0 is not available, it is estimated by the program as outlined in section 3.5). Let $\|\ .\ \|$ denote the divided Euclidean norm (i.e. Euclidean norm divided by the square root of the number of components). The idea is now to estimate $\frac{1}{2}\sigma_0^{-2} y^{(2)}(x_0)$ which represents the last Taylor term taken into account by the actual start formula, obtained with stepsize σ_0^{-1}. By relating this conservative estimation of the principal local truncation error of the start formula with TOL, we reasonably obtain a safe estimation of h_{start}. Following this idea h_{start} is then defined by

$$h_{start} = \sigma_0^{-1}(\eta_t / \eta_e)^{1/2}/10, \tag{3.3}$$

where

$$\begin{aligned} \eta_t &= TOL + TOL * \|y_0\|, \\ \eta_e &= \sigma_0^{-1} \|f(y_o + \sigma_0^{-1} f(y_0)) - f(y_0)\|. \end{aligned} \tag{3.4}$$

It is observed that in the root formula (3.3), which is used to extrapolate a new stepsize(cf. GEAR [5],p.156), the estimation of $\frac{1}{2}\sigma_0^{-2} y^{(2)}(x_0)$ is multiplied by 200 to obtain an extra safety margin.

In order to obtain absolute stability at the start of the process, the initial steplength must satisfy the stability condition

$$h_{start}\sigma_0 \leq \tilde{\beta}_2(m_{max}), \tag{3.5}$$

where m_{max} denotes the maximal degree allowed with respect to internal stability. The estimation of σ_0 and m_{max} is discussed in section 3.5.
If h_{start} does not satisfy (3.5), we put $h_{start} = b(m_{max})/s_0$.
If h_{start} does not satisfy (3.5), we put $h_{start} = \tilde{\beta}_2(m_{max})/\sigma_0$.

3.2. ESTIMATION AND CONTROL OF THE LOCAL ERROR

For the error control we use the local truncation error which is estimated by LTE_p (see(2.5)). For the estimation of LTE_p, p= 1,2, we apply the simple interpolation formulas (n > 2)

$$
\begin{aligned}
LTE_1 &\approx \frac{c_2}{c_2 + \frac{1}{2}} [y_{n+1} - 2y_n + y_{n-1}], \quad c_2 = \frac{1}{2} - C_2, \\
LTE_2 &\approx \frac{c_3}{c_3 + \frac{5}{6}} [y_{n+1} - 3(y_n - y_{n-1}) + y_{n-2}], \quad c_3 = \frac{1}{6} - C_3,
\end{aligned} \tag{3.6}
$$

where, according to the definition of LTE_p, y_{n-1} and y_{n-2} are assumed to be approximations of a sufficiently high order to a local analytical solution at the points x_{n-1} and x_{n-2}, respectively.

The error criterion which is to be performed after each n-th integration step, n > 2, is the mixed criterion

$$
\|LTE_p\| \leq TOL + TOL * \|y_{n+1}\|, \tag{3.7}
$$

where TOL stands for a user specified tolerance parameter and $\|\cdot\|$ denotes the divided Euclidean norm. If (3.7) is satisfied the integration step is accepted, otherwise rejected.

During the start of the process, i.e. if n = 1,2, no error control is performed. This is justified by the conservative estimation of the initial steplength. If the third step fails however, all results are rejected and the process is restarted with h = h/10.

3.3. CHANGING THE STEPLENGTH

Before discussing the estimation and control of the steplength we first mention how we realize the change. Our integration formulas (2.1) are

developed for a fixed steplength. This means changing stepsize must be handled apart. In a Nordsieck implementation, changing stepsize is an interpolation-extrapolation process (see GEAR [5]). We also use interpolation-extrapolation to determine the new y-values.

Let h and αh denote the old and new steplength, respectively. The new values of y_{n-1} and y_{n-2} are then interpolated or extrapolated by means of the quadratic formula

$$(3.8)\quad y(x-\bar{\alpha}h) \simeq \tfrac{1}{2}\bar{\alpha}(\bar{\alpha}-1)y(x-2h) + \bar{\alpha}(2-\bar{\alpha})y(x-h) + \tfrac{1}{2}(2-\bar{\alpha})(1-\bar{\alpha})y(x),$$

where $\bar{\alpha} = \alpha$ and $\bar{\alpha} = 2\alpha$, respectively. Formula (3.8) is applied for $p = 1$ and $p = 2$. The error introduced by (3.8) is of order three and is ignored at the estimation of LTE_2.

Applying (3.8) too frequently may lead to severe instabilities. Therefore we also use the rule of thumb: after a change of h at least 4 steps are performed with h fixed, provided a step is not rejected. We return to this point in the next section. The new values of y'_{n-1} are not computed by means of interpolation or extrapolation, but by using the derivative function f(y). To our experience this leads to a more stable process of steplength changing.

3.4. ESTIMATION AND CONTROL OF THE STEPLENGTH AND ORDER

The new steplength αh is estimated using the well known root formula. Let $\bar{\alpha}$ be defined by

$$(3.9)\qquad \bar{\alpha} = \left(\frac{TOL + TOL * \|y_{n+1}\|}{\|LTE_p\|} \right)^{\frac{1}{p+1}}.$$

Then we put

$$(3.10)\qquad \alpha = \begin{cases} \bar{\alpha}/2.0, & p = 1, \\ \bar{\alpha}/1.6, & p = 2. \end{cases}$$

The factors 2.5 and 1.3 are to provide a conservative estimate. In order to prevent marginal changes the change is not performed when $0.9 < \alpha < 1.1$. Moreover, in order to prevent an excessive decrease or increase of the steplength, α is bounded by 0.1 and 3.0, respectively. Because of the factors in (3.10), a decrease of the steplength is not necessarily due to a step failure.

The estimate of α is made when a step fails or at least 4 steps have been performed after the last change. Because of the fact that repeated rejections may be caused by severe errors, the process is interrupted if this happens three times in succession. After this interruption we make a restart as described in section 3.1.

Our formulas are explicit and thus conditionally stable. Therefore the steplength h is always bounded by

$$(3.11) \qquad h_{max}(p) = \beta(m_{max}) \; / \; \sigma,$$

$h_{max}(p)$ being the maximal steplength with respect to absolute stability. In (3.11), $\beta(m_{max})$ stands for $\beta_1(m_{max})$, $\beta_2(m_{max})$ or $\tilde{\beta}_2(m_{max})$.

Next we mention the implemented order control which is very simple and based on the fact that, in general, the steplength is bounded by stability requirements. As already observed the process is always started with a second order one-step scheme and a second order three-step scheme. During the process h normally increases until $h = h_{max}(2)$. If h reaches this value, 4 steps are performed with the second order scheme and $h = h_{max}(2)$, provided no step failure occurs. Then, α is estimated for $p = 1$. If this particular $\alpha < 1.1$, the process is continued with $h = h_{max}(2)$ and the current second order scheme. Otherwise the process is continued with a first order scheme, but, as a matter of caution, with $h = h_{max}(2)$. Then, the next time α is estimated, h is allowed to increase while $p = 1$. This specific check for an order decrease is made every four steps, provided $h = h_{max}(2)$.

If during a first order integration h becomes smaller than $h_{max}(2)$, p is reset to 2. It is observed that an order increase is not necessarily due to a step failure.

3.5. ESTIMATION AND CONTROL OF THE DEGREE AND SPECTRAL RADIUS

In order to control the propagation of local errors per integration step we want to satisfy condition (2.8). This is achieved by putting $m \leq m_{max}$, m_{max} being the maximal degree of the schemes, which satisfies (see table 2.1)

$$(3.12) \qquad Q_{m-1}^{[p]}(\beta_p(m)) \leq \frac{\text{TOL}}{\text{arithmetic precision}} \cdot$$

For the three-step schemes m_{max} thus depends on p; there holds $m_{max}(2) \leq m_{max}(1)$. The maximal degree for the starting scheme is chosen equal to $m_{max}(2)$ of the three-step scheme. If the exceptional situation arises that $m_{max} < 2$ (the quotient of TOL and arithmetic precision is too small), the process is discontinued.

A property of stabilized methods is that the local truncation error of the formulas is approximately independent of m. Thus it is useful to mini - mize m with respect to the stability condition

$$(3.13) \qquad h\sigma \leq \beta(m)$$

for given h and σ, while $\beta(m)$ represents $\beta_1(m)$, $\beta_2(m)$ and $\tilde{\beta}_2(m)$, respectively. The degree m is computed in this way at the start of the process, at the change-over from a one-step to a three-step scheme, and further every time h or p is changed.

From the foregoing it is clear that we need an estimation of σ. Because of the fact that σ is used to determine $h_{max}(p)$ and to select m minimal with respect to (3.13), it must always be an upper estimation. Once σ is estimated and the system to be integrated is non-linear, it may be necessary to control the variation of σ. Thus we also need a control mechanism for the spectral radius. With respect to the estimation and control of σ we distinguish between 3 options, which option is chosen has to be specified by the user.

OPTION I. The user provides an estimation of σ. Especially for linear problems this is often easy to do.

OPTION II. The user does not provide an estimation. In this case σ is estimated by means of a power method which is adapted for general non-linear vector functions f(cf. LINDBERG [6]). This method may be described as follows.

Suppose $\sigma_0 = \sigma(J(y_0))$ is to be estimated. Let r_i be a random number from $[-\varepsilon,\varepsilon]$, $\varepsilon > 0$, and let v_0 be defined componentwise by

$$v_{0,i} = \begin{cases} y_{0,i}(1+r_i), & y_{0,i} \neq 0, \\ r_i, & y_{0,i} = 0. \end{cases} \tag{3.14}$$

Let $\varepsilon_{max} = \max(\varepsilon, \varepsilon \|v_0\|_2)$. The adapted power method is then defined by the iteration

$$v_{j+1} = v_0 + \varepsilon_{max} \frac{f(v_j) - f(v_0)}{\|f(v_j) - f(v_0)\|_2}, \tag{3.15}$$

$$\rho_{j+1} = \frac{\|f(v_{j+1}) - f(v_0)\|_2}{\varepsilon_{max}},$$

where $v_1 = y_0$ and $j = 1,2,\ldots$. If f is linear, i.e. J(y) constant, then $\rho_j \to \sigma_0$. If f is non-linear, then choosing ε sufficiently small, J(y) is approximately constant in $S(v_0,\varepsilon)$. Thus for ε sufficiently small, ρ_j will converge to an accurate estimation of σ_0.

In our program we have set $\varepsilon = 10^4$APR, APR denoting the arithmetic precision (e.g. for CDC Cyber $\varepsilon = 10^{-10}$). The iteration (3.15) is stopped as soon as $|\rho_{j+1} - \rho_j| \leq 10^{-3}\rho_{j+1}$, provided $j \geq 4$. If this inequality is not satisfied within 50 iterations, the whole process is discontinued. In general, the process converges slowly because of the absence of a dominant eigenvalue. As a matter of safety we therefore put $\sigma_0 = 1.1\rho$, ρ being the last iterate.

In case of the second option the variation of the spectral radius is also controlled. We distinguish between two situations. Firstly, σ increases during the course of the integration and the process becomes unstable. The instability is immediately detected by the error control and results in a step failure. After a step failure we therefore simply reestimate σ, provided the failure was not in succession. Secondly, σ decreases during the course of the integration. To detect a decrease of σ we use an inaccurate estimation of σ, say σ^*, which is given by $\sigma^* = \rho_3$. The estimation σ^* is computed every 25 steps since the last estimation of σ or σ^*, and we decide to reestimate σ if σ^* has been decreased with more than ten percent.

We note that ρ_3 is in most cases a very rough estimation to σ. To our purpose this rough estimation suffices. At this place it is emphasized that random values are used in formula (3.14). Though these values are small, they may slightly influence σ and, in particular, ρ_3. As a consequence, one should use a random generator using a fixed generative value to be able to recover earlier obtained results (see also section 4).

OPTION III. The user does not provide an estimation, but decides that an initial estimation by means of the power method at the start suffices. Thus in this case no control on the variation of σ is performed. For linear problems it is clear that one chooses between the first and third option.

3.6. ALGORITHMIC CONNECTION BETWEEN THE CONTROL MECHANISMS

This short section is added in order to clarify the algorithmic connection between the mechanisms for controlling the order, the degree, the stepsize and the spectral radius. To this end an informal flowchart showing this connection is given in fig. 3.1. In this flowchart it is assumed that we choose option 2 for estimating and controlling the spectral radius. Because of the fact that during the start of the process no control is performed (see section 3.2), it is also assumed that we are already integrating with a first or second order three-step scheme (in particular it is assumed that $n > 3$). Note that at the start of the process the spectral radius is estimated, the maximal degree m_{max} is determined, the initial steplength h_{start} is estimated, the maximal steplength $h_{max}(p)$, $p = 1,2$, given by (3.11), is determined (see at the end of section 3.1), and the degree is minimized according to (3.13). A more detailed flowchart showing the complete implementation is given in the next section.

4. THE PROGRAM

In this section we describe a FORTRAN version of the implementation discussed in the previous section. The program consists of a main program and 11 small subprograms which are written to structure the program in order to make it more easily readable and easy to modify. We emphasize that the subprograms are meant to be called by the main program, which is the sub-

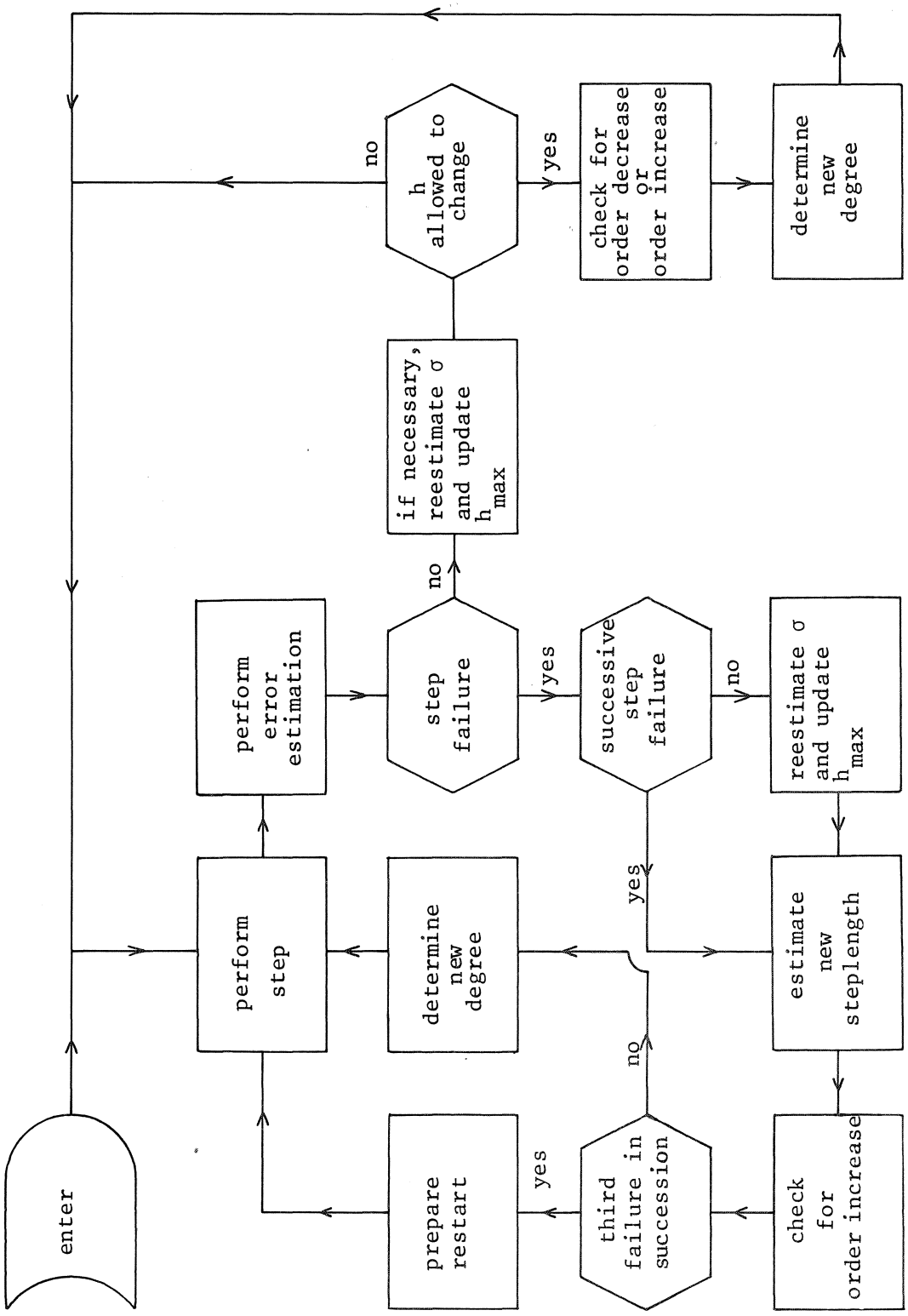

Fig. 3.1 Informal flowchart showing the algorithmic connection between control mechanisms.

routine M3RK, and thus can not stand on their own.

The program integrates a given problem from an initial value of x to a value of x which may be slightly beyond a user specified output point, say x_e. The desired solution at the output point x_e is always interpolated by means of the quadratic formula

$$(4.1) \qquad y(x_e) \simeq \tfrac{1}{2}\mu(\mu-1)y(x-2h) + \mu(2-\mu)y(x-h) + \tfrac{1}{2}(\mu-1)(\mu-2)y(x),$$

where $\mu = (x-x_e) / h$. After a normal return of M3RK the parameters in the call list are ready to continue the integration. This means that when the user decides to continue the integration, he only needs to define a new output point x_e and call again. In fact, M3RK is written in such a way that the choice of output points does not influence the integration process itself.

All information to and from M3RK is passed through its parameters in the call list

```
M3RK (X, XE, N, H, HMIN, SIGMA, TOL, F, Y, Y1, Y2, YXE, DY,
      DY1, IFLAG, INFO)
```

X and XE represent the independent variable x and the output point x_e, respectively. N represents the number of differential equations of the system to be integrated. H an HMIN denote the steplength h and a smallest steplength, respectively.
SIGMA represents the spectral radius σ and TOL is the local tolerance parameter. F is the name of a subroutine defining the differential equations.
Y, Y1, Y2, YXE, DY and DY1 are arrays of length N which represent the solution vector at x, x - h, x - 2h, x_e and the derivative vector at x and x - h, respectively. We prefer the use of 6 arrays of length N instead of one of length 6N for clarity, and in order to avoid the overhead costs of pointers. When using one array this overhead is not negligeable for our class of formulas because of the high degree. A small disadvantage is of course a longer parameter list. IFLAG is an error flag and INFO an integer array of length 15, which is used to initialize the code, to pass information between M3RK and the subprograms, to pass information to the user about the status of

the integration, and finally to retain information for subsequent calls.

For specific information about input requirements and output we refer to the prologue of comments of M3RK, which further explains how to use the program (see appendix). Here we confine ourselves to the observation that for a first call the only input parameters are X, XE, N, TOL, F, Y, INFO(i), i = 1,2,3, while SIGMA is optional. For a subsequent call the only input parameter to be changed is normally XE. The user must always give a maximum for the number of evaluations of f(y) to be spent by M3RK. If this maximum number is reached while $x < x_e$, the process is interrupted and IFLAG is set equal to 1. In this situation the user has the possibility to continue the process in a simple way. The message IFLAG = 0 means that x_e is reached.

To give insight in the structure of the program we give a short list of the names and meanings of the subroutines which are called by M3RK:

HSTART computes the initial steplength according to section 3.1.

PARAM delivers the integration parameters according to expressions (2.6) and (3.1). PARAM contains a data-statement to store the coefficients of the stability polynomials S, P and $\tilde{R}_m^{(2)}$ into an internally declared array of length 440.

POWERM estimates the spectral radius (see section 3.5). If the computation fails IFLAG is set equal to 3.

MAXDEG evaluates the maximal degree m_{max} (cf. (3.12)). If $m_{max} < 2$, IFLAG is set equal to 2. MAXDEG contains a data-statement to store the 11 values given in table 2.1.

MINDEG evaluates the minimal degree satisfying the stability condition (3.3).

STEP contains the actual integrator and performs precisely one integration step with (2.1).

ESTIMA computes the local error bound and estimates the local error (see 3.7)).

NEWH delivers the new steplength and the factor α according to (3.10).

INTER1 performs the interpolation (3.8) and evaluates $f(y(x-\alpha h))$.

INTER2 performs the interpolation (4.1) at the output point x_e.

SHIFT shifts the x- and y-variables and calculates a new derivative to prepare the next integration step.

In order to enlarge the portability of the program and to keep the structure as simple as possible we have avoided the use of common-statements.

As a consequence the subroutines are completely local, i.e. all information they need is passed through the parameter list. On purpose we do not discuss the various parameter lists. The meaning of the parameters should be immediately clear when reading M3RK which is extensively commented. Moreover, the subroutines called by M3RK are short and, as indicated above, directly associated to small parts of section 3. Thus these subroutines are easily verified by inspection. In figure 4.1 a flow chart is given which shows the structure of a complete program comprising a calling program, a user-supplied subroutine F, our main program M3RK and the subroutines called by M3RK. A downward sloping line from one box to another indicates that the lower program is called by the upper one. Observe however that F is always called via a parameter list.

A macroscopic flow chart of M3RK is given in fig. 4.2. In this flow chart k = 1 and k = 3 for a one-step and three-step formula, respectively; r denotes the number of successive rejected steps, n denotes the number of steps performed after start or restart, and s denotes the number of steps performed after an estimation, or check for an estimation of the spectral radius.

There remains to make some comments about the portability of the program. The whole package has been tested on a CDC 73/28 using 14 digits. It has been accepted by the PFORT Verifier (see RYDER [9]). The PFORT Verifier is a program which checks a FORTRAN program for adherence to PFORT, a portable subset of American National Standard FORTRAN. M3RK uses one machine dependent constant, namely the arithmetic precision represented by the internal variable APR. POWERM contains the CDC system subprograms RANSET and RANF, constituting a random generator. RANF is the actual generator, while RANSET initializes the generative value of RANF. It is emphasized that replacing the CDC random generator slightly influences the results given in the next section.

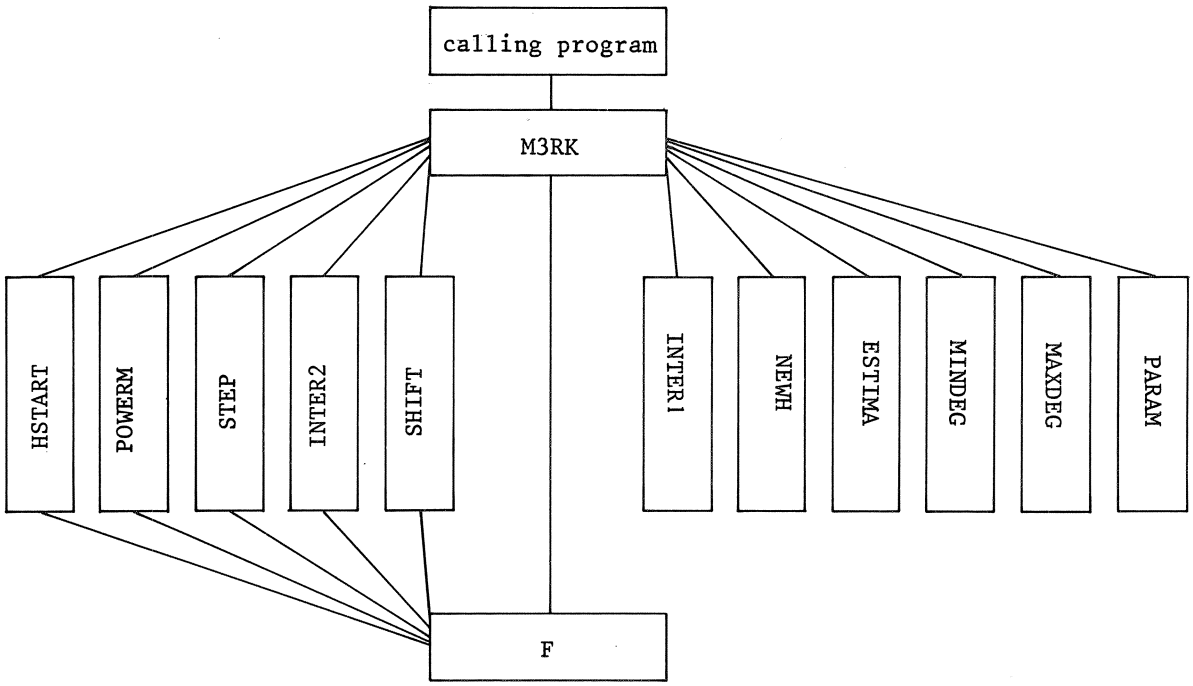

Fig. 4.1 Structure of a complete program

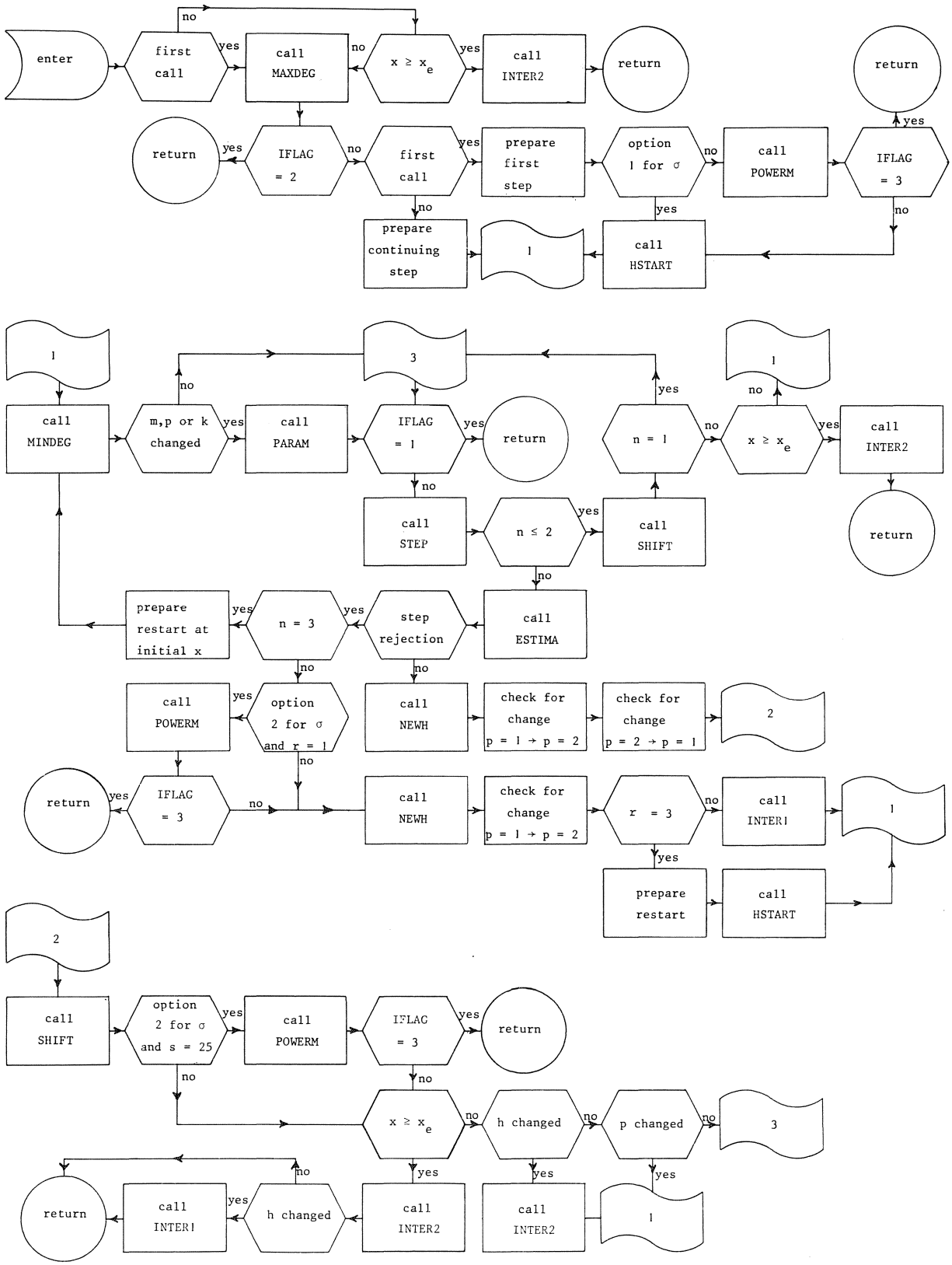

Fig. 4.2 Macroscopic flow chart of M3RK

5. NUMERICAL EXAMPLES

In order to test subroutine M3RK, it was applied to several problems. Two of these problems are discussed in this section. Both problems are non-linear and arise in practice. For both problems the semi-discretization has been performed by means of a continuous time Galerkin method (see e.g. DOUGLAS & DUPONT [4]).

5.1. A ONE-DIMENSIONAL SYSTEM OF TWO NON-LINEAR EQUATIONS

The first problem we consider is a one-dimensional system of two non-linear equations from electricity theory (cf. TE RIELE [12], section 3.2.6):

$$\begin{aligned} \frac{\partial u}{\partial t} &= \varepsilon\rho \frac{\partial^2 u}{\partial x^2} - g(u-v), \\ \frac{\partial v}{\partial t} &= \rho \frac{\partial^2 v}{\partial x^2} + g(u-v), \end{aligned} \tag{5.1}$$

where $0 \le x \le 1$, $g(z) = \exp(\frac{1}{3}\mu z) - \exp(-\frac{2}{3}\mu z)$, $\mu = 17.19$, $\varepsilon = 0.143$, $\rho = 0.1743$. The corresponding initial and boundary conditions read:

$$\begin{aligned} &u = 1, \quad v = 0, \quad 0 \le x \le 1, \quad t = 0, \\ &\frac{\partial u}{\partial x} = v = 0, \quad x = 0, \quad t > 0, \\ &u = 1, \ \frac{\partial v}{\partial x} = 0, \ x = 1, \ t > 0. \end{aligned} \tag{5.2}$$

Equation (5.1) was semi-discretized by means of a Galerkin method based on piecewise quadratic polynomials, and implemented to yield a purely explicit system of ordinary differential equations (cf. BAKKER [1]). It is beyond the scope of this paper to discuss this discretization technique. We confine ourselves therefore to the definition of the semi-discretized system obtained for the equidistant grid $\{x_i \mid x_i = (i-1)/(M-1),\ i = 1,\ldots,M\ ;M \text{ odd}\}$. Let $u_i(t) \approx u(x_i,t)$ and $v_i(t) \approx v(x_i,t)$. The equations for the u_i-componerts then are:

$$\dot{u}_1 = -\tfrac{1}{2}\varepsilon\rho(M-1)^2[7u_1 - 8u_2 + u_3] - g(u_1-v_1),$$

$$\dot{u}_{2i} = -\varepsilon\rho(M-1)^2[2u_{2i} - u_{2i-1} - u_{2i+1}] - g(u_{2i}-v_{2i})$$

$$i = 1,\ldots,\frac{M-1}{2}, \tag{5.3}$$

$$u_{2i+1} = -\tfrac{1}{4}\varepsilon\rho(M-1)^2[14u_{2i+1} - 8(u_{2i+2} + u_{2i}) + u_{2i+3} + u_{2i-1}] -$$

$$- g(u_{2i+1} - v_{2i+1}), \qquad i = 1,\ldots,\frac{M-3}{2},$$

$$\dot{u} = 0.$$

We do not give the equations for the v_i-components, because there are now easy to find.

We integrated two systems, viz. system I and II obtained by setting M = 31 and M = 61, respectively. Both systems were integrated over the interval [0,20] by calling M3RK for XE = 10^{-2} (first call), and for XE = 10^{-1},1,5,10,20 (subsequent calls). They were integrated for 3 values of TOL, viz. 10^{-3},10^{-4} and 10^{-5}. The parameter INFO(2) was chosen equal to 2, indicating the second option for the spectral radius (the specification of the remaining input parameters should be clear from the foregoing).

Results of the integrations are listed in tables 5.1 and 5.2. Table 5.1 gives the system I - approximations, and system II - approximations to u(x,t) for the specified number of output times and some grid values x_i. The approximations were rounded to 4 decimal places. Comparing the results for a system for several values of TOL yields an indication about the accuracy of the time integration. Comparing the results for both systems for several values of TOL yields an indication about the accuracy of the space discretization. From this table we thus have an indication about the accuracy of the approximations to u(x,t), i.e. the solution of one of the components of the partial differential equation, at several times and points.

To get some insight in the course of the time integrations, and thus in the behaviour of M3RK, table 5.2 gives for all integrations for each output time the following information: step = the total number of steps, restep = the number of rejected steps, fev = the total number of f(y)-evaluations, sig = the number of f(y)-evaluations needed for the estimation

and control of the spectral radius, sigma = the estimation of the spectral radius used by M3RK. We observe that for both systems no step rejections occurred. Among others, table 5.2 clearly shows the increase of the stepsize during the process. Because of the fact that the problem possesses a steady state solution, such an increase must occur. From this table we can also calculate the average number of function evaluations per step at the given output times. We also see that the average number of function evaluations decreases as TOL becomes smaller. This is due to the fact that for the present problem the stepsize is mostly restricted by accuracy requirements. If this is the case the degree is minimized. As a consequence, a significantly larger number of steps, due to a smaller value value of TOL, not always yields a significantly larger number of function evaluations. This fact is clearly illustrated by table 5.2. It may thus be preferred to choose TOL not too large. Moreover, the various control mechanisms work better for smaller values of TOL.

We conclude this example by noting that all integrations were performed without the necessity of making restarts.

5.2. A NON-LINEAR, TWO-DIMENSIONAL PROBLEM

The second problem we consider is a two-dimensional diffusion problem:

$$
\begin{aligned}
&\frac{\partial u}{\partial t} = \beta(u)\left[\frac{\partial^2 u}{\partial z^2} + \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right)\right] + \gamma(u), \quad 0 \le r \le r_e,\ 0 \le z \le z_e,\\
&u(0,r,z,) = 500, \quad 0 \le r \le r_e,\ 0 \le z \le z_e,\\
&u(t,r,0) = u(t,r,z_e) = 500, \quad 0 \le r \le r_e,\ t > 0,\\
&u_r(t,0,z) = 0,\ u_r(t,r_e,z) = \mu(u(t,r_e,z)),\ 0 \le z \le z_e,\ t > 0,
\end{aligned}
\tag{5.4}
$$

where $r_e = 10^{-4}$, $z_e = 0.15$, and $\beta(u) = \lambda(u)/\rho(u)$, $\gamma(u) = \eta(u)/\rho(u)$, $\mu(u) = \psi(u)/\lambda(u)$. The functions λ, ρ, η, and ψ are defined by

		system I						system II					
	t \ x	0	0.2	0.4	0.6	0.8	0.9	0	0.2	0.4	0.6	0.8	0.9
TOL = 10^{-3}	10^{-2}	.5283	.6879	.6879	.6879	.6879	.6883	.5498	.6879	.6879	.6879	.6879	.6883
	10^{-1}	.2193	.4742	.5105	.5124	.5242	.5773	.2219	.4743	.5105	.5123	.5225	.5717
	1	.0422	.1988	.3676	.5122	.6516	.7377	.0424	.1981	.3664	.5102	.6489	.7345
	5	.0330	.1636	.3225	.4810	.6396	.7318	.0332	.1626	.3207	.4784	.6364	.7283
	10	.0327	.1623	.3204	.4785	.6375	.7301	.0330	.1617	.3190	.4764	.6347	.7269
	20	.0326	.1623	.3203	.4785	.6375	.7300	.0329	.1617	.3190	.4764	.6347	.7269
TOL = 10^{-4}	10^{-2}	.5271	.6870	.6870	.6870	.6870	.6874	.5488	.6870	.6870	.6870	.6870	.6874
	10^{-1}	.2184	.4738	.5099	.5118	.5236	.5767	.2209	.4739	.5099	.5118	.5219	.5711
	1	.0419	.1981	.3671	.5123	.6521	.7381	.0422	.1977	.3660	.5103	.6492	.7348
	5	.0330	.1637	.3226	.4811	.6398	.7318	.0332	.1629	.3210	.4788	.6368	.7285
	10	.0327	.1624	.3204	.4786	.6375	.7301	.0329	.1617	.3190	.4765	.6348	.7270
	20	.0327	.1623	.3204	.4785	.6375	.7301	.0329	.1617	.3190	.4764	.6347	.7269
TOL = 10^{-5}	10^{-2}	.5268	.6867	.6867	.6867	.6867	.6871	.5485	.6867	.6867	.6867	.6867	.6871
	10^{-1}	.2181	.4737	.5098	.5117	.5234	.5766	.2206	.4738	.5098	.5116	.5217	.5709
	1	.0419	.1979	.3670	.5124	.6523	.7383	.0422	.1975	.3659	.5103	.6493	.7349
	5	.0330	.1637	.3226	.4811	.6397	.7318	.0332	.1630	.3213	.4791	.6370	.7287
	10	.0328	.1624	.3205	.4786	.6376	.7302	.0329	.1617	.3190	.4765	.6348	.7270
	20	.0327	.1623	.3204	.4785	.6375	.7301	.0329	.1617	.3190	.4764	.6347	.7269

Table 5.1 Approximations to u(t,x) of (5.1) at several times and points.

	system I						system II				
	t	step	re-step	fev	sig	sigma	step	re-step	fev	sig	sigma
TOL = 10^{-3}	0	0	0	21	21	4462.2	0	0	21	21	6871.4
	10^{-2}	38	0	127	40	1290.9	38	0	127	40	4352.3
	10^{-1}	58	0	183	45	1290.9	58	0	203	45	4352.3
	1	79	0	307	50	1290.9	80	0	415	50	4352.3
	5	98	0	516	50	1290.9	116	0	832	55	4352.3
	10	113	0	669	55	1290.9	145	0	1185	60	4352.3
	20	149	0	1068	60	1290.9	204	0	1908	75	4352.3
TOL = 10^{-4}	0	0	0	21	21	4462.2	0	0	21	21	6871.4
	10^{-2}	66	0	215	65	1185.0	66	0	194	44	4892.5
	10^{-1}	103	0	308	75	1185.0	104	0	318	54	4892.5
	1	142	0	469	80	1185.0	143	0	614	59	4892.5
	5	171	0	725	85	1185.0	210	0	1266	74	4892.5
	10	195	0	949	90	1185.0	244	0	1675	79	4892.5
	20	213	0	1165	95	1185.0	310	0	2482	94	4892.5
TOL = 10^{-5}	0	0	0	21	21	4462.2	0	0	21	21	6871.4
	10^{-2}	122	0	375	99	1304.2	122	0	344	68	4704.6
	10^{-1}	195	0	554	114	1304.2	196	0	541	83	4704.6
	1	273	0	819	129	1304.2	273	0	972	98	4704.6
	5	323	0	1189	139	1304.2	377	0	1931	123	4704.6
	10	358	0	1508	149	1304.2	445	0	2556	133	4704.6
	20	383	0	1775	154	1304.2	509	0	3339	148	4704.6

Table 5.2 Information about M3RK concerning (5.3).

$$
\begin{aligned}
\lambda(u) &= 418.4 * \{0.1287496_{10}{-13}u^4 - 0.116873_{10}{-9}u^3 + \\
&\quad + 0.384891_{10}{-6}u^2 - 0.569812_{10}{-3}u + 0.57185303\}, \\
\rho(u) &= 19300 * \{0.14644_{10}3 + 0.18513_{10}{-1} * (u-0.104_{10}4)\}, \\
(u) &= 7.1486^2 * \pi^{-2} * 10^{10} * \{-0.378808_{10}{-1} + 0.267688_{10}{-3}u + \\
&\quad + 0.1724588_{10}{-7}u^2\}, \\
(u) &= 0.56696_{10}{-7}u^4 * \{-0.270491_{10}{-17}u^5 + 0.3438691_{10}{-13}u^4 - \\
&\quad - 0.163905_{10}{-9}u^3 + 0.3305647_{10}{-6}u^2 - 0.1194_{10}{-3}u + \\
&\quad + 0.2667112_{10}{-1}\}.
\end{aligned}
\tag{5.5}
$$

Problem (5.4) describes the temperature in a filament and was communicated to us by POLAK [7], who is gratefully acknowledged for his cooperation.

For the semi-discretization of (5.4) we have applied a Galerkin method based on piecewise bilinear functions. We have made use of an implementation, written by BAKKER [2], yielding a purely explicit initial value problem. Again we shall confine ourselves to the definition of the resulting system of ordinary differential equations.

Let $\{r_i | r_1 = 0,\ r_i < r_{i+1},\ i = 1,\ldots,M-1,\ r_M = r_e\}$ and $\{z_j | z_1 = 0,\ z_j < z_{j+1},\ j = 1,\ldots,N-1,\ z_N = z_e\}$ be partitions of the r-axis and z-axis, respectively. Let $p_i(r)$, $i = 1,\ldots,M$, and $q_j(z)$, $j = 1,\ldots,N$, be piecewise linear functions, i.e. $p_i(r_k) = \delta_{ik}$ and $q_j(z_l) = \delta_{jl}$, where δ denotes the Kronecker symbol. Further, let $u_{ij}(t) \simeq u(t,r_i,z_j)$. The resulting system of ordinary differential equations is then defined by

$$
\begin{aligned}
\dot{u}_{ij} &= 0, \qquad i = 1,\ldots,M,\ j = 1,N, \\
\dot{u}_{ij} &= v_i^{-1} w_j^{-1} \beta(u_{ij}) \{r_e w_j \mu(u_{Mj}) \delta_{iM} - a_{ij}u_{ij} - a_{ij-1}u_{ij-1} - \\
&\quad - a_{ij+1}u_{ij+1} - a_{i-1j}u_{i-1j} - a_{i+1j}u_{i+1j}\} + \gamma(u_{ij}), \\
&\quad i = 1,\ldots,M,\quad j = 2,\ldots,N-1,
\end{aligned}
\tag{5.6}
$$

where

$$v_i = \int_0^{r_e} r p_i(r)\,dr, \quad w_j = \int_0^{z_e} q_j(z)\,dz,$$

(5.7)

$$a_{kl} = \int_0^{r_e}\int_0^{z_e} r[\dot{p}_i(r)\dot{p}_k(r)q_j(z)q_l(z) + p_i(r)p_k(r)\,\dot{q}_j(z)\dot{q}_l(z)]\,dr\,dz.$$

Again M3RK was applied to two systems. For both systems $r_i = (i-1)r_e/(M-1)$, $i = 1,\ldots,M$, and $z_j = 0.06*(j-1)/(N-1)$, $j = 1,\ldots,(N-1)/4$; $z_j = z_{j-1} + 0.24/(N-1)$, $j = (N+3)/4,\ldots,(3N+1)/4$; $z_j = z_{j-1} + 0.06/(N-1)$, $j = (3N+5)/4,\ldots,N$. The choice M=5, N=21 and M=5, N=41 yields system I and system II, respectively. Because of the physical nature of the problem it is not necessary to choose $M > 5$. A finer grid at the endpoints of the filament is necessary in order to deal with boundary layers. It should be observed that, because of two reasons, the resulting systems are difficult problems for our explicit integrator. Firstly, the systems are strongly non-linear. Secondly,in spite of the relatively small number of gridpoints, we have to deal with a large spectral radius because of the very small size of the r-interval.

Both systems were integrated over the interval [0,1] by calling M3RK for XE = 0.1 (first call), and for XE = 0.2,0.4,0.6,0.8,0.9 and 1 (subsequent calls). They were integrated for 3 values of TOL, viz. 10^{-3},10^{-4} and 10^{-5}. Again, the parameter INFO(2) was chosen equal to 2.

Some results of the integrations are listed in tables 5.3 and 5.4. Table 5.3 gives the system I-approximations and system II-approximations to $u_{1j}(t)$ at the specified output times for some values of $z_j \in [0,z_e/2]$. Results for $z \in [z_e/2,z_e]$ are omitted, as the solution is approximately symmetric with respect to $z_e/2$. With respect to comparing and interpreting results of table 5.3, we refer to the remarks made at the preceding example.

Table 5.4 yields the same type of information as table 5.2.The rejections clearly indicate that the guidance of the process is better for smaller values of TOL. Because of this, and because of the minimizing property of the degree, we again conclude that it may be preferred to chose TOL not too large. This conclusion particularly applies when integrating stongly non-linear problems.

We conclude this section by observing that all integrations were performed without the necessity of making restarts.

		system I					system II				
	t \ z	0.003	0.006	0.009	0.015	0.075	0.003	0.006	0.009	0.015	0.075
TOL = 10^{-3}	0.1	700.8	748.9	757.1	758.3	758.4	706.0	751.4	757.1	757.6	757.5
	0.2	951.3	1123.6	1172.4	1185.2	1185.2	955.7	1129.9	1174.5	1183.8	1183.9
	0.4	1746.2	2316.1	2514.2	2593.4	2596.3	1736.1	2322.0	2520.3	2591.1	2593.5
	0.6	2551.3	3154.0	3284.7	3323.9	3325.4	2508.5	3156.7	3288.8	3324.0	3325.5
	0.8	2816.7	3304.4	3382.0	3397.2	3397.6	2755.4	3308.0	3384.3	3396.9	3397.2
	0.9	2847.8	3317.5	3388.2	3400.7	3400.9	2785.6	3321.9	3390.8	3400.6	3400.8
	1.0	2858.8	3321.9	3390.1	3401.5	3401.6	2797.2	3326.7	3392.7	3401.5	3401.6
TOL = 10^{-4}	0.1	700.8	748.9	757.1	758.3	758.4	706.1	752.0	757.8	758.3	758.3
	0.2	952.1	1123.9	1172.4	1185.6	1185.8	956.3	1131.0	1176.0	1185.6	1185.6
	0.4	1746.5	2316.9	2514.8	2594.0	2596.9	1737.4	2323.8	2522.4	2593.6	2595.9
	0.6	2552.1	3153.9	3285.3	3324.4	3325.9	2509.6	3157.7	3289.6	3324.7	3326.2
	0.8	2815.3	3303.7	3381.6	3397.0	3397.3	2756.0	3308.8	3384.8	3397.2	3397.6
	0.9	2847.2	3317.3	3388.1	3400.6	3400.8	2786.4	3322.2	3391.0	3400.7	3400.9
	1.0	2858.6	3321.9	3390.1	3401.5	3401.6	2797.4	3326.8	3392.8	3401.5	3401.6
TOL = 10^{-5}	0.1	701.0	749.5	758.0	759.3	759.3	706.3	752.6	758.7	759.3	759.3
	0.2	952.8	1125.3	1174.2	1187.7	1187.9	957.0	1132.5	1178.0	1187.8	1187.9
	0.4	1748.9	2319.8	2517.7	2597.1	2600.0	1740.0	2327.4	2526.2	2597.6	2600.0
	0.6	2552.1	3153.3	3284.7	3323.7	3325.2	2509.4	3156.9	3288.6	3323.7	3325.2
	0.8	2815.3	3303.5	3381.5	3396.9	3397.2	2755.6	3308.1	3384.3	3396.9	3397.2
	0.9	2846.7	3317.0	3388.0	3400.6	3400.8	2785.7	3321.8	3390.8	3400.6	3400.8
	1.0	2858.3	3321.7	3390.0	3401.5	3401.6	2797.1	3326.7	3392.7	3401.5	3401.6

Table 5.3 Approximations to u(t,0,z) of (5.6) at several times and points.

	system I						system II				
	t	step	re-step	fev	sig	sigma	step	re-step	fev	sig	sigma
TOL = 10^{-3}	0	0	0	11	11	433026.3	0	0	11	11	433627.5
	0.1	146	5	1595	92	361136.0	114	1	1241	73	387114.9
	0.2	275	8	3043	191	314911.0	245	6	2759	206	353507.5
	0.4	483	14	5418	377	250766.5	460	10	5172	340	291676.3
	0.6	666	21	7466	543	209500.5	652	15	7306	468	253444.2
	0.8	767	22	8593	593	195533.7	865	24	9578	615	243312.1
	0.9	803	22	8992	598	195533.7	904	24	10042	625	243312.1
	1.0	830	22	9321	603	195533.7	951	25	10600	648	240319.6
TOL = 10^{-4}	0	0	0	11	11	433026.3	0	0	11	11	433627.5
	0.1	176	0	1548	53	381046.8	176	0	1541	54	383557.4
	0.2	327	0	3059	142	315213.0	336	0	3108	130	353077.8
	0.4	573	1	5395	238	260536.2	617	3	5714	271	286028.2
	0.6	776	1	7402	311	210908.2	854	5	7915	367	257379.4
	0.8	984	9	9437	560	193543.2	1003	6	9373	410	244102.1
	0.9	1026	9	9877	565	193543.2	1038	6	9803	420	244102.1
	1.0	1053	9	10206	570	193543.2	1098	7	10413	443	240106.2
TOL = 10^{-5}	0	0	0	11	11	433026.3	0	0	11	11	433627.5
	0.1	237	0	2131	63	388688.4	237	0	2131	63	390626.6
	0.2	425	0	3902	140	327591.8	440	0	4049	152	357651.3
	0.4	739	0	6858	267	249339.1	792	0	7291	257	306499.2
	0.6	996	0	9237	332	233658.6	1088	0	10047	347	259726.0
	0.8	1231	0	11294	382	233658.6	1337	0	12198	397	259726.0
	0.9	1299	0	11901	392	233658.6	1403	0	12831	412	259726.0
	1.0	1340	0	12372	402	233658.6	1444	0	13315	417	259726.0

Table 5.4 Information about M3RK concerning (5.6)

6. CONCLUDING REMARKS

The purpose of this paper was to develop a robust and efficient time integrator, being easy to apply to a wide class of linear, and in particular non-linear, semi-discretized parabolic problems in one and more dimensions. Although it is not clear that, e.g. for a two-dimensional non-linear problem, our method needs less computer time than an unconditionally stable method (such as an implicit method or an ADI method), the easy applicability of the subroutine, when used in conjunction with semi-discretization, reduces the human effort needed to solve a problem under consideration. For example, the subroutine is easy to use with a software interface performing the semi-discretization (see e.g. SINCOVEC & MADSEN [11] or BAKKER [2]).

It is of interest to observe that until now stabilized, explicit methods, as well as their implementations, did not receive very much attention in literature. As a consequence, it is most likely that the present implementation and its underlying algorithm can be improved significantly.

Acknowledgement

The author wishes to thank Professor P.J. van der Houwen and Professor T.J. Dekker of the University of Amsterdam for their constructive comments and careful reading of the manuscript. The author is also grateful to Mr. M. Bakker for his assistance in programming the Galerkin discretizations, and to Mr. B. Sommeijer for his assistance in testing the program.

REFERENCES

[1] BAKKER, M., *Software for semi-discretization of time dependent partial differential equations in one space variable*, Report NW, Mathematisch Centrum, Amsterdam, (to appear).

[2] BAKKER, M., *Unpublished manuscript*, Mathematisch Centrum, Amsterdam.

[3] BEENTJES, P.A., *NUMAL, a library of ALGOL 60 procedures in numerical mathematics*, section 5.2.1.1.1.1., procedure ARK, Mathematisch Centrum, Amsterdam, 1974.

[4] DOUGLAS, J. & T. DUPONT, *Galerkin methods for parabolic equations*, SIAM J. Numer. Anal. 7, pp. 575-626, 1970.

[5] GEAR, C.W. *Numerical initial value problems in ordinary differential equations*, Prentice Hall, Englewood Cliffs, New Yersey, 1971.

[6] LINDBERG, B., *IMPEX - A program package for solution of systems of stiff differential equations*, Report NA 72.50, The Royal Institute of Technology, Stockholm, 1972.

[7] POLAK, S.J., *Private communication*, ISA, Gebouw VM, Philips, Eindhoven, The Netherlands.

[8] RICHTMEYER, R.D. & K.W. MORTON, *Difference methodes for initial value problems*, Interscience, New York, 1967.

[9] RYDER, B.G., *The PFORT verifier*, *Software Practice and Experience*, 4, pp. 359-378, 1974.

[10] SHAMPINE, L. & M.K. GORDON, *Computer solution of ordinary differential equations*, The initial value problem, W.H. Freeman and Co., San Francisco, 1975.

[11] SINCOVEC, R.F. & N.K. MADSEN, *Software for non-linear partial differential equations*, ACM Transactions on mathematical software 1, pp. 232-260, 1975.

[12] TE RIELE, H.J.J., (ed.), *Colloquium Numerieke Programmatuur* (Dutch), MC syllabus, Mathematisch Centrum, Amsterdam, (to appear).

[13] VAN DER HOUWEN, P.J., *Construction of integration formulas for initial*

value problems, North-Holland Publishing Company, Amsterdam, 1976.

[14] VERWER, J.G., *A class of stabilized three-step Runge-Kutta methods for the numerical integration of parabolic equations*, Journal of Computational and Applied Mathematics, (to appear).

[15] VERWER, J.G., *Multipoint multistep Runge-Kutta methods I: On a class of two-step methods for parabolic equations*, Report NW30/76, Mathematisch Centrum, Amsterdam, 1976.

[16] VERWER, J.G., *Multipoint multistep Runge-Kutta methods II: The construction of a class of stabilized three-step methods for parabolic equations*, Report NW31/76, Mathematisch Centrum, Amsterdam, 1976.

```
      SUBROUTINE M3RK(X,XE,N,H,HMIN,SIGMA,TOL,F,Y,
     +                Y1,Y2,YXE,DY,DY1,IFLAG,INFO)
C*********************************************************************
C* ABSTRACT                                                          *
C*********************************************************************
C* M3RK IS DESIGNED TO SOLVE INITIAL VALUE PROBLEMS FOR SYSTEMS OF   *
C* ORDINARY DIFFERENTIAL EQUATIONS OF THE FORM                       *
C*                    DY/DX=F(Y(1),...,Y(N)),                        *
C*                    Y(I) GIVEN AT X,                               *
C* WHICH ORIGINATE FROM SEMI-DISCRETIZATION OF INITIAL-BOUNDARY VALUE*
C* PROBLEMS FOR PARABOLIC PARTIAL DIFFERENTIAL EQUATIONS. MR3K IS    *
C* BASED ON STABILIZED,EXPLICIT THREE-STEP RUNGE-KUTTA FORMULAS OF   *
C* ORDER ONE AND TWO,OF WHICH THE DEGREE CAN VARY BETWEEN 2 AND 12.  *
C*                                                                   *
C* M3RK NEEDS 6 ARRAYS OF LENGTH N,WHICH ALL APPEAR IN THE CALL LIST.*
C* THE CODE INTEGRATES FROM X TO XE.ON NORMAL RETURN THE PARAMETERS IN*
C* THE CALL LIST ARE READY FOR CONTINUING THE INTEGRATION,TO CONTINUE*
C* THE INTEGRATION,THE USER NEEDS ONLY TO REDEFINE THE OUTPUT POINT XE*
C* AND CALL AGAIN.                                                   *
C*                                                                   *
C* M3RK CALLS 11 SUBROUTINES WHICH HAVE BEEN WRITTEN TO STRUCTURE THE*
C* PROGRAM.THESE SUBROUTINES ARE:                                    *
C* HSTART = HSTART COMPUTES THE INITIAL STEPLENGTH                   *
C* PARAM  = PARAM COMPUTES PARAMETERS OF THE VARIOUS IMPLEMENTED     *
C*          SCHEMES FROM THE COEFFICIENTS OF THE STABILITY POLYNOMIALS*
C* POWERM = POWERM ESTIMATES THE SPECTRAL RADIUS OF THE JACOBIAN OF F*
C* MAXDEG = MAXDEG COMPUTES THE MAXIMAL DEGREE OF THE FORMULAS,      *
C*          WHICH IS ALLOWED WITH RESPECT TO INTERNAL STABILITY      *
C* MINDEG = MINDEG COMPUTES THE MINIMAL DEGREE OF THE FORMULAS,      *
C*          WHICH IS ALLOWED WITH RESPECT TO ABSOLUTE STABILITY      *
C* STEP   = STEP CONTAINS THE ACTUAL INTEGRATOR                      *
C* ESTIMA = ESTIMA COMPUTES A LOCAL ERROR BOUND AND ESTIMATES A LOCAL*
C*          ERROR                                                    *
C* NEWH   = NEWH DELIVERS A NEW STEPLENGTH                           *
C* INTER1 = INTER1 PERFORMS INTERPOLATION AFTER A CHANGE OF THE      *
C*          STEPLENGTH                                               *
C* INTER2 = INTER2 INTERPOLATES THE SOLUTION AT THE OUTPUT POINT XE  *
C* SHIFT  = SHIFT SHIFTS THE DATA FOR A NEXT STEP                    *
C*                                                                   *
C* THESE SUBROUTINES ARE COMPLETELY LOCAL,I.E.THE INFORMATION THEY   *
C* NEED IS PASSED THROUGH THE PARAMETER LISTS.THE WHOLE PACKAGE HAS  *
C* BEEN TESTED ON A CDC CYBER 73-28 USING AN ARITHMETIC PRECISION OF *
C* 14 DIGITS.THE CODE POWERM USES THE CDC SYSTEM SUBPROGRAMS RANSET  *
C* AND RANF CONSTITUTING A RANDOM GENERATOR.RANSET AND RANF MUST BE  *
C* REPLACED WHEN USING THE PROGRAM ON ANOTHER COMPUTER.              *
C* THE CODES CALLED BY M3RK USE NO MACHINE DEPENDENT CONSTANTS.      *
C* M3RK USES ONE MACHINE DEPENDENT CONSTANT,NAMELY THE ARITHMETIC    *
C* PRECISION OF THE COMPUTER.IN THE PROGRAM THE INTERNAL VARIABLE APR,*
C* WHICH REPRESENTS THE ARITHMETIC PRECISION,EQUALS 1.0E-14. APR MUST*
C* BE CHANGED ACCORDINGLY WHEN USING THE PROGRAM ON ANOTHER COMPUTER.*
C*                                                                   *
C* THE WHOLE PROGRAM PACKAGE IS ACCEPTED BY THE PFORT VERIFIER.THE   *
C* PFORT VERIFIER IS A PROGRAM WHICH CHECKS A FORTRAN PROGRAM,I.E. A  *
C* MAIN PROGRAM AND SUBPROGRAMS,FOR ADHERENCE TO PFORT,A PORTABLE    *
C* SUBSET OF AMERICAN NATIONAL STANDARD FORTRAN(SEE[1]).THE WHOLE     *
C* PROGRAM PACKAGE IS COMPLETELY EXPLAINED AND DESCRIBED IN [2].     *
C*                                                                   *
```

```
C* [1] RYDER,B.G.,THE PFORT VERIFIER,SOFTWARE PRACTICE AND EXPERIENCE, *
C*     VOL.4,PP.359-378,1974.                                          *
C* [2] VERNER,J.G.,AN IMPLEMENTATION OF A CLASS OF STABILIZED,EXPLICIT *
C*     METHODS FOR THE TIME INTEGRATION OF PARABOLIC EQUATIONS,(TO      *
C*     APPEAR).                                                        *
C*******************************************************************
C* MEANING OF THE PARAMETERS                                         *
C*******************************************************************
C* X      - VARIABLE     : INDEPENDENT VARIABLE                      *
C* XE     - EXPRESSION   : OUTPUT POINT AT WHICH SOLUTION IS DESIRED *
C* N      - EXPRESSION   : NUMBER OF EQUATIONS                       *
C* H      - VARIABLE     : STEPLENGTH                                *
C* HMIN   - VARIABLE     : MINIMAL STEPLENGTH                        *
C* SIGMA  - ARRAY        : AN ARRAY OF LENGTH 2 CONTAINING ESTIMATES *
C*                         OF THE SPECTRAL RADIUS OF THE JACOBIAN OF F*
C* TOL    - EXPRESSION   : LOCAL ERROR TOLERANCE                     *
C* F      - SUBROUTINE   : DERIVATIVE                                *
C* Y      - ARRAY        : SOLUTION VECTOR AT X,INPUT AND WORK ARRAY *
C* Y1     - ARRAY        : SOLUTION VECTOR AT X+H,WORK ARRAY         *
C* Y2     - ARRAY        : SOLUTION VECTOR AT X+2H,WORK ARRAY        *
C* YXE    - ARRAY        : SOLUTION VECTOR AT XE,OUTPUT AND WORK ARRAY*
C* DY     - ARRAY        : DERIVATIVE VECTOR AT X,WORK ARRAY         *
C* DY1    - ARRAY        : DERIVATIVE VECTOR AT X+H,WORK ARRAY       *
C* IFLAG  - VARIABLE     : ERROR FLAG                                *
C* INFO   - ARRAY        : INTEGER ARRAY OF LENGTH 15,INFO IS USED TO*
C*                         PASS INFORMATION TO INITIALIZE THE CODE,TO*
C*                         PASS INFORMATION BETWEEN THE MAIN PROGRAM AND*
C*                         THE SUBPROGRAMS,TO PASS INFORMATION TO THE*
C*                         USER ABOUT THE STATUS OF THE INTEGRATION,AND*
C*                         TO RETAIN INFORMATION FOR SUBSEQUENT CALLS.*
C*******************************************************************
C* FIRST CALL TO M3RK                                                *
C*******************************************************************
C* THE USER MUST PROVIDE STORAGE IN HIS CALLING PROGRAM FOR THE ARRAYS *
C* IN THE CALL LIST.HE HAS TO SUPPLY THE SUBROUTINE F(N,Y) FOR       *
C* EVALUATING THE DERIVATIVES DY(I)/DT,I=1,...,N,WHICH MUST BE OVER- *
C* WRITTEN ON Y(I).FOR THE SPECTRAL RADIUS THERE EXIST THREE OPTIONS  *
C* WHICH MUST BE SELECTED WITH INFO(2).IN INFO(3) THE USER MUST GIVE  *
C* A MAXIMUM FOR THE NUMBER OF EVALUATIONS OF F(Y) TO BE SPENT(ON RE- *
C* TURN IT MAY OCCUR THAT INFO(3) IS EXCEEDED WITH ABOUT 50)          *
C*************************                                          *
C* INPUT PARAMETERS                                                  *
C*************************                                          *
C* X      - INITIAL VALUE OF THE INDEPENDENT VARIABLE                *
C* XE     - OUTPUT POINT AT WHICH SOLUTION IS DESIRED                *
C* N      - NUMBER OF EQUATIONS                                      *
C* SIGMA  - (1)= AN UPPER ESTIMATION OF THE SPECTRAL RADIUS OF THE JA-*
C*              COBIAN OF F IN CASE OF OPTION 1.FOR OPTION 2 AND 3 NO *
C*              INITIALIZATION IS REQUIRED                            *
C* TOL    - LOCAL ERROR TOLERANCE                                    *
C* Y      - VECTOR OF INITIAL VALUES OF THE DEPENDENT VARIABLES      *
C* INFO   - (1)= 0 TO INDICATE FIRST CALL                            *
C*          (2)= 1 TO INDICATE OPTION 1 FOR THE SPECTRAL RADIUS,I.E. *
C*                 THE USER MUST INITIALIZE SIGMA(1)                 *
C*             = 2 TO INDICATE OPTION 2 FOR THE SPECTRAL RADIUS,I.E. *
C*                 THE CODE INITIALIZES SIGMA(1) AND CONTROLS SIGMA(1)*
C*             = 3 TO INDICATE OPTION 3 FOR THE SPECTRAL RADIUS,I.E. *
```

```
C*                  THE CODE ONLY INITIALIZES SIGMA(1) AT THE FIRST CALL *
C*          (3)= MAXIMUM NUMBER OF F(Y) EVALUATIONS TO BE SPENT          *
C************************                                               *
C* OUTPUT PARAMETERS                                                    *
C************************                                               *
C* X      - LAST POINT REACHED IN INTEGRATION,NORMALLY X IS SLIGHTLY    *
C*          BEYOND XE                                                   *
C* H      - INITIAL STEPLENGTH FOR SUBSEQUENT CALL                      *
C* HMIN   - MINIMAL STEPLENGTH USED BY M3RK                             *
C* SIGMA  - (1)= UPPER ESTIMATION OF THE SPECTRAL RADIUS INITIALIZED BY *
C*              THE USER OR BY POWERM                                   *
C*        - (2)= INACCURATE ESTIMATION OF THE SPECTRAL RADIUS USED FOR  *
C*              ITS CONTROL                                             *
C* Y      - SOLUTION VECTOR AT X                                        *
C* Y1     - SOLUTION VECTOR AT X-H                                      *
C* Y2     - SOLUTION VECTOR AT X-2H                                     *
C* YXE    - SOLUTION VECTOR AT OUTPUT POINT XE                          *
C* DY     - DERIVATIVE VECTOR AT X                                      *
C* DY1    - DERIVATIVE VECTOR AT X-H                                    *
C* IFLAG  - = 0 NORMAL RETURN,I.E.OUTPUT POINT IS REACHED               *
C*          = 1 OUTPUT POINT IS NOT REACHED,THE MAXIMUM NUMBER OF       *
C*              F(Y)-EVALUATIONS HAS BEEN SPENT,THE PROCESS CAN BE CON- *
C*              TINUED BY INCREASING INFO(3) AND CALLING AGAIN          *
C*          = 2 MAXIMAL DEGREE FALLS OUTSIDE THE RANGE AS TOL/APR IS    *
C*              TOO SMALL,THE PROCESS IS NOT STARTED                    *
C*          = 3 POWERM FAILED IN THE ESTIMATION OF THE SPECTRAL RADIUS, *
C*              THE PROCESS IS DISCONTINUED                             *
C* INFO   - (1) = 1 TO INDICATE THAT THE NEXT CALL IS A SUBSEQUENT ONE  *
C*          (2) = 1 IN CASE OF OPTION 1 OR 3,ELSE 2                     *
C*          (3) = UNCHANGED                                             *
C*          (4) = TOTAL NUMBER OF INTEGRATION STEPS PERFORMED,I.E.       *
C*                ACCEPTED AND REJECTED ONES                            *
C*          (5) = NUMBER OF REJECTED INTEGRATION STEPS                  *
C*          (6) = NUMBER OF RESTARTS INITIATED BY THE CODE              *
C*          (7) = TOTAL NUMBER OF DERIVATIVE EVALUATIONS                *
C*          (8) = NUMBER OF DERIVATIVE EVALUATIONS USED FOR THE ESTIMA- *
C*                TION AND CONTROL OF THE SPECTRAL RADIUS               *
C*          (9) = CURRENT DEGREE                                        *
C*          (10)= MAXIMAL DEGREE FOR THE FIRST ORDER FORMULAS           *
C*          (11)= MAXIMAL DEGREE FOR THE SECOND ORDER FORMULAS          *
C*          (12)= CURRENT ORDER                                         *
C*          (13)= NUMBER OF STEPS PERFORMED AFTER START OR RESTART      *
C*          (14)= NUMBER OF STEPS PERFORMED AFTER CHANGE OF H OR ORDER  *
C*          (15)= NUMBER OF STEPS PERFORMED AFTER ESTIMATION OF SPEC-   *
C*                TRAL RADIUS                                           *
C*                                                                      *
C* IT IS EMPHASIZED THAT THE OUTPUT LIST GIVEN ABOVE IS NOT STRICTLY    *
C* VALID WHEN ON RETURN IFLAG IS EQUAL TO 2 OR 3.                       *
C* IT IS FURTHER EMPHASIZED THAT ON RETURN THE ARRAY Y ALWAYS CONTAINS  *
C* THE SOLUTION VECTOR AT THE POINT X,THUS WHEN IFLAG=3,THE USER HAS    *
C* THE POSSIBILITY TO RESTART THE PROCESS AT THE POINT X,PROVIDED HE    *
C* IS ABLE TO TAKE ACTION WITH RESPECT TO THE ESTIMATION OF THE SPEC-   *
C* TRAL RADIUS.                                                         *
C************************************************************************
C* SUBSEQUENT CALLS TO M3RK                                             *
C************************************************************************
C* ON RETURN OF M3RK ALL PARAMETERS ARE READY FOR CONTINUING THE INTE-  *
```

```
C* GRATION,PROVIDED IFLAG IS NOT EQUAL TO 2 OR 3,IF XE IS REACHED AND  *
C* A NORMAL CONTINUATION IS DESIRED,THE USER NEED ONLY TO DEFINE A NEW *
C* OUTPUT POINT XE AND CALL AGAIN.IF ON RETURN IFLAG=1 AND THE USER    *
C* WANTS TO CONTINUE,HE ONLY NEEDS TO INCREASE INFO(3) AND CALL AGAIN. *
C* THE PROGRAM IS WRITTEN IN SUCH A WAY THAT THE CHOICE OF OUTPUT      *
C* POINTS DOES NOT AFFECT THE INTEGRATION PROCESS ITSELF.BETWEEN SUB-  *
C* SEQUENT CALLS THE USER MAY INCREASE TOL.ALL OTHER PARAMETERS MUST   *
C* REMAIN UNCHANGED.THE COUNTERS IN INFO ARE USED ACCUMULATIVELY.      *
C*********************************************************************
C* PROGRAM TEXT                                                      *
C*********************************************************************
      DIMENSION Y(N),Y1(N),Y2(N),YXE(N),DY(N),DY1(N),SIGMA(2),INFO(15)
C************************************                                 *
C* MEANING OF THE INTERNAL VARIABLES                                 *
C************************************                                 *
C* ALFA   - THE FACTOR FOR CHANGING THE STEPLENGTH                   *
C* APR    - THE ARITHMETIC PRECISION                                 *
C* HOLD   - PREVIOUSLY ACCEPTED STEPLENGTH                           *
C* KOLD   - PREVIOUSLY USED DEGREE                                   *
C* REJECT - NUMBER OF SUCCESSIVE STEP FAILURES                       *
C* B      - NONZERO B-PARAMETERS OF THE SCHEME                       *
C* C      - C-PARAMETERS OF THE SCHEME                               *
C* LA     - LAMDA-PARAMETERS OF THE SCHEME                           *
C* HMAX   - MAXIMAL STEPSIZES WITH RESPECT TO ABSOLUTE STABILITY     *
C* EPS    - LOCAL ERROR BOUND                                        *
C* ERROR  - ESTIMATED LOCAL ERROR                                    *
C*********************************************************************
      INTEGER REJECT
      REAL LA
      DIMENSION B(2),C(12),LA(12),HMAX(2)
C*********************************************************************
C* IF ALREADY PAST OUTPUT POINT DURING A SUBSEQUENT CALL,THEN INTER-  *
C* POLATE AND RETURN                                                 *
C*********************************************************************
      IF(INFO(1).EQ.0.OR.X.LT.XE) GOTO 10
      CALL INTER2(N,Y,Y1,Y2,YXE,(X-XE)/H)
      RETURN
C*********************************************************************
C* SET THE ERROR FLAG IFLAG EQUAL TO ZERO AND INITIALIZE APR,DETERMINE *
C* THE MAXIMAL DEGREES WITH RESPECT TO INTERNAL STABILITY,IF NECESSARY *
C* INTERRUPT                                                         *
C*********************************************************************
   10 IFLAG=0
      APR=1.0E-14
      CALL MAXDEG(TOL,APR,IFLAG,INFO)
      IF(IFLAG.NE.2) GOTO 20
      RETURN
C*********************************************************************
C* SET THE CONTROL VARIABLES REJECT AND HOLD FOR A CONTINUING CALL,AND *
C* INITIALIZE THE ARRAY HMAX                                         *
C*********************************************************************
   20 IF(INFO(1).EQ.0) GOTO 30
      REJECT=0
      HOLD=H
      HMAX(1)=5.15*FLOAT(INFO(10))*FLOAT(INFO(10))/SIGMA(1)
      HMAX(2)=2.29*FLOAT(INFO(11))*FLOAT(INFO(11))/SIGMA(1)
      GOTO 80
```

```
C*********************************************************************
C* SET REJECT,HOLD AND H FOR THE FIRST CALL, SET THE ELEMENTS INFO(I), *
C* I=4,...,9,13,14,15 EQUAL TO ZERO AND INFO(12) EQUAL TO 2,TO PREPARE *
C* THE FIRST STEP EVALUATE F(Y) AND SUBSTITUTE INTO DY,IF NECESSARY    *
C* ESTIMATE THE SPECTRAL RADIUS AND CHECK FOR A FAILURE OF THE POWER   *
C* METHOD.INITIALIZE ARRAY HMAX AND ESTIMATE THE INITIAL STEPLENGTH    *
C*********************************************************************
   30 INFO(1)=1
      REJECT=0
      DO 40 I=4,9
   40 INFO(I)=0
      DO 50 I=13,15
   50 INFO(I)=0
      INFO(12)=2
      DO 60 I=1,N
      DY(I)=Y(I)
      DY1(I)=0.
      Y1(I)=0.
      Y2(I)=0.
   60 CONTINUE
      CALL F(N,DY)
      INFO(7)=INFO(7)+1
      IF(INFO(2).EQ.1) GOTO 70
      SIGMA(1)=0.
      CALL POWERM(N,Y,Y1,YXE,DY,DY1,F,SIGMA,APR,IFLAG,INFO)
      IF(IFLAG.NE.3) GOTO 70
      RETURN
C
   70 HMAX(1)=5.15*FLOAT(INFO(10))*FLOAT(INFO(10))/SIGMA(1)
      HMAX(2)=2.29*FLOAT(INFO(11))*FLOAT(INFO(11))/SIGMA(1)
      CALL HSTART(N,Y,DY,YXE,F,TOL,APR,SIGMA(1),HMIN,INFO)
      H=HMIN
      HOLD=H
C*********************************************************************
C* DETERMINE THE DEGREE,AND,IF NECESSARY,CALCULATE THE PARAMETERS OF   *
C* THE SCHEME TO BE USED                                               *
C*********************************************************************
   80 HOLD=INFO(9)
      CALL MINDEG(N,SIGMA(1),INFO)
      IF(INFO(9).EQ.HOLD) GOTO 90
      CALL PARAM(C,LA,B,INFO)
C*********************************************************************
C* CHECK IF THE MAXIMUM NUMBER OF EVALUATIONS IS REACHED,UPDATE HMIN   *
C* AND CALCULATE A SOLUTION AT X+H                                     *
C*********************************************************************
   90 IF(INFO(7).GE.INFO(3)) IFLAG=1
      IF(IFLAG.NE.1) GOTO 100
      RETURN
C
  100 IF(H.LT.HMIN) HMIN=H
      CALL STEP(N,Y,Y1,Y2,YXE,DY,DY1,H,F,C,LA,B,INFO)
      INFO(13)=INFO(13)+1
      INFO(4)=INFO(4)+1
C*********************************************************************
C* IF THE PROCESS IS IN THE START PHASE,SHIFT THE DATA,CHECK FOR THE    *
C* CONTINUATION WITH A THREE-STEP SCHEME,IF THE OUTPUT POINT IS PASSED *
C* INTERPOLATE AND RETURN                                              *
```

```
C***********************************************************************
      IF(INFO(13).GE.3) GOTO 110
      CALL SHIFT(N,Y,Y1,Y2,YXE,DY,DY1,X,HOLD,F,INFO)
      INFO(14)=INFO(14)+1
      INFO(15)=INFO(15)+1
      IF(INFO(13).EQ.1) GOTO 90
      INFO(9)=0
      IF(X.LT.XE) GOTO 80
      CALL INTER2(N,Y,Y1,Y2,YXE,(X-XE)/HOLD)
      RETURN
C***********************************************************************
C* CALCULATE THE LOCAL ERRORBOUND AND ESTIMATE THE LOCAL ERROR.        *
C***********************************************************************
  110 CALL ESTIMA(N,Y,Y1,Y2,YXE,TOL,EPS,ERROR,INFO)
C***********************************************************************
C* IF THE ERROR IS TOO LARGE FOR THE THIRD STEP AFTER START,THEN RE-   *
C* START AT INITIAL POINT WITH H=H/10                                  *
C***********************************************************************
      IF(EPS.GE.ERROR.OR.INFO(13).NE.3) GOTO 130
      INFO(6)=INFO(6)+1
      INFO(5)=INFO(5)+3
      INFO(9)=0
      INFO(13)=0
      INFO(14)=0
      INFO(15)=0
      X=X-2.*H
      H=H/10.
      DO 120 I=1,N
      Y(I)=Y2(I)
      DY(I)=Y(I)
  120 CONTINUE
      CALL F(N,DY)
      INFO(7)=INFO(7)+1
      GOTO 80
C***********************************************************************
C* IF STEP FAILED,CHECK FOR A REESTIMATION OF THE SPECTRAL RADIUS,IF    *
C* NECESSARY,CHECK FOR A FAILURE OF POWERM AND UPDATE HMAX             *
C***********************************************************************
  130 IF(INFO(2).EQ.1.OR.INFO(15).EQ.0.OR.EPS.GE.ERROR) GOTO 150
      SIGMA(1)=0.
      CALL POWERM(N,Y,Y1,YXE,DY,DY1,F,SIGMA,APR,IFLAG,INFO)
      IF(IFLAG.NE.3) GOTO 140
      RETURN
C
  140 HMAX(1)=5.15*FLOAT(INFO(10))*FLOAT(INFO(10))/SIGMA(1)
      HMAX(2)=2.29*FLOAT(INFO(11))*FLOAT(INFO(11))/SIGMA(1)
C***********************************************************************
C* CALCULATE A NEW STEPLENGTH                                          *
C***********************************************************************
  150 HOLD=H
      CALL NEWH(EPS/ERROR,HOLD,H,ALFA,HMAX,INFO)
C***********************************************************************
C* IF THE ORDER EQUALS 1 AND THE STEPLENGTH IS SMALLER THAN THE MAXI-  *
C* MAL STEPLENGTH FOR ORDER 2 RESET THE ORDER                          *
C***********************************************************************
      IF(INFO(12).EQ.1.AND.H.LT.HMAX(2)) INFO(9)=0
      IF(INFO(9).EQ.0) INFO(12)=2
```

```
C*************************************************************************
C* IF STEP FAILED REJECT THE INTEGRATION STEP,CHECK FOR THREE SUCCES-  *
C* SIVE FAILURES,AND,IF NECESSARY,INTERPOLATE FOR THE NEW STEPLENGTH   *
C* ********************************************************************
      IF(EPS.GE.ERROR) GOTO 170
      REJECT=REJECT+1
      INFO(5)=INFO(5)+1
      IF(REJECT.EQ.3) GOTO 160
      CALL INTER1(N,Y,Y1,Y2,DY1,F,ALFA,INFO)
      GOTO 80
C*************************************************************************
C* RESET REJECT,INFO(I),I=6,9,12,13,14 AND THE STEPLENGTH FOR A        *
C* RESTART                                                             *
C*************************************************************************
  160 REJECT=0
      INFO(6)=INFO(6)+1
      INFO(9)=0
      INFO(12)=2
      INFO(13)=0
      INFO(14)=0
      CALL HSTART(N,Y,DY,YXE,F,TOL,APR,SIGMA(1),H,INFO)
      HOLD=H
      GOTO 80
C*************************************************************************
C* FIND OUT IF THE ORDER SHOULD CHANGE FROM 2 TO 1,ON EXIT OF THIS     *
C* PROGRAM PART H SHOULD BE RESET TO HMAX(2),IF THE ORDER HAS TO BE    *
C* CHANGED FROM 2 TO 1 SET INFO(9)=0                                   *
C*************************************************************************
  170 IF(INFO(14).LT.3.OR.INFO(12).EQ.1) GOTO 180
      IF(HOLD.NE.HMAX(2).OR.H.NE.HMAX(2)) GOTO 180
      INFO(12)=1
      CALL ESTIMA(N,Y,Y1,Y2,YXE,TOL,EPS,ERROR,INFO)
      CALL NEWH(EPS/ERROR,HOLD,H,ALFA,HMAX,INFO)
      IF(ALFA.LE.1.) INFO(12)=2
      H=HMAX(2)
      INFO(14)=-1
      IF(INFO(12).EQ.2) GOTO 180
      INFO(9)=0
C*************************************************************************
C* SHIFT THE DATA                                                      *
C*************************************************************************
  180 CALL SHIFT(N,Y,Y1,Y2,YXE,DY,DY1,X,HOLD,F,INFO)
      REJECT=0
      INFO(14)=INFO(14)+1
      INFO(15)=INFO(15)+1
C*************************************************************************
C* CHECK FOR A REESTIMATION OF THE SPECTRAL RADIUS,IF NECESSARY,CHECK   *
C* FOR A FAILURE OF POWER AND UPDATE HMAX                              *
C*************************************************************************
      IF(INFO(2).EQ.1.OR.INFO(15).NE.25) GOTO 200
      CALL POWER1(N,Y,Y1,YXE,DY,DY1,F,SIGMA,APR,IFLAG,INFO)
      IF(IFLAG.NE.3) GOTO 190
      RETURN
C
  190 HMAX(1)=5.15*FLOAT(INFO(10))*FLOAT(INFO(10))/SIGMA(1)
      HMAX(2)=2.20*FLOAT(INFO(11))*FLOAT(INFO(11))/SIGMA(1)
C*************************************************************************
```

```
C* IF X IS SMALLER THAN XE CONTINUE THE INTEGRATION,IF NECESSARY,IN=   *
C* TERPOLATE FOR A CHANGE OF H                                          *
C*************************************************************************
  200 IF(X.GE.XE) GOTO 210
      IF(H.NE.HOLD) CALL INTER1(N,Y,Y1,Y2,DY1,F,ALFA,INFO)
      IF(H.NE.HOLD.OR.INFO(9).EQ.0) GOTO 80
      GOTO 90
C*************************************************************************
C* INTERPOLATE AT XE AND,IF NECESSARY,INTERPOLATE FOR A CHANGE OF H     *
C* BEFORE RETURN                                                        *
C*************************************************************************
  210 CALL INTER2(N,Y,Y1,Y2,YXE,(X-XE)/HOLD)
      IF(H.NE.HOLD) CALL INTER1(N,Y,Y1,Y2,DY1,F,ALFA,INFO)
      RETURN
      END
```

```
      SUBROUTINE HSTART(N,Y,DY,YXE,F,TOL,APR,SIGMA,H,INFO)
C*****************************************************************************
C* SUBROUTINE HSTART CALCULATES THE INITIAL STEPLENGTH                       *
C*****************************************************************************
      DIMENSION Y(N),DY(N),YXE(N),INFO(15)
      DO 10 I=1,N
      YXE(I)=Y(I)+DY(I)/SIGMA
   10 CONTINUE
      CALL F(N,YXE)
      INFO(7)=INFO(7)+1
      ETAT=0.0
      ETAE=0.0
      DO 20 I=1,N
      ETAT=ETAT+Y(I)*Y(I)
      E=YXE(I)-DY(I)
      ETAE=ETAE+E*E
   20 CONTINUE
      ETAT=TOL+TOL*SQRT(ETAT/FLOAT(N))
      ETAE=SQRT(ETAE/FLOAT(N))/SIGMA+APR
      H=SQRT(ETAT/ETAE)/SIGMA/10.0
C
      RMMAX=INFO(11)
      BETA=(0.03*RMMAX+0.44)*RMMAX*RMMAX
      IF(H.GT.BETA/SIGMA)H=BETA/SIGMA
      RETURN
      END
```

```
      SUBROUTINE PARAM(C,LA,B,INFO)
C*****************************************************************************
C* PARAM DETERMINES THE PARAMETERS OF THE INTEGRATION SCHEME,THESE           *
C* PARAMETERS ARE EXPRESSIONS DEPENDING ON THE COEFFICIENTS OF THE           *
C* STABILITY POLYNOMIALS,WHICH ARE STORED IN THE ARRAY D. DURING             *
C* THE START,I.E.INFO(13) IS 0 OR 1,THE PARAMETERS OF A ONE-STEP SCHE-       *
C* ME ARE DETERMINED.                                                        *
C*****************************************************************************
      REAL LA
      DIMENSION C(12),LA(12),B(2),D(440),P(13),S(13),INFO(15)
      INTEGER ORDER
      DATA D(1),D(2),D(3),D(4),D(5),D(6),D(7),D(8),D(9),D(10),
     +D(11),D(12),D(13),D(14),D(15),D(16),D(17),D(18),D(19),D(20),
     +D(21),D(22),D(23),D(24),D(25),D(26),D(27),D(28),D(29),D(30)/
     +.5454545454545E+00,.32974222139503E+00,.15874540963720E-01,
     +.5454545454545E+00,.32957779372404E+00,.18807742712872E-01,
     +.26931406295668E-03,.5454545454545E+00,.32959633626966E+00,
     +.19843848141588E-01,.38370516974646E-03,.23214008199836E-05,
     +.5454545454545E+00,.32940480780399E+00,.20289622974884E-01,
     +.43922728369494E-03,.38881393659393E-05,.12058633806519E-07,
     +.5454545454545E+00,.32915730747582E+00,.20529934599533E-01,
     +.47014565070180E-03,.48729959290595E-05,.23293337843875E-07,
     +.41768932909961E-10,.5454545454545E+00,.32940290556199E+00,
     +.20695785946957E-01,.48959305208277E-03,.55250561672575E-05/
      DATA D(31),D(32),D(33),D(34),D(35),D(36),D(37),D(38),D(39),
     +D(40),D(41),D(42),D(43),D(44),D(45),D(46),D(47),D(48),D(49),
     +D(50),D(51),D(52),D(53),D(54),D(55),D(56),D(57),D(58),D(59),
     +D(60)/
     +.32042572866540E-07,.92217187087773E-10,.10431596770258E-12,
     +.5454545454545E+00,.32904622047855E+00,.20769127247467E-01,
     +.50144873108237E-03,.59538312498493E-05,.38413639404624E-07,
     +.13735161731261E-09,.25579038177072E-12,.19355325549260E-15,
     +.5454545454545E+00,.32902403825404E+00,.20835887364795E-01,
     +.51019371265378E-03,.62671263848834E-05,.43273294211670E-07,
     +.17557956622083E-09,.41529099936815E-12,.52977760638010E-15,
     +.28162396623902E-18,.5454545454545E+00,.32900701770523E+00,
     +.20847709891773E-01,.51474022625718E-03,.64655444450446E-05,
     +.46694681553560E-07,.20545752812528E-09,.55985683650846E-12/
      DATA D(61),D(62),D(63),D(64),D(65),D(66),D(67),D(68),D(69),
     +D(70),D(71),D(72),D(73),D(74),D(75),D(76),D(77),D(78),D(79),
     +D(80),D(81),D(82),D(83),D(84),D(85),D(86),D(87),D(88),D(89),
     +D(90)/
     +.92238940881933E-15,.84171480799272E-18,.32655765072494E-21,
     +.5454545454545E+00,.32923047518706E+00,.20942105514450E-01,
     +.52155114179973E-03,.66698403229501E-05,.49787290971999E-07,
     +.23175109073032E-09,.69290153772379E-12,.13309584971259E-14,
     +.15876152476716E-17,.10702794409632E-20,.31159779644428E-24,
     +.5454545454545E+00,.32888824622955E+00,.20930564082556E-01,
     +.52398048369937E-03,.67865318951772E-05,.51892263386503E-07,
     +.25160762750795E-09,.80319349035922E-12,.17105594522426E-14,
     +.24063256323529E-17,.21467888957759E-20,.11002739569540E-23,
     +.24672233557657E-27,.4545454545454545E+00,.39753050587769E+00/
      DATA D(91),D(92),D(93),D(94),D(95),D(96),D(97),D(98),D(99),
     +D(100),D(101),D(102),D(103),D(104),D(105),D(106),D(107),
     +D(108),D(109),D(110),D(111),D(112),D(113),D(114),D(115),
     +D(116),D(117),D(118),D(119),D(120)/
     +.19106034236683E-01,.4545454545454545E+00,.39769493354868E+00,
```

```
+.2267510092260 8E-01,.3243861497774 9E-03,.4545454545454 5E+00,
+.3976763910030 6E+00,.2392124693948 1E-01,.4622133454575 8E-03,
+.2794549253979 6E-05,.4545454545454 5E+00,.3978679194687 4E+00,
+.2450975404486 7E-01,.5307184801079 8E-03,.4700258940027 5E-05,
+.1458530335618 1E-07,.4545454545454 5E+00,.3981154197969 1E+00,
+.2486458958895 6E-01,.5699886029338 8E-03,.5913834060351 0E-05,
+.2830060892631 6E-07,.5081259848616 2E-10,.4545454545454 5E+00,
+.3978698217107 3E+00,.2500000228255 3E-01,.5914885843786 4E-03,
+.6675756299675 8E-05,.3872046896477 0E-07,.1114476116339 6E-09/
 DATA D(121),D(122),D(123),D(124),D(125),D(126),D(127),
+D(128),D(129),D(130),D(131),D(132),D(133),D(134),D(135),
+D(136),D(137),D(138),D(139),D(140),D(141),D(142),D(143),
+D(144),D(145),D(146),D(147),D(148),D(149),D(150)/
+.1260815372800 0E-12,.4545454545454 5E+00,.3982265067941 7E+00,
+.2518352501380 3E-01,.6088031401311 3E-03,.7235943301432 8E-05,
+.4672531710010 6E-07,.1671959450350 1E-09,.3115752774262 1E-12,
+.2359037048211 0E-15,.4545454545454 5E+00,.3982486890186 9E+00,
+.2526581987776 7E-01,.6191041154485 9E-03,.7607471114885 0E-05,
+.5253634410155 9E-07,.2131764266756 9E-09,.5042134550040 6E-12,
+.6431768853902 8E-15,.3418716316147 4E-18,.4545454545454 5E+00,
+.3982657095675 0E+00,.2528784409165 5E-01,.6249442573462 3E-03,
+.7854029693845 0E-05,.5674315145640 8E-07,.2497386916976 9E-09,
+.6806656716793 5E-12,.1121626068804 2E-14,.1023680297853 3E-17/
 DATA D(151),D(152),D(153),D(154),D(155),D(156),D(157),
+D(158),D(159),D(160),D(161),D(162),D(163),D(164),D(165),
+D(166),D(167),D(168),D(169),D(170),D(171),D(172),D(173),
+D(174),D(175),D(176)/
+.3972065796459 6E-21,.4545454545454 5E+00,.3980422520856 7E+00,
+.2534170805816 7E-01,.6315150132549 6E-03,.8080755464925 8E-05,
+.6035466440226 9E-07,.2811164209261 1E-09,.8410633463865 7E-12,
+.1616728039767 1E-14,.1929994178990 7E-17,.1302173295641 3E-20,
+.3794445366470 0E-24,.4545454545454 5E+00,.3983844810431 8E+00,
+.2541676249594 6E-01,.6369771207607 6E-03,.8255198350822 8E-05,
+.6314972988877 0E-07,.3062985870888 8E-09,.9780868191644 0E-12,
+.2083656313059 6E-14,.2932067957433 1E-17,.2616649701720 4E-20,
+.1341532828478 4E-23,.3009279619622 3E-27/
 DATA D(177),D(178),D(179),D(180),D(181),D(182),D(183),D(184),
+D(185),D(186),D(187),D(188),D(189),D(190),D(191),D(192),D(193),
+D(194),D(195),D(196),D(197),D(198),D(199),D(200),D(201),D(202),
+D(203),D(204),D(205),D(206)/
+-.5806451612903 2E+00,-.2155939517467 2E+00,-.3054469657949 2E-01,
+-.5806451612903 2E+00,-.1911522303155 4E+00,-.2947383478610 2E-01,
+-.1006634725813 5E-02,-.5806451612903 2E+00,-.1823330729713 8E+00,
+-.2823654447363 2E-01,-.1203003960123 2E-02,-.1520800833481 5E-04,
+-.5806451612903 2E+00,-.1783306994149 6E+00,-.2847524689482 7E-01,
+-.1466630203048 2E-02,-.2996411136460 0E-04,-.2134380411561 3E-06,
+-.5806451612903 2E+00,-.1761036709831 5E+00,-.2826067664509 0E-01,
+-.1532245931033 6E-02,-.3658632011421 2E-04,-.3986752942793 5E-06,
+-.1622666513519 9E-03,-.5806451612903 2E+00,-.1748339011491 1E+00,
+-.2774118622624 6E-01,-.1481595689891 8E-02,-.3630104539372 9E-04/
 DATA D(207),D(208),D(209),D(210),D(211),D(212),D(213),D(214),
+D(215),D(216),D(217),D(218),D(219),D(220),D(221),D(222),D(223),
+D(224),D(225),D(226),D(227),D(228),D(229),D(230),D(231),D(232),
+D(233),D(234),D(235),D(236)/
+-.4468500755077 8E-06,-.2687558450728 1E-08,-.6283123108335 2E-11,
+-.5806451612903 2E+00,-.1739607123867 1E+00,-.2808413422659 3E-01,
+-.1743249591946 6E-02,-.5084586009299 3E-04,-.7913777312667 8E-06,
```

```
+-.6737475354772 9E-08,-.2958925231863 2E-10,-.5241629406350 7E-13,
+-.5806451612903 2E+00,-.1733987837384 2E+00,-.2804472727200 7E-01,
+-.1625314353245 3E-02,-.4581006572423 1E-04,-.7115578800385 6E-06,
+-.6404907274814 0E-08,-.3325935591593 9E-10,-.9239227819052 2E-13,
+-.1062514796895 7E-15,-.5806451612903 2E+00,-.1714417337700 9E+00,
+-.2801509293851 8E-01,-.1615722237363 3E-02,-.4622772482905 7E-04,
+-.7547922234554 4E-06,-.7489483501751 1E-08,-.4597905353583 7E-10/
 DATA D(237),D(238),D(239),D(240),D(241),D(242),D(243),D(244),
+D(245),D(246),D(247),D(248),D(249),D(250),D(251),D(252),D(253),
+D(254),D(255),D(256),D(257),D(258),D(259),D(260),D(261),D(262),
+D(263),D(264),D(265),D(266)/
+-.1706013779 770E-12,-.3505666308659 7E-15,-.3062888661819 1E-18,
+-.5806451612903 2E+00,-.1730028416629 4E+00,-.2732800902421 4E-01,
+-.1505602154505 1E-02,-.4136745329211 4E-04,-.6588428978007 4E-06,
+-.6553614631734 1E-08,-.4209102515936 4E-10,-.1747997705631 7E-12,
+-.4536840682099 8E-15,-.6692870320841 2E-18,-.4284431115575 2E-21,
+-.5806451612903 2E+00,-.1722936696098 4E+00,-.2880027238157 4E-01,
+-.1859617443042 0E-02,-.5997697873090 6E-04,-.1109056694535 9E-05,
+-.1274593569223 8E-07,-.9515869465786 9E-10,-.4696940647697 5E-12,
+-.1521806413652 8E-14,-.3113095210850 8E-17,-.3646696153211 9E-20,
+-.1864493436819 3E-23,.1580645161290 3E+01,.1505916532391 9E+01/
 DATA D(267),D(268),D(269),D(270),D(271),D(272),D(273),D(274),
+D(275),D(276),D(277),D(278),D(279),D(280),D(281),D(282),D(283),
+D(284),D(285),D(286),D(287),D(288),D(289),D(290),D(291),D(292),
+D(293),D(294),D(295),D(296)/
+.1697894545101 9E+00,.1580645161290 3E+01,.1481474810960 7E+01,
+.1931603141479 8E+00,.6233416339884 3E-02,.1580645161290 3E+01,
+.1472655653616 5E+01,.2007421811796 6E+00,.8633697663050 8E-02,
+.1155808458719 7E-03,.1580645161290 3E+01,.1468653280060 1E+01,
+.2049832571572 8E+00,.9993811886329 1E-02,.1992234803629 6E-03,
+.1391920144892 2E-05,.1580645161290 3E+01,.1466426251628 3E+01,
+.2069957153393 5E+00,.1068027540554 5E-01,.2493340506726 3E-03,
+.2685503911166 6E-05,.1085485071360 1E-07,.1580645161290 3E+01,
+.1465156481794 3E+01,.2077459947545 6E+00,.1097186105226 7E-01,
+.2752436883000 2E-03,.3540365693442 3E-05,.2257532123676 1E-07/
 DATA D(297),D(298),D(299),D(300),D(301),D(302),D(303),D(304),
+D(305),D(306),D(307),D(308),D(309),D(310),D(311),D(312),D(313),
+D(314),D(315),D(316),D(317),D(318),D(319),D(320),D(321),D(322),
+D(323),D(324),D(325),D(326)/
+.5657448788258 5E-10,.1580645161290 3E+01,.1464283293531 9E+01,
+.2095621310173 0E+00,.1150956610737 5E-01,.3109798816382 8E-03,
+.4563098613330 2E-05,.3707964113088 3E-07,.1568259095019 4E-09,
+.2693343003558 8E-12,.1580645161290 3E+01,.1463721364383 6E+01,
+.2094846532110 1E+00,.1153641674770 9E-01,.3178461487054 8E-03,
+.4911672004700 9E-05,.4452819608048 1E-07,.2350532833139 3E-09,
+.6637011849380 9E-12,.7923943846638 5E-15,.1580645161290 3E+01,
+.1461764314415 2E+01,.2114120688458 4E+00,.1180505628802 4E-01,
+.3344074399447 9E-03,.5441423529388 0E-05,.5391808643411 1E-07,
+.3307545012190 3E-09,.1226405738675 4E-11,.2518106403672 4E-14/
 DATA D(327),D(328),D(329),D(330),D(331),D(332),D(333),D(334),
+D(335),D(336),D(337),D(338),D(339),D(340),D(341),D(342),D(343),
+D(344),D(345),D(346),D(347),D(348),D(349),D(350),D(351),D(352)/
+.2197761007281 0E-17,.1580645161290 3E+01,.1463325422308 1E+01,
+.2091638772386 9E+00,.1157945455023 9E-01,.3285730552900 8E-03,
+.5445838801578 4E-05,.5637197751929 4E-07,.3754508369244 0E-09,
+.1609100595032 5E-11,.4288439877616 6E-14,.6466583268541 4E-17,
+.9214845792410 5E-20,.1580645161290 3E+01,.1462616250255 0E+01,
```

```
     +.21134531244915E+00,.1210812156855 4E-01,.3589415374545 0E-03,
     +.6266442262467 1E-05,.6919336261629 7E-07,.5019517926183 5E-09,
     +.2425329203631 7E-11,.7731036546180 3E-14,.1561392181052 6E-16,
     +.1810281959915 5E-19,.9177610747672 7E-23/
C
      DATA D(353),D(354),D(355),D(356),D(357),D(358),D(359),D(360),
     +D(361),D(362),D(363),D(364),D(365),D(366),D(367),D(368),D(369),
     +D(370),D(371),D(372),D(373),D(374),D(375),D(376),D(377),D(378),
     +D(379),D(380),D(381),D(382),D(383),D(384),D(385),D(386),D(387),
     +D(388),D(389),D(390),D(391),D(392),D(393),D(394),D(395),D(396),
     +D(397),D(398),D(399),D(400),D(401),D(402)/
     +1.,1.,.5,
     +1.,1.,0.5,63304085.E-09,1.,1.,0.5,79027358.E-09,37089254.E-10,
     +1.,1.,0.5,85605575.E-09,56773923.E-10,12758716.E-11,1.,1.,0.5,
     +89018496.E-09,67947003.E-10,23143226.E-11,2898388.E-12,1.,1.,0.5,
     +91025408.E-09,74822425.E-10,30567580.E-11,6072001.E-12,
     +4679323.E-14,1.,1.,0.5,92308224.E-09,79335426.E-10,35849723.E-11,
     +8800161.E-12,11108474.E-14,5648779.E-16,1.,1.,0.5,93178948.E-09,
     +82451157.E-10,39680345.E-11,11000301.E-12,17552367.E-14/
      DATA D(403),D(404),D(405),D(406),D(407),D(408),D(409),D(410),
     +D(411),D(412),D(413),D(414),D(415),D(416),D(417),D(418),D(419),
     +D(420),D(421),D(422),D(423),D(424),D(425),D(426),D(427),D(428),
     +D(429),D(430),D(431),D(432),D(433),D(434),D(435),D(436),D(437),
     +D(438),D(439),D(440)/
     +14976734.E-16,5293311.E-18,1.,1.,0.5,93797465.E-09,84690176.E-10,
     +42524183.E-11,12746945.E-12,23355062.E-14,25642831.E-16,
     +1549596 7.E-18,3962808.E-22,1.,1.,0.5,94252811.E-09,86352191.E-10,
     +44683802.E-11,14135170.E-12,28359079.E-14,36242802.E-16,
     +28591262.E-18,12691526.E-20,24249737.E-23,1.,1.,0.5,94597848.E-09,
     +87619296.E-10,46357872.E-11,15246910.E-12,32599754.E-14,
     +46107927.E-16,42819928.E-18,25110371.E-20,84319465.E-23,
     +12357105.E-25/
C
      ORDER=INFO(12)
      M=INFO(9)
      IF(INFO(13).LT.2) GOTO 50
      M1=176*(ORDER-1)+M*(M+1)/2-3
      M2=88+M1
      MM=M+1
      DO 10 J=1, MM
      M1PJ=M1+J
      M2PJ=M2+J
      P(J)=D(M1PJ)
      S(J)=D(M2PJ)
   10 CONTINUE
      IF(M.GT.2) GOTO 20
      P(4)=0.0
      S(4)=0.0
   20 DD=1.375-.5*FLOAT(ORDER-1)
      E=P(2)+2.0*(-P(3)+P(4)+S(4))
      C(M)=(E/DD-((DD-0.5)/DD)**2)/(2.+E)
      LA(M)=1.0/DD-C(M)
      LA(M-1)=S(3)/LA(M)
      C(M-1)=P(3)/LA(M)
      P(1)=(P(2)-C(M))/LA(M)
      S(2)=P(1)
      IF(M.GT.2) GOTO 30
```

```
      RETURN
C
   30 MM=M-2
      DO 40 J=1,MM
      MP2MJ=M+2-J
      MP1MJ=M+1-J
      C(J)=P(MP2MJ)/S(MP1MJ)
      LA(J)=S(MP2MJ)/S(MP1MJ)
   40 CONTINUE
      RETURN
C
   50 M1=352+M*(M+1)/2-3
      MM=M+1
      DO 60 J=1,MM
      M1PJ=M1+J
   60 S(J)=D(M1PJ)
      B(1)=0.0
      B(2)=0.0
      C(M)=0.0
      LA(M)=S(1)
      MM=M-1
      DO 70 J=1,MM
      MP2MJ=M+2-J
      MP1MJ=M+1-J
      LA(J)=S(MP2MJ)/S(MP1MJ)
      C(J)=0
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE POWERM(N,Y,Y1,YXE,DY,DY1,F,SIGMA,APR,IFLAG,INFO)
C*****************************************************************
C* IF ON ENTRY SIGMA(1)=0 POWERM COMPUTES AN ESTIMATION OF THE SPECTR- *
C* AL RADIUS OF THE JACOBIAN BY MEANS OF AN ADJUSTED POWER METHOD,THE  *
C* ITERATION IS STOPPED IF TWO SUCCESSIVE ITERATES DIFFER RELATIVELY   *
C* LESS THAN 0.001,THE MINIMAL NUMBER OF ITERATIONS IS 5,IF THE COMPU- *
C* TATION DID NOT SUCCEED WITHIN 50 ITERATIONS AN ERRORMESSAGE IS GIV- *
C* EN,AS A SAFETY MARGIN,THE LAST ITERATE IS ENLARGED WITH 10 PERCENT. *
C* THE RESULT IS STORED IN SIGMA(1).                                  *
C*                                                                    *
C* IF ON ENTRY SIGMA(1) NOT EQUALS 0 POWERM PERFORMS THREE ITERATIONS  *
C* WITH THE ADJUSTED POWER METHOD,IF THE THIRD ITERATE IS MORE THAN 10 *
C* PERCENT SMALLER THAN SIGMA(2),THE ITERATION IS CONTINUED AS IF      *
C* SIGMA(1)=0.IN THIS CASE THE THIRD ITERATE IS STORED IN SIGMA(2).    *
C*                                                                    *
C* THE POWER METHOD REQUIRES THREE WORK ARRAYS,WE USE YXE,DY AND DY1,  *
C* WHERE DY AND DY1 ARE OVERWRITTEN.ON ENTRY OF THIS ROUTINE DY AND    *
C* DY1 CONTAIN F(Y) AT Y=Y(X) AND Y=Y(X-H),RESPECTIVELY,THUS ON EXIT   *
C* OF THIS ROUTINE TWO EXTRA EVALUATIONS OF F ARE NECESSARY FOR RESTO- *
C* RING THE DERIVATIVES.                                              *
C*                                                                    *
C* THIS CODE USES THE CDC SYSTEM SUBPROGRAMS RANSET AND RANF,GENE-     *
C* RATING RANDOM VALUES FROM THE INTERVAL [0,1], THE ARGUMENT OF RANF  *
C* IS DUMMY AND IGNORED.RANSET INITIALIZES THE GENERATIVE VALUE OF RANF*
C*****************************************************************
      DIMENSION Y(N),Y1(N),YXE(N),DY(N),DY1(N),SIGMA(2),INFO(15)
      REAL NORM,NORM0
      IF(INFO(2).EQ.3) INFO(2)=1
      INFO(15)=0
      TOLLIP=1.E+4*APR
      SIGM=0.0
      S0=0.0
      CALL RANSET(0)
      DO 10 I=1,N
      RA=2.*RANF(IDUM)-1.
      YXE(I)=DY(I)
      IF(Y(I).EQ.0.0) DY(I)=RA*TOLLIP
      IF(Y(I).NE.0.0) DY(I)=Y(I)*(1.0+TOLLIP*RA)
      DY1(I)=DY(I)
      S0=S0+DY(I)*DY(I)
   10 CONTINUE
      NORM0=TOLLIP*SQRT(S0)
      IF(NORM0.LT.TOLLIP)NORM0=TOLLIP
      CALL F(N,DY1)
      INFO(7)=INFO(7)+1
      INFO(8)=INFO(8)+1
C
      DO 70 K=1,51
      IF(K.LT.51) GOTO 20
      IFLAG=3
      RETURN
C
   20 S0=0
      DO 30 I=1,N
      S1=YXE(I)-DY1(I)
      S0=S0+S1*S1
   30 CONTINUE
```

```
      NORM=SQRT(S0)
      SIGM1=SIGM
      SIGM=NORM/NORM0
C
      IF(K.EQ.3.AND.SIGMA(1).EQ.0.0) SIGMA(2)=SIGM
      IF(K.LE.2.OR.SIGMA(1).EQ.0.0) GOTO 40
      IF(SIGM.GE.0.9*SIGMA(2)) GOTO 80
      SIGMA(2)=SIGM
      SIGMA(1)=0.
C
   40 IF(ABS(SIGM1-SIGM)/SIGM.GT.0.001.OR.K.LE.4) GOTO 50
      SIGMA(1)=1.1*SIGM
      GOTO 80
   50 DO 60 I=1,N
   60 YXE(I)=DY(I)+(YXE(I)-DY1(I))/SIGM
      CALL F(N,YXE)
      INFO(7)=INFO(7)+1
      INFO(8)=INFO(8)+1
   70 CONTINUE
C
   80 DO 90 I=1,N
   90 DY(I)=Y(I)
      CALL F(N,DY)
      INFO(7)=INFO(7)+1
      INFO(8)=INFO(8)+1
      IF(INFO(13).NE.0) GOTO 100
      RETURN
  100 DO 110 I=1,N
  110 DY1(I)=Y1(I)
      CALL F(N,DY1)
      INFO(7)=INFO(7)+1
      INFO(8)=INFO(8)+1
      RETURN
      END
```

```
      SUBROUTINE MAXDEG(TOL,APR,IFLAG,INFO)
C*****************************************************************
C* IN MAXDEG THE MAXIMAL DEGREES WITH RESPECT TO THE INTERNAL STABILI* *
C* TY CONDITION ARE COMPUTED. IF ONE OF THESE MAXIMAL DEGREES FALLS   *
C* OUTSIDE THE RANGE,AN ERRORMESSAGE IS GIVEN.                        *
C*****************************************************************
      DIMENSION Q(11),INFO(15)
      DATA Q(1),Q(2),Q(3),Q(4),Q(5),Q(6),Q(7),Q(8),Q(9),Q(10),Q(11)/
     +3.E1,1.E2,7.E2,4.E3,3.E4,2.E5,9.E5,5.E6,3.E7,2.E8,1.E9/
      E=TOL/APR
      DO 10 I=2,12
      J=13-I
      IF(Q(J).LE.E) GOTO 20
   10 CONTINUE
      IFLAG=2
      RETURN
C
   20 INFO(10)=14-I
      DO 30 I=2,12
      J=13-I
      IF(100.0*Q(J).LE.E) GOTO 40
   30 CONTINUE
      IFLAG=2
      RETURN
C
   40 INFO(11)=14-I
      RETURN
      END
```

```
      SUBROUTINE MINDEG(H,SIGMA,INFO)
C*****************************************************************
C* MINDEG DETERMINES THE MINIMAL DEGREE M WHICH STILL GIVES RISE TO A  *
C* STABLE INTEGRATION STEP.                                            *
C*****************************************************************
      DIMENSION INFO(15)
      LOGICAL START
      L=INFO(12)+9
      MMAX=INFO(L)
      START=INFO(13).LT.2
      IF(.NOT.START) BETAC=5.15+2.86*FLOAT(1-INFO(12))
      DO 10 M=2,MMAX
      RM=M
      IF(START) BETAC=0.03*RM+0.44
      IF(H.LE.BETAC*RM*RM/SIGMA) GOTO 20
   10 CONTINUE
      M=MMAX
   20 INFO(9)=M
      RETURN
      END
```

```
      SUBROUTINE STEP(N,Y,Y1,Y2,YXE,DY,DY1,H,F,C,LA,B,INFO)
C*****************************************************************
C* STEP CONTAINS THE ACTUAL INTEGRATOR.FOR CONVENIENCE,THE ONE-STEP  *
C* SCHEME IS ALSO FORMULATED AS A THREE-STEP SCHEME BY INTRODUCING ZE- *
C* RO-PARAMETERS.ON EXIT OF THIS ROUTINE THE NEW CALCULATED SOLUTION  *
C* VECTOR IS CONTAINED IN YXE.                                        *
C*****************************************************************
      REAL LA
      DIMENSION Y(N),Y1(N),Y2(N),YXE(N),DY(N),DY1(N),
     +LA(12),C(12),B(2),INFO(15)
      N=INFO(9)
      D=1.375-.6*FLOAT(INFO(12)-1)
      IF(INFO(13).LT.2) D=1.0
      DO 10 I=1,N
   10 YXE(I)=DY(I)
      IF(N.EQ.2) GOTO 40
      NM=N-2
C
      DO 30 J=1,NM
      HC=H*C(J)
      HLA=H*LA(J)
      DO 20 I=1,N
      YXE(I)=Y(I)+HC*DY1(I)+HLA*YXE(I)
   20 CONTINUE
      CALL F(N,YXE)
      INFO(7)=INFO(7)+1
   30 CONTINUE
C
   40 BM1=B(1)
      BM11=1.0-BM1
      HC=H*C(N-1)
      HLA=H*LA(N-1)
      DO 50 I=1,N
   50 YXE(I)=BM11*Y(I)+BM1*Y1(I)+HC*DY1(I)+HLA*YXE(I)
      D1=1.-D
      BM1=B(2)*D
      BM11=(1.-B(2))*D
      HC=H*C(N)*D
      HLA=H*LA(N)*D
      CALL F(N,YXE)
      INFO(7)=INFO(7)+1
      DO 60 I=1,N
   60 YXE(I)=BM11*Y(I)+BM1*Y1(I)+D1*Y2(I)+HC*DY1(I)+HLA*YXE(I)
      RETURN
      END
```

```
      SUBROUTINE ESTIMA(N,Y,Y1,Y2,YXE,TOL,EPS,ERROR,INFO)
C*****************************************************************
C* ESTIMA CALCULATES THE LOCAL ERROR BOUND EPS=(1+NORM(Y))*TOL FOR THE *
C* MIXED ERROR TEST AND ESTIMATES THE LOCAL ERROR ERROR.           *
C*****************************************************************
      DIMENSION Y(N),Y1(N),Y2(N),YXE(N),INFO(15),CONST(2)
      INTEGER ORDER
      CONST(1)=2.85
      CONST(2)=0.49
      ORDER=INFO(12)
      EPS=0.0
      ERROR=0.0
      IF(ORDER.EQ.2) GOTO 20
      DO 10 I=1,N
      YI=Y(I)
      EPS=EPS+YI*YI
      E=Y1(I)-YI-YI+YXE(I)
      ERROR=ERROR+E*E
   10 CONTINUE
C
      GOTO 40
   20 DO 30 I=1,N
      EPS=EPS+Y(I)*Y(I)
      E=-Y2(I)+3.0*(Y1(I)-Y(I))+YXE(I)
      ERROR=ERROR+E*E
   30 CONTINUE
C
   40 EPS=TOL+TOL*SQRT(EPS/FLOAT(N))
      ERROR=CONST(ORDER)*SQRT(ERROR/FLOAT(N))
      RETURN
      END
```

```
      SUBROUTINE NEWH(EPSERR,HOLD,H,ALFA,HMAX,INFO)
C*****************************************************************************
C* NEWH DELIVERS A NEW STEPLENGTH AND THE FACTOR ALFA BY WHICH THE      *
C* STEPLENGTH IS CHANGED. EPSERR DENOTES EPS/ERROR.                     *
C*****************************************************************************
      DIMENSION HMAX(2),INFO(15)
      INTEGER ORDER
      ALFA=1.0
      IF(INFO(14).GE.3.OR.EPSERR.LE.1.0) GOTO 10
      RETURN
C
   10 ORDER=INFO(12)
      ALFA=EPSERR**(1./FLOAT(ORDER+1))/(2.0-FLOAT(ORDER-1)*0.4)
      IF(ALFA.GT.0.9.AND.ALFA.LT.1.1) ALFA=1.0
      IF(ALFA.NE.1.0) GOTO 20
C
      RETURN
   20 IF(ALFA.GT.3.0) ALFA=3.0
      IF(ALFA.LT.0.1) ALFA=0.1
      H=HOLD*ALFA
      IF(H.GT.HMAX(ORDER)) H=HMAX(ORDER)
      ALFA=H/HOLD
      RETURN
      END
```

```
      SUBROUTINE INTER1(N,Y,Y1,Y2,DY1,F,ALFA,INFO)
C*************************************************************************
C* INTER1 COMPUTES VALUES FOR Y(X=ALFA*H) AND Y(X=2*ALFA*H) FROM Y(X), *
C* Y(X=H) AND Y(X=2H) BY QUADRATIC INTERPOLATION,AND COMPUTES THE DER= *
C* IVATIVE AT X=H BY CALLING F.                                        *
C*************************************************************************
      DIMENSION Y(N),Y1(N),Y2(N),DY1(N),INFO(15)
      REAL NU
      NU=2.-ALFA
      C12=(NU-1.)*(NU-2.)/2.
      C11=NU*(2.-NU)
      C10=NU*(NU-1.)/2.
      NU=2.-2.*ALFA
      C22=(NU-1.)*(NU-2.)/2.
      C21=NU*(2.-NU)
      C20=NU*(NU-1.)/2.
C
      DO 10 I=1,N
      CY0=Y(I)
      CY1=Y1(I)
      CY2=Y2(I)
      Y1(I)=C12*CY2+C11*CY1+C10*CY0
      DY1(I)=Y1(I)
      Y2(I)=C22*CY2+C21*CY1+C20*CY0
   10 CONTINUE
      CALL F(N,DY1)
      INFO(7)=INFO(7)+1
      INFO(14)=0
      RETURN
      END
```

```
      SUBROUTINE INTER2(N,Y,Y1,Y2,YXE,A)
C*************************************************************************
C* INTER2 COMPUTES THE SOLUTION AT THE OUTPUT POINT XE=X-A*H BY QUAD-   *
C* RATIC INTERPOLATION BETWEEN Y(X),Y(X-H) AND Y(X-2H),THE RESULT IS    *
C* STORED IN YXE.                                                       *
C*************************************************************************
      DIMENSION Y(N),Y1(N),Y2(N),YXE(N)
      REAL NU
      NU=2.-A
      C12=(NU-1.)*(NU-2.)/2.
      C11=NU*(2.-NU)
      C10=NU*(NU-1.)/2.
      DO 10 I=1,N
   10 YXE(I)=C12*Y2(I)+C11*Y1(I)+C10*Y(I)
      RETURN
      END
```

```
      SUBROUTINE SHIFT(N,Y,Y1,Y2,YXE,DY,DY1,X,H,F,INFO)
C*************************************************************************
C* SHIFT SHIFTS THE X AND Y VARIABLES AND COMPUTES F(Y(X+H))              *
C*************************************************************************
      DIMENSION Y(N),Y1(N),Y2(N),YXE(N),DY(N),DY1(N),INFO(15)
      DO 10 I=1,N
      Y2(I)=Y1(I)
      Y1(I)=Y(I)
      Y(I)=YXE(I)
      DY1(I)=DY(I)
      DY(I)=Y(I)
   10 CONTINUE
      CALL F(N,DY)
      INFO(7)=INFO(7)+1
      X=X+H
      RETURN
      END
```