stichting
mathematisch
centrum

MC

AFDELING TOEGEPASTE WISKUNDE TW 130/71 SEPTEMBER

P.J. VAN DER HOUWEN, P. BEENTJES, K. DEKKER
and E. SLAGT

ONE STEP METHODS FOR LINEAR INITIAL VALUE PROBLEMS III
NUMERICAL RESULTS

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), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

Contents

1. Introduction

This paper contains the ALGOL 60 realization and a number of applications of some of the numerical integration methods described and analysed in references [3] and [4]. Two ALGOL 60 procedures are presented, procedure modified taylor and procedure exponential fitted taylor. The applications are chosen in the fields of ordinary differential equations, including stiff equations, and of partial differential equations.

Section 2 deals with procedure modified taylor; a definition of its parameters is given as well as an outline of the several subprocedures. Modified taylor chooses its step sizes automatically depending on the required accuracy and the spectral radius of the system of differential equations to be solved. The storage requirements of the procedure are minimized which makes it an appropriate method for the integration of large systems (1000 or more equations) such as the systems which arise when by discretizing the space variables a partial differential equation is reduced to a set of ordinary differential equations. As a consequence,modified taylor does not reject an integration step when it turns out that the discrepancy exceeds the tolerance. On the other hand, a stability criterion, derived in the theoretical treatment of the modified Taylor method ([3] and [4]), is incorporated. It is here that the spectral radius of the system enters into the considerations. A further remark is that in those cases where the required order of accuracy p is less than the number n of the available derivatives of the local solution (to be given by the user of the program), the discrepancy is estimated by the first neglected Taylor terms instead of the last correction term.

In section 3 a number of experiments carried out on a simple ordinary differential equation are analysed. Attention is paid to the local error of the integration method, weak and strong stability, and the consequences of the stiffness of a differential equation.

Sections 4 and 5 are devoted to the integration of hyperbolic and parabolic differential equations, respectively.

Finally, in section 6 and 7 the procedure exponential fitted taylor and its application to stiff equations are given. Unlike procedure modified taylor, exponential fitted taylor estimates the discrepancy by the residual term obtained when the local numerical solution is substituted into the differential equation. It turned out that for stiff equations such an estimate is more realistic than an estimate based on Taylor terms.

The research presented in this paper was carried out by members of the Applied Mathematical Department (Van der Houwen, Slagt) and by members of the Computational Department (Beentjes, Dekker). The experiments were carried out on the EL X8 computer of the Mathematical Centre.

2. The modified Taylor method

In this section we describe the ALGOL 60 version of the polynomial methods, with constant coefficients, as discussed in [3] and [4]. These methods can be used for the numerical integration of initial value problems of the type

$$(2.1)\quad \begin{cases} \dfrac{dU}{dt} = H(U,t), & t_0 \leq t \leq T, \\ \\ U = \tilde{U}_0, & t = t_0, \end{cases}$$

where the function H(U,t) is such that a sufficient number, say n, of derivatives of U can be derived explicitly by repeated differentiation of the differential equation. For instance, linear initial value problems of the type

$$(2.1')\quad \begin{cases} \dfrac{dU}{dt} = DU + F, & t_0 \leq t \leq T, \\ \\ U = \tilde{U}_0, & t = t_0, \end{cases}$$

where D is a matrix with constant entries and F is an easy to differentiate function, can be dealed with polynomial methods.

The procedure modified taylor, discussed in the following subsections, represents the ALGOL 60 version of the polynomial method with constant coefficients.

2.1 Procedure modified taylor

Firstly, the heading of procedure modified taylor and the meaning of its parameters will be given:

```
procedure  modified taylor (t, te, m0, m, u, sigma, i, derivative, k, data,
                             alfa, norm, aeta, reta, eta, rho, output);

integer    m0, m, i, k, norm;

real       t, te, sigma, alfa, aeta, reta, eta, rho;

array      u, data;

procedure  derivative, output;
```

The actual parameters corresponding to the formal parameters are:

t:	<variable>; t is used as Jensen parameter; when modified taylor is called t should have its initial value;
te:	<expression>; the end value of t;
m0,m:	<expression>; indices of the first and last equation of the system to be solved;
u:	a one-dimensional array u[m0:m]; when modified taylor is called u should contain the initial values of the analytical solution $\tilde{U}(t)$;
sigma:	<expression>; largest absolute value of those eigenvalues of the Jacobian D of the system which are not in the positive half-plane; sigma should be given by the user of the procedure;
i:	<variable>; a Jensen parameter for procedure derivative;
derivative:	a procedure to be declared by the user: procedure derivative (i,a); integer i; array a; <body>;

i assumes the values 1, 2, ..., n and a is a one-dimensional array a[m0:m];
when this procedure is called in modified taylor array a contains the components of the (i-1)-st derivative of U(t) at the point (t,u);
upon completion of a call of derivative array a should contain the components of the i-th derivative of U(t) at the point (t,u);

k: <variable>;
counts the integration steps;
if $k = 0$ then the integration starts with a trial step

$$\tau_0 = \mathrm{Min}\left[\frac{\eta_0}{||H(t_0,\tilde{U}_0)||}, \frac{\beta(n)}{\sigma(D_0)}\right] ;$$

if $k > 0$ then the integration proceeds with a step based on the last three computed discrepancies;
in the very first call of modified taylor it is required that $k = 0$;

data: a one-dimensional array data[-2: data[-2]];
data [-2]: the number n of derivatives of U(t) to be used;
data [-1]: order of accuracy of the method;
data [0]: stability parameter $\beta(n)$ (see references [3] and [5]);
data [1], ..., data [data [-2]]: coefficients β_j, $j = 1, \ldots, n$ of the polynomial method (cf. section 8);

alfa: <expression>;
the step sizes τ_k satisfy the condition $\tau_k \leq$ alfa * τ_{k-1};

norm: <expression>;
selects the norm according to which the discrepancy is estimated (see section 2.3);

aeta,reta: <expression>;
desired absolute and relative local accuracy;
when both aeta and reta are negative modified taylor skips the accuracy conditions;

eta: <variable>;
the tolerance η_k which is some function of aeta and reta (see section 2.4);

rho: <variable>;
discrepancy used as an estimate for the local error produced in the last integration step;

output: a procedure to be declared by the user:
procedure output;
<body>;
by this procedure one may order to print the values of e.g.
t, u[m0], ..., u[m], sigma, k, eta, rho;

Next we give the body of procedure modified taylor. For a discussion of the procedures declared within modified taylor we refer to the following subsections. For the procedure vecvec one is referred to [2], section 3.2.3.

```
begin i:=0;
    begin integer n,p,q;
        own real ec0,ec1,ec2,tau0,tau1,tau2,taus,t2;
        real tau,taui,tauec,ecl,betan,gamma;
        real array c[m0:m],beta,betha[1:data[-2]];
        boolean start,step1;
        procedure coefficient;
        begin integer j;real ifac;
            ifac:=1; gamma:=.5; n:=data[-2]; p:=data[-1];
            betan:=data[0]; q:= if p<n then p+1 else n;
            for j:=1 step 1 until n do
            begin beta[j]:=data[j]; ifac:=ifac/j;
                betha[j]:=ifac-beta[j]
            end;
            if p=n then betha[n]:=ifac
        end;
```

```
real procedure normfunction(norm,w);
integer norm; array w;
begin integer j; real s,x;
    s:=0;
    if norm=1 then
    begin for j:=m0 step 1 until m do
        begin  x:=abs(w[j]); if x>s then s:=x end
    end else
    s:=sqrt(vecvec(m0,m,0,w,w));
    normfunction:=s
end;

procedure local error bound;
eta:=aeta+reta × normfunction(norm,u);

procedure local error construction(i);integer i;
begin if i=p then begin ecl:=0;tauec:=1 end;
    if i>p+1 then tauec:=tauec×tau;
    ecl:=ecl+abs(betha[i])×tauec×normfunction(norm,c);
    if i=n then
    begin ec0:=ec1;ec1:=ec2;ec2:=ecl;
        rho:=ecl×tau↑q
    end
end;

procedure stepsize;
begin real tauacc,taustab,aa,bb,cc,ec;
    local error bound;
    if eta>0 then
    begin if start then
begin if k=0 then
    begin integer j;
        for j:=m0 step 1 until m do c[j]:=u[j];
        i:=1; derivative(i,c);
        tauacc:=eta/normfunction(norm,c);
        step1:=true
    end else
    if step1 then
    begin tauacc:=(eta/rho)⋀(1/q)×tau2;
        if tauacc>10×tau2 then
        tauacc:=10×tau2 else step1:=false
    end else
    begin bb:=(ec2−ec1)/tau1; cc:=ec2−bb×t2;
        ec:=bb×t+cc;
        tauacc:=if ec<0 then tau2 else
        (eta/ec)⋀(1/q);
        start:=false
    end
end else
begin aa:=((ec0−ec1)/tau0+(ec2−ec1)/tau1)/
          (tau1+tau0);
    bb:=(ec2−ec1)/tau1−aa×(2×t2−tau1);
    cc:=ec2−t2×(bb+aa×t2); ec:=cc+t×(bb+t×aa);
    tauacc:=if ec<0 then taus else(eta/ec)⋀(1/q);
    if tauacc>alfa×taus then tauacc:=alfa×taus;
    if tauacc<gamma×taus then tauacc:=gamma×taus;
```

```
                if tauacc<⏨-12 × t then tauacc:= ⏨-12 × t
            end
        end else tauacc:=te-t;
        taustab:=betan/sigma;
        if taustab<⏨-12 × t then goto end of modified taylor;
        tau:=if tauacc>taustab then taustab else tauacc;
        taus:=tau; if tau>te-t then tau:=te-t;
        tau0:=tau1;tau1:=tau2;tau2:=tau
    end;
    procedure difference scheme;
    begin integer j; real b;
        for j:=m0 step 1 until m do c[j]:=u[j]; taui:=1;
    next term: i:=i+1; derivative(i,c); taui:=taui×tau;
        b:=beta[i]×taui;
        if eta>0 ∧ i≥p then local error construction(i);
        for j:=m0 step 1 until m do u[j]:=u[j]+b×c[j];
        if i<n then goto next term;
        t2:=t; t:=t+tau
    end;

    start:= if k=0 then true else false;
    coefficient;
    next level:
    stepsize; k:=k+1; i:=0; difference scheme; output;
    if t<te then goto next level
  end;
  end of modified taylor:
end;
```

2.2 Procedure difference scheme

By this procedure the values of u[j], representing the components of the numerical solution u_k, are replaced by the components of u_{k+1}.

The procedure exactly follows the computational scheme given in [3], section 2.1. During the construction of u_{k+1}, by summing the successive correction terms $\beta_i \tau_k^i c_k^{(i)}$, $i = 1, 2, \ldots, n$, an estimate of the local error, i.e. the discrepancy, is build up by procedure local error construction.

2.3 Procedure local error construction

Procedure stepsize is based on estimates of the local errors $||\rho_k(\tau_k)||$ evaluated by procedure local error construction. Therefore, this procedure plays a central role in our calculations given in subsequent sections.

The local error $\rho_k(\tau_k)$ is defined by (cf. [3], section 2.2).

$$(2.2) \qquad \rho_k(\tau_k) = \tilde{U}_{k+1} - u'_{k+1},$$

where $\tilde{U}_{k+1}$ is the analytical solution and u'_{k+1} is the numerical result at the point $t = t_{k+1}$ which would be obtained when the difference scheme is applied at the point $(t_k, \tilde{U}_k)$ (see figure 2.1).

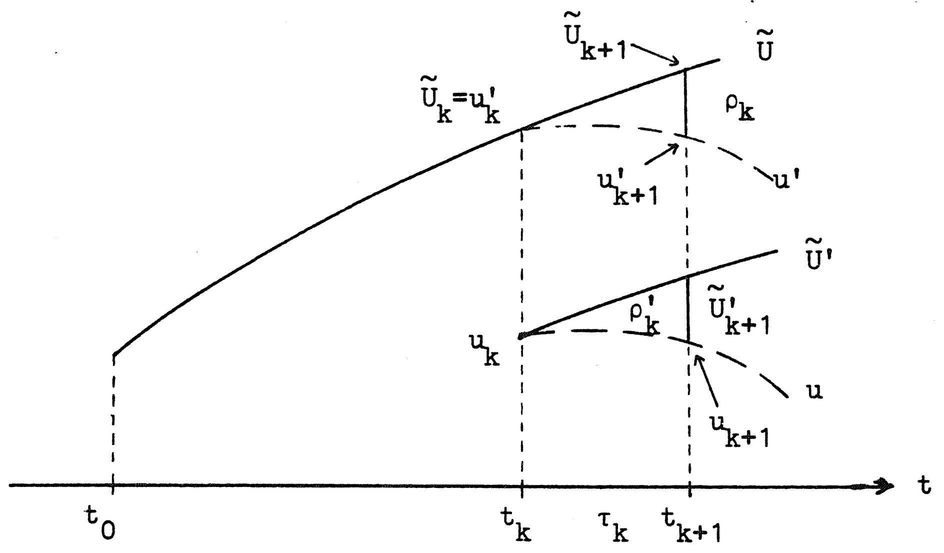

fig. 2.1 Local discretization error and discrepancy in a scalar case

However, as already observed in [4], section 2.2, it is not ρ_k which is available during the integration process, but some approximation of the error

$$(2.3) \qquad \rho'_k(\tau_k) = \tilde{U}'_{k+1} - u_{k+1},$$

where $\tilde{U}'_{k+1} = \tilde{U}'\ (t_k+\tau_k)$, $\tilde{U}'(t)$ being the local analytical solution through the point (t_k,u_k). We shall call $\rho'_k(\tau_k)$ the discrepancy. In most cases the local error $\rho_k(\tau_k)$ and the discrepancy are comparable. However, when the Jacobian of the differential equation has eigenvalues with a large negative real part, considerable differences may occur when the eigenfunctions corresponding to these "late" eigenvalues vanish rapidly in the analytical solution and when these eigenfunctions should continue to be present in the numerical solution. In such cases one may improve the approximation by using strongly stable methods in order to reduce the effect of the "late" eigenfunctions (compare section 3).

The discrepancy $\rho'_k(\tau_k)$ can be expressed as

$$(2.4) \qquad \rho'_k(\tau_k) = (1-\beta_1)\ \tau_k\ c_k^{(1)} + (\tfrac{1}{2} - \beta_2)\ \tau_k^2\ c_k^{(2)} + \dots + (\tfrac{1}{n!} - \beta_n)\ \tau_k^{(n)}\ c_k^{(n)} + O(\tau_k^{n+1}).$$

Let p be the order of accuracy of the method, i.e.

$$\rho'_k(\tau) = O(\tau^{p+1}) \qquad \text{as } \tau \to 0,$$

and let $p < n$. Then we take as a measure for the discrepancy the value of

$$(2.5) \qquad ||(1-\beta_1)\ \tau_k c_k^{(1)} + \dots + (\tfrac{1}{p!} - \beta_p)\ \tau_k^p\ c_k^{(p)}|| + \left|\tfrac{1}{(p+1)!} - \beta_{p+1}\right|\ \tau_k^{p+1}\ ||c_k^{(p+1)}|| + \dots + \left|\tfrac{1}{n!} - \beta_n\right|\ \tau_k^n\ ||c_k^{(n)}||.$$

In the case of the modified Taylor method the first term of this expression vanishes and, generally, the discrepancy is mainly determined by the second term. However, when $||c_k^{(p+1)}||$ has a zero near $t = t_k$, the third

term is a measure for the discrepancy. In scalar cases such a situation is not fictitious. This means that it is dangerous to approximate the discrepancy by only the first neglected term. Of course, when $n = p+1$ we have no choice and must hope for the best. Note that we have taken the sum of norms of neglected terms instead of the norm of the sum of these terms in order to be safe for zeroes of the sum function.
Furthermore, it may be remarked that formula (2.5) generally holds for methods with varying coefficients as discussed in [4].

Next we consider the case $p = n$. We then approximate the discrepancy by the value of the last correction term:

$$(2.5') \qquad \frac{1}{n!}\, \tau_k^n \, ||c_k^{(n)}||.$$

Procedure local error construction sums step by step the terms given in (2.5). Since the vectors $c_k^{(i)}$ are computed in procedure difference scheme, local error construction is called in difference scheme each time a new vector $c_k^{(i)}$ is computed.

2.4 Procedure local error bound

Procedure local error bound assigns a prescribed value to η_k, the local error bound to be imposed on the scheme. By putting

$$(2.6) \qquad \eta_k = \eta_{abs} + \eta_{rel}\, ||u_k||,$$

where η_{abs} and η_{rel} are respectively the absolute and relative error bound for the local error, we have a flexible formula for η_k.

2.5 Procedure stepsize

The step τ_k is determined both by accuracy conditions and stability conditions. First we consider the accuracy conditions.

Ideally, the step τ_k should be such that

$$(2.7) \qquad ||\rho_k'(\tau_k)|| = \eta_k.$$

As we have seen in preceding sections, however, the value of $||\rho_k'(\tau_k)||$ is unknown until the integration step τ_k is completed. Thus, when the integration process is arrived at the point $t = t_k$ we only know the values of $||\rho_j'(\tau_j)||$ for $0 \leq j \leq k-1$. Therefore, some strategy for predicting τ_k is necessary, which is based on the values of preceding discrepancies. For instance, let it be assumed that in the neighbourhood of (τ_k, t_k) the error $||\rho'||$, as a function of τ and t, behaves as

$$(2.8) \qquad ||\rho'|| = f(\tau,t,A,B,C,\ldots),$$

where f is a given function and A, B, C, ... are parameters to be determined by the equations

$$(2.9) \qquad f(\tau_j,t_j,A,B,C,\ldots) = ||\rho_j'(\tau_j)||, \qquad j = k-1,\ k-2,\ \ldots\ .$$

Then, the new step τ_k may be predicted by solving the equation

$$(2.10) \qquad f(\tau_k,t_k,A,B,C,\ldots) = \eta_k.$$

In the case of constant coefficients we have the error formula (see (2.5) and (2.5'))

$$(2.5'') \qquad ||\rho_k'(\tau_k)|| \sim \tau_k^q\Big[\Big|\frac{1}{q!} - \beta_q\Big|\ ||c_k^{(q)}|| + \ldots + \Big|\frac{1}{n!} - \beta_n\Big|\ \tau_k^{n-q}\ ||c_k^{(n)}||\Big],$$

where q = n if p = n and q = p+1 otherwise. We have considered the following representations of $||\rho'||$:

$$(2.8a) \qquad ||\rho'|| = C\ \tau^q, \qquad k = 1,$$

$$(2.8b) \qquad ||\rho'|| = (Bt+C)\tau^q, \qquad k = 2,$$

$$(2.8c) \qquad ||\rho'|| = (At^2+Bt+C)\tau^q, \qquad k \geq 3.$$

In these representations the dependence on the step τ of the factor between brackets, the "error constant", is neglected in the neighbourhood of (τ_k, t_k). Note that in this case equation (2.10) can be solved explicitly for τ_k, provided that the representation of the error constant is positive for $t = t_k$. If not we have put $\tau_k = \tau_{k-1}$.

When the coefficients β_j depend on τ we may choose the representation

$$(2.8d) \qquad ||\rho'|| = (A\tau+Bt+C)\tau^q, \qquad k \geq 3,$$

where q is some function of τ tending to n (if p=n) or p + 1 (if p<n) as $\tau \to 0$. Since we only consider polynomial methods with constant coefficients in this section, we postpone the discussion of (2.8d) to section 6.5.

The first step τ_0 cannot be predicted by formulae of type (2.8) since no estimates of the local error at preceding points are available. We have used the rather rough formula

$$(2.11) \qquad \tau_0 = \frac{\eta_0}{||c_0^{(1)}||} .$$

This step corresponds to monitoring the discrepancy of a zero order scheme and is expected to be sufficiently small. The next step then follows from (2.8a), i.e.

$$\tau_1 = \sqrt[q]{\frac{\eta_1}{||\rho_0'(\tau_0)||}}\,\tau_0 = \frac{\eta_0}{||c_0^{(1)}||}\;\sqrt[q]{\frac{\eta_1}{||\rho_0'(\tau_0)||}} .$$

However, it may happen that $||\rho_0'(\tau_0)||$ is so small that τ_1 is assigned a value much too large to be acceptable. We have allowed τ_1 to increase with a factor 10 with respect to τ_0 until a realistic step length is reached. We shall call a step length realistic when the last computed discrepancy is larger than 10^{-q} * tolerance (when the next step is again 10 larger than the preceding one, we expect that the discrepancy will exceed the tolerance). In this phase of the integration process, where procedure stepsize is searching for an appropriate step, we used the simple extrapolation formula (2.8a). As soon as procedure stepsize reaches a realistic step length the next step shall be based on (2.8b) and thereafter on (2.8c). In this phase of the integration process the step sizes are subject to the conditions

(2.12) $\frac{1}{2}\tau_{k-1} \leq \tau_k \leq \text{alfa} * \tau_{k-1}$,

where alfa is prescribed by the user of the program.

When procedure modified taylor is called with $k \neq 0$ the step size prediction at once uses (2.8c), since the necessary error constants are already supposed to be evaluated in the first call of modified taylor.

Having established a step size which satisfies the accuracy conditions we consider a second important condition, the stability condition of the scheme. From the theory presented in [3] and [4] it follows that polynomial methods give rise to inequalities of the type

(2.13) $\tau_k \leq \frac{\beta(n)}{\sigma(D_k)}$,

where $\beta(n)$ is the stability parameter of the generating polynomial and $\sigma(D_k)$ the spectral radius of the Jacobian matrix D_k of the equation to be solved.

It may happen that

$$\frac{\beta(n)}{\sigma(D_k)} < \varepsilon t,$$

where ε is the relative precision of the computer used ($\varepsilon \sim 10^{-12}$ for the EL X8). In such cases the integration variable is not changed so that further calculations are meaningless. When this situation arises procedure stepsize jumps to the end of procedure modified taylor.

2.6 Procedure coefficient

This procedure calculates the coefficients $\beta_1, \ldots, \beta_n$ and the parameter q.

2.7 Real procedure normfunction (norm, w)

If the parameter norm has the actual value 1 this procedure assigns the maximum norm of the array w to normfunction. If norm = 2 the Euclidean norm of w is calculated.

3. A single ordinary differential equation

Our first experiments were done in order to illustrate the advantage of a variable step length, the consequences of weakly and strongly stable schemes and the difficulties which arise when an equation becomes stiff.

3.1 The initial value problem $\dot{U} = -e^tU+e^t\ln t+1/t$

Consider the following initial value problem

$$(3.1)\qquad \begin{cases} \dfrac{dU}{dt} = -e^tU + e^t\ln t + \dfrac{1}{t}, & t \geq 10^{-2}, \\[2ex] U(10^{-2}) = \ln(10^{-2}). & \end{cases}$$

Clearly, this problem has the solution

$$(3.2)\qquad \tilde{U}(t) = \ln t.$$

3.2 The difference scheme

Suppose that one decides to solve (3.1) by a Taylor method. Which method depends on the accuracy desired and the range over which the solution is to be computed. We have considered the following generating polynomials (see [3], section 4.1, 6.2, 6.3 and [5] , table 4.1):

$$(3.3)\qquad P_4(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4,$$

$$(3.4)\qquad P_4(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + .0184557\ z^4,$$

$$(3.5)\qquad P_4(z) = 1 + z + \frac{1}{2}z^2 + .0786845\ z^3 + .00360845\ z^4,$$

$$(3.6)\qquad P_4(z) = 1 + z + \frac{5}{32}z^2 + \frac{1}{128}z^3 + \frac{1}{8192}z^4.$$

These polynomials generate difference schemes which are fourth, third, second and first order exact, respectively and are appropriate for the numerical integration of equations of which the Jacobian matrix (in this case the scalar $-e^t$) has negative eigenvalues. For such equations the approximate values of the stability parameters $\beta(4)$ are, respectively,

(3.7) 2.78, 6, 12, 32.

These values complete the data necessary to define the integration method to be used. In the actual program the data are stored in array data [-2:4]. For instance, the method corresponding to (3.3) is defined by

```
data [-2]:= 4;
data [-1]:= 4;
data  [0]:= 2.78;
data  [1]:= 1;
data  [2]:=  .5;
data  [3]:=  .166667;
data  [4]:=  .04166667.
```

Next, we have to consider the ALGOL 60 description of the initial value problem. We restrict our considerations to procedure derivative:

```
procedure derivative (i,a); integer i; array a;
begin if i = 1 then
    begin expt:= exp(t); lnt:= ln(t); c0:= a[0];
       c1:= a[0]:=-expt * c0 + 1/t + expt * lnt
    end;
    if  i = 2 then c2:= a[0]:= expt * (lnt+1/t-c0-c1) - 1/t/t;
    if  i = 3 then c3:= a[0]:= expt * (lnt+2/t-c0-2*c1-c2-1/t/t) + 2/t/t/t;
    if  i = 4 then a[0]:= c3 - 2 * (1+3/t)/t/t/t + expt * ((1-(2-2/t)/t-c1-2*c2-c3)
end;
```

Here, the variables expt, lnt, c0, c1, c2 and c3 are to be declared at the beginning of the program. Note that c0, c1, c2 and c3 correspond to the quantities $c_k^{(i)}$, i = 0, 1, 2, 3, occurring in procedure difference scheme.

Finally, we give the actual call of procedure modified taylor by which the results listed in tables 3.1 - 3.4 were produced:

k:= 0;
modified taylor (t, if k < 200 then t else 8, 0, 0, u, exp(t), i,
derivative, k, data, 1.2, 1, 10^{-5}, 10^{-4}, eta, rho, output);

3.3 Variation of the step size

In tables 3.1 - 3.4 the results at the points t = t_k, k = 0, 10, 20, ..., which were obtained by the methods generated by (3.3) - (3.6), are listed. We have respectively given the step number k, the value t_k of the integration variable, the step size τ_k, the maximal step allowed by stability, the tolerance η_k divided by the discrepancy $|\rho_k'|$, the global discretization error ε_k and its maximal absolute value $||\varepsilon||_\infty$.

Table 3.1 Fourth order Taylor method

k	t_k	τ_k	τ_{stab} $=2.78 \exp(-t_k)$	$\eta_k/\|\rho_k'\|$	$\varepsilon_k=\tilde{U}_k-u_k$	$\|\|\varepsilon\|\|_\infty$
0	.010	$5\ 10^{-6}$	2.78	$4\ 10^{10}$	0	$3.4\ 10^{-4}$
10	.041	.006	2.78	2.25	$-2.5\ 10^{-4}$	
20	.165	.024	2.38	1.90	$-3.4\ 10^{-4}$	
30	.572	.067	1.57	1.49	$-2.4\ 10^{-4}$	
40	1.474	.165	.19	1.18	$-2.4\ 10^{-5}$	
50	3.340	.099	.099	59	$-6.6\ 10^{-7}$	
60	4.043	.049	.049	776	$-7.1\ 10^{-8}$	
70	4.447	.033	.033	3135	$-1.9\ 10^{-8}$	
...	...	...	...	...	...	
200	6.107	.006	.006	$2\ 10^{6}$	$-4.7\ 10^{-11}$	

Table 3.2 Third order Taylor method with one stability term

k	t_k	τ_k	$\tau_{stab} = 6\exp(-t_k)$	$\eta_k/\lvert\rho'_k\rvert$	$\varepsilon_k=\tilde{U}_k-u_k$	$\lVert\varepsilon\rVert_\infty$
0	.010	$5\ 10^{-6}$	5.94	$7\ 10^{10}$	0	$1.7\ 10^{-3}$
10	.045	.008	5.76	2.78	$+1.2\ 10^{-3}$	
20	.207	.037	4.86	2.24	$+1.7\ 10^{-3}$	
30	.764	.088	2.82	1.57	$+9.2\ 10^{-4}$	
40	2.272	.333	.61	1.71	$+4.3\ 10^{-5}$	
50	4.018	.071	.11	.88	$-3.1\ 10^{-5}$	
60	4.570	.042	.060	1.01	$-2.8\ 10^{-5}$	
70	4.926	.030	.044	1.00	$-2.8\ 10^{-5}$	
80	5.189	.023	.033	1.00	$-2.7\ 10^{-5}$	
...	...	...	...	...	...	
200	6.530	.006	.009	1.00	$-2.7\ 10^{-5}$	

In all four tables we see a rapidly varying step size τ_k. This variation is due to the accuracy condition as well as the stability condition imposed on the difference scheme.

In the first part of the integration interval the step size prediction is governed by the accuracy condition

(3.8) $$\tau \leq \tau_{acc} = \sqrt[q]{\frac{\eta}{E_p}} ,$$

where q is the lowest power of τ occurring in the formula for the discrepancy ρ' and E_p is the extrapolated (predicted) value of the error constant E. For instance, the fourth and third order method respectively have the discrepancies

(3.9) $$\lvert\rho'_k(\tau_k)\rvert = \frac{1}{24}\tau_k^4 \lvert c_k^{(4)}\rvert \sim .042\ \tau_k^4 \lvert c_k^{(4)}\rvert$$

and

(3.10) $$|\rho_k'(\tau_k)| = (\tfrac{1}{24} - .0184557)\tau_k^4\,|c_k^{(4)}| \sim .023\,\tau_k^4\,|c_k^{(4)}|,$$

from which we conclude that $q = 4$ and E behaves like $|c_k^{(4)}|$. The value of $|c_k^{(4)}|$ is defined by the fourth derivative of the local analytical solution $\tilde{U}'(t)$, i.e.

(3.11) $$c_k^{(4)} = \frac{d^4}{dt^4}\tilde{U}'(t)\Big|_{t=t_k} = \frac{d^4}{dt^4}[\ln t + Ce^{-e^t}]\Big|_{t=t_k},$$

where C is some constant (we have in fact $\varepsilon_k = -Ce^{-e^{t_k}}$). When the global error ε is negligible with respect to the analytical solution $\tilde{U}(t) = \ln t$ we have $\tilde{c}_k^{(4)} = c_k^{(4)} = -6/t_k^4$ so that the analytical error constant $\tilde{E}$ behaves as t^{-4}. This implies a linear increase of τ_{acc} for the fourth and third order method as may be concluded from formula (3.8) (a similar conclusion holds for the second and first order method). However, the presence of the error ε, which is of the form $-Ce^{-e^t}$, introduces into the error constant a component which is increasing with t so that, in practice, τ_{acc} will have a lower rate of increase and will finally decrease when the increasing component becomes dominant.

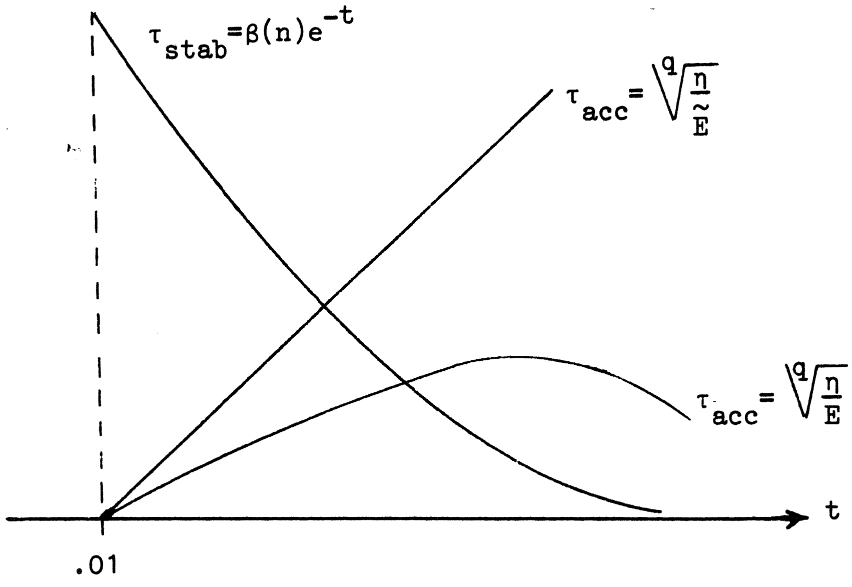

fig. 3.1 Analytical and numerical behaviour of the stepsize τ

Table 3.3 Second order Taylor method with two stability terms

k	t_k	τ_k	$\tau_{stab} = 12 \exp(-t_k)$	$\eta_k/\lvert\rho'_k\rvert$	$\varepsilon_k=\tilde{U}_k-u_k$	$\lVert\varepsilon\rVert_\infty$
0	.010	$5\ 10^{-6}$	11.9	$3\ 10^{7}$	0	$4.6\ 10^{-3}$
10	.021	.003	11.8	1.55	$-1.1\ 10^{-3}$	
20	.062	.007	11.3	1.14	$-3.3\ 10^{-3}$	
30	.165	.016	10.2	1.09	$-4.5\ 10^{-3}$	
40	.389	.031	8.2	1.05	$-4.3\ 10^{-3}$	
50	.770	.044	6.3	1.02	$-2.6\ 10^{-3}$	
60	1.227	.067	3.5	1.01	$-8.4\ 10^{-4}$	
70	2.292	.158	1.2	1.01	$-9.1\ 10^{-5}$	
80	3.588	.091	.33	.99	$-2.0\ 10^{-5}$	
90	4.312	.055	.16	1.00	$-1.0\ 10^{-5}$	
100	4.776	.038	.096	1.00	$-7.1\ 10^{-6}$	
110	5.117	.030	.072	1.00	$-4.9\ 10^{-6}$	
120	5.434	.046	.054	2.08	$-3.6\ 10^{-7}$	
130	5.768	.026	.036	6.48	$-9.2\ 10^{-8}$	
140	6.072	.028	.028	17.11	$-1.3\ 10^{-8}$	
150	6.319	.022	.022	35.07	$-5.5\ 10^{-9}$	
160	6.517	.018	.018	70.13	$-2.8\ 10^{-9}$	
...	...	...	...	...	...	
200	7.056	.010	.010	466	$-4.6\ 10^{-9}$	

From figure 3.1, where the observations given above are illustrated, we see that analytically the step size linearly increases until the stability condition

$$(3.12) \qquad \tau \le \tau_{stab} = \beta(n)\ e^{-t}$$

becomes more severe than the accuracy condition (3.8), after which the step size decreases exponentially. In this region the equation is said to have a stiff behaviour. The fourth order method (see table 3.1) exhibits approximately such a behaviour of the step size. The third and second order method, however, show some irregularities which will be explained in section 3.5. The steps of the first order method are completely controlled by accuracy requirements because of the small integration interval covered in 200 steps.

Table 3.4 Fourth degree Chebyshev polynomial

k	t_k	τ_k	$\tau_{stab} = 32\exp(-t_k)$	$\eta_k/\lvert\rho_k'\rvert$	$\varepsilon_k=\tilde{U}_k-u_k$	$\lVert\varepsilon\rVert_\infty$
0	.010	$5\ 10^{-6}$	31.68	6000	0	$2.6\ 10^{-2}$
10	.013	.0004	31.59	.99	$+2.6\ 10^{-3}$	
20	.018	.0006	31.43	1.00	$+6.6\ 10^{-3}$	
30	.025	.0008	31.21	1.00	$+1.0\ 10^{-2}$	
40	.034	.0011	30.93	1.00	$+1.4\ 10^{-2}$	
50	.047	.0014	30.53	1.00	$+1.7\ 10^{-2}$	
60	.062	.0018	30.08	1.00	$+1.9\ 10^{-2}$	
70	.082	.0022	29.48	1.00	$+2.1\ 10^{-2}$	
80	.106	.0028	28.78	1.00	$+2.3\ 10^{-2}$	
90	.136	.0033	27.93	1.00	$+2.4\ 10^{-2}$	
100	.172	.0040	26.94	1.00	$+2.5\ 10^{-2}$	
110	.214	.0046	25.84	1.00	$+2.6\ 10^{-2}$	
120	.263	.0053	24.60	1.00	$+2.5\ 10^{-2}$	
130	.319	.0060	23.26	1.00	$+2.5\ 10^{-2}$	
...	...	...	...	...	...	
200	.835	.0074	13.88	1.00	$+1.2\ 10^{-2}$	

3.4 Accuracy

In the region governed by the accuracy condition ($\tau = \tau_{acc} < \tau_{stab}$) one would expect to find approximately equal values for the tolerance and discrepancy, i.e. $\eta_k/\lvert\rho_k'\rvert \sim 1$.

Except for the first order method (table 3.4) this is, at least initially, not in agreement with the numerical results. The reason is that the extrapolation process, by which the error constant E is predicted, cannot follow the rapid decrease of E. Hence, procedure step size predicts values for E which are larger that the real ones, so that, according to formula (3.8), the step sizes turn out slightly lower than when we had known E in advance. As a consequence we find $\eta_k/|\rho_k'| > 1$. As soon as ρ', as a function of t, slows down the prediction of τ_k becomes more accurate so that we find $\eta_k/|\rho_k'| \sim 1$.

In the region governed by the stability condition ($\tau = \tau_{stab}$) we have, of course, $\eta_k/|\rho_k'| >> 1$.

Next we consider the global error ε_k in the region governed by condition (3.8). As was pointed out in reference [3], section 2.2, the error ε_k satisfies the relation

$$(3.13) \qquad \varepsilon_{k+1} = P_4(\tau_k\, e^{t_k})\, \varepsilon_k + \rho_k,$$

where ρ_k is the local discretization error. For instance, the fourth and third order methods have local errors which are given by

$$(3.9') \qquad \rho_k(\tau_k) = \frac{1}{120}\, \tau_k^5\, \tilde{c}_k^{(5)} + O(\tau_k^6) = .2\, \tau_k^5\, t_k^{-5} + O(\tau_k^6)$$

and

$$(3.10') \qquad \rho_k(\tau_k) = (\frac{1}{24} - .0184557)\, \tau_k^4\, \tilde{c}_k^{(4)} + O(\tau_k^5) \sim -\ .14\, \tau_k^4\, t_k^{-4} + O(\tau_k^5),$$

respectively. These formulae, together with (3.9) and (3.10), explain why the fourth order method yields more accurate results than the third order method, although both methods are applied with the same values for the tolerances. For the integration process tries to keep $|\rho_k'(\tau_k)|$ equal to η_k which results in step sizes differing by a factor $\sqrt[4]{\frac{.041}{.023}} \sim 1.2$, i.e. 20%. Thus, the fourth order method chooses smaller steps and has a higher order local discretization error. This implies a smaller global error ε_k.
In connection with this it may be remarked that the third and second order method yield errors of comparable order of magnitude. This is due to the

fact that both methods use the first neglected terms (instead of the last correction terms) of the local Taylor expansions as a measure of the discrepancies.

Finally, we discuss the behaviour of ε_k in the region controlled by stability. In this region (3.13) assumes the form

$$(3.13') \qquad \varepsilon_{k+1} = P_4\,(-\beta(4))\;\varepsilon_k + \rho_k.$$

In the case of the fourth order method we have

$$\varepsilon_{k+1} = P_4\,(-2.78)\;\varepsilon_k + \rho_k \sim \varepsilon_k + \rho_k.$$

Since ε_k is negative and ρ_k positive in the region where (3.13') is valid (cf. formula (3.9')), we may expect that ε_k increases so that initially $|\varepsilon_k|$ is decreasing. This is in agreement with the results listed in table 3.1. For the second order method we have

$$\varepsilon_{k+1} = P_4\,(-12)\;\varepsilon_k + \rho_k \sim .85\;\varepsilon_k + \rho_k.$$

Although both ε_k and ρ_k are negative in the region where (3.13') applies we may again expect a decreasing behaviour of $|\varepsilon_k|$ (cf. formula (3.10')). Note that the parameter $\beta(4) = 12$, which was taken from [3], formula (6.11), is slightly lower than the parameter calculated in [5], table 4.1, where we found the value $\beta(4) = 12.0464$. When this larger value was employed we should have found an increasing behaviour of $|\varepsilon_k|$ since then the relation

$$\varepsilon_{k+1} \sim \varepsilon_k + \rho_k$$

holds for ε_k.

3.5 Weak and strong stability

In the preceeding two subsections the results listed in table 3.1 - 3.4 were discussed. One phenomenon, however, namely the fact that the third order method never, and the second order method only after some "hesitation", reaches the region controlled by stability, was postponed to this subsection. This phenomenon is related to weak and strong stability. In this paper, a method is called strongly stable when the amplification factors of the scheme are all within the unit circle. In the present example the amplification factors are given by $P_4(-\tau_k e^{t_k})$. In figure 3.2 these factors are illustrated for the polynomials (3.4) - (3.7). From this figure it is seen that the fourth order method can be made strongly stable by requiring

$$0 < \tau_k < 2.78\, e^{-t_k},$$

i.e. by excluding neighbourhoods of $\tau = 0$ and $\tau = 2.78\, e^{-t_k}$. In the case of the other polynomials we may obtain strong stability at the cost of intermediate regions of forbidden τ values, which is, however, undesirable from a practical point of view. Before we give a construction of a strongly stable scheme we shall demonstrate the danger of weakly stable schemes.

Consider the third order method applied to example (3.1) for $t \geq 3$. For such values of t we derive from formula (3.10) and (3.11)

$$\rho_k'(\tau_k) \sim .023\, \tau_k^4\, e^{4t_k},$$

so that τ_k will behave as

$$\tau_k \sim \sqrt[4]{\frac{\eta_k}{.023|\varepsilon_k|}}\; e^{-t_k}.$$

(Note the different behaviour of the local discretization error $\rho_k(\tau_k) \sim -.14\, \tau_k^4\, t_k^{-4}$ and the discrepancy $\rho_k'(\tau_k) \sim .023\, \tau_k^4\, e^{t_k}$).
From table 3.2 and the relation $\eta_k \sim 10^{-5} + 10^{-4} \ln t_k$ we deduce that the coefficient of e^{-t_k} in this expression for τ_k increases from 3.74 at k = 50 to 4.25 at k = 200 and never reaches the value 6 of the stability parameter $\beta(4)$ (see (3.12)). One may ask why $|\varepsilon_k|$ remains almost constant in this region of the integration. In order to explain this consider figure 3.2. We see that the polynomial $P_4(-\tau_k e^{t_k})$, generating the third order method,

assumes values near -1 in the neighbourhood of $\tau_k\, e^{t_k} = 4.39$ or $\tau_k = 4.39\, e^{-t_k}$. For such step sizes we have

$$\varepsilon_{k+1} \sim -\varepsilon_k + \rho_k \sim -\varepsilon_k$$

since ρ_k may be neglected with respect to ε_k. This implies that $|\varepsilon_k|$ remains approximately constant, which is a direct consequence of the weak stability of the method.

This example clearly demonstrates the danger in using weakly stable methods. The temporary increase of the step sizes in table 3.3 is due to the same phenomenon as described above. But in this case the integration process did eventually succeed to pass the region where the amplification factors equal -1.

We now try to improve the weakly stabilized Taylor methods by constructing strongly stabilized methods. For the class of methods generated by the polynomials $T_n\,(1+z/n^2)$ we refer to [3], section 4.1. Here, we consider the third order method generated by (3.4). From figure 3.2 it is seen that $P_4(z)$ reaches its minimal value at $z \sim -4.39$. Suppose that the last coefficient is changed with the amount $d\,\beta_4$. Then we have

$$d\,P_4(-4.39) = (-4.39)^4\, d\,\beta_4 \sim 371\, d\,\beta_4.$$

As a rough estimate of the minimum of the new amplification factor we may take the value of

$$-1 + 371\, d\,\beta_4.$$

The stability parameter $\beta(4)$ of the new polynomial can be estimated from the relation

$$d\,\beta(4) = \frac{\beta^3(4)}{\frac{1}{2} - \frac{1}{3}\,\beta(4) + 3\,\beta_4\,\beta^2(4)}\; d\,\beta_4 \sim -438\; d\,\beta_4.$$

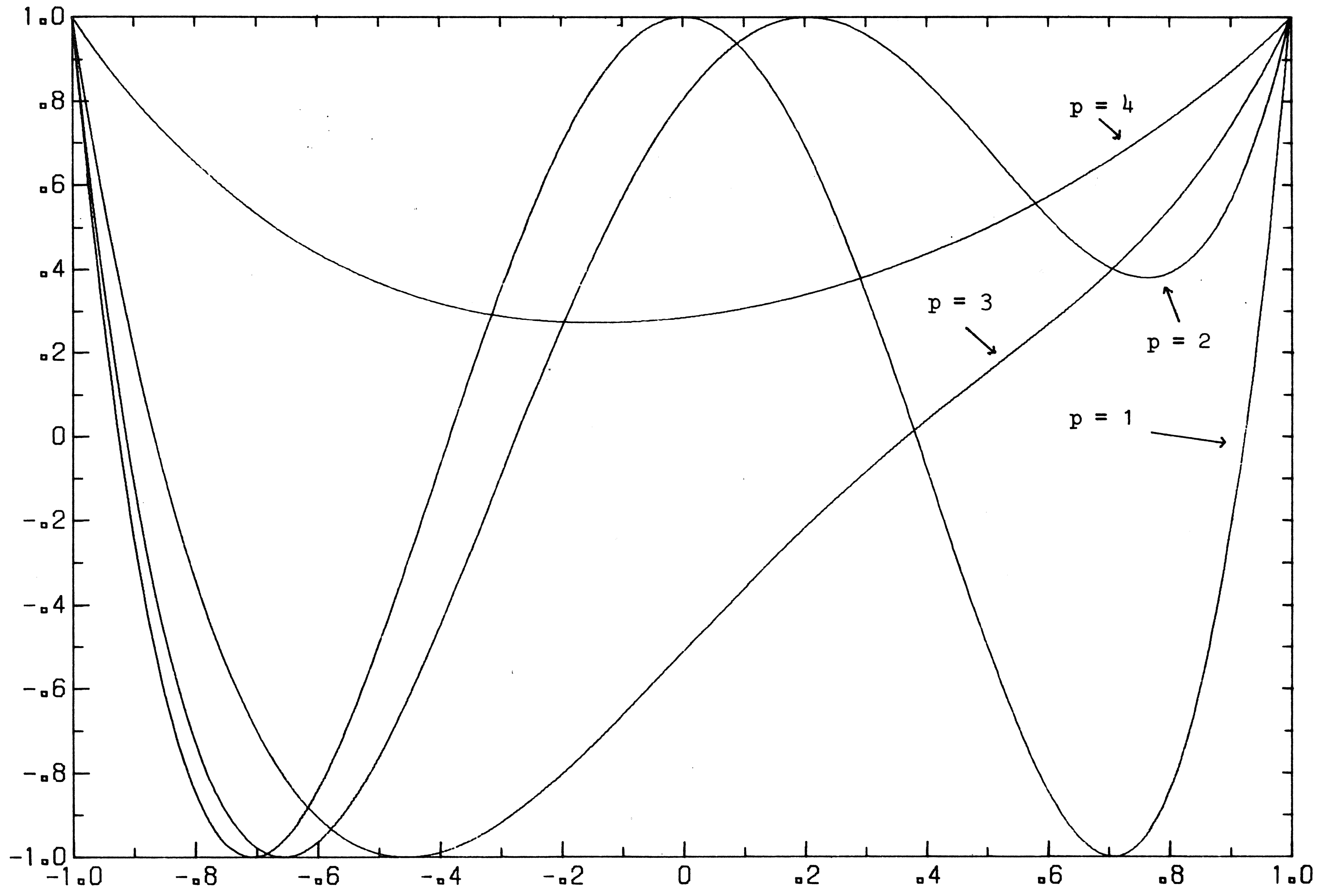

fig. 3.2 Amplification factors $P_4\ (-\tau_k\, e^{t_k})$ as functions of the variable $\frac{-2}{\beta(4)}\ \tau_k\, e^{t_k} + 1$.
p denotes the order of accuracy of the corresponding method.

From these relations one easily derives a polynomial which has, for instance, a minimal value of about .9. We found

(3.14) $$P_4(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + .01872597\ z^4,\ \beta(4) \sim 5.90.$$

In table 3.5 results obtained by this polynomial are given. The stability parameter $\beta(4)$ was given the value 5.8 in order to make sure that the amplification factors are within the interval [-1,1] for all step sizes.

Table 3.5 Strongly stable third order Taylor method with one stability term

k	t_k	τ_k	τ_{stab} $=5.80\ \exp(-t_k)$	$\eta_k/\lvert\rho_k'\rvert$	$\varepsilon_k=\tilde{U}_k-u_k$	$\lVert\varepsilon\rVert_\infty$
0	.010	$5\ 10^{-5}$	5.742	$7\ 10^{10}$	0	$1.6\ 10^{-3}$
10	.047	.0081	5.534	2.78	$+1.1\ 10^{-3}$	
20	.215	.033	4.678	2.23	$+1.6\ 10^{-3}$	
30	.790	.090	2.632	1.55	$+8.4\ 10^{-4}$	
40	2.385	.357	.534	1.91	$+4.0\ 10^{-5}$	
50	4.215	.086	.086	1.33	$+8.0\ 10^{-6}$	
60	4.846	.046	.046	2541	$+2.2\ 10^{-9}$	
70	5.228	.031	.031	15200	$+4.1\ 10^{-10}$	
...	...	...	...	...	...	
200	6.851	.006	.006	$1.9\ 10^{6}$	$+3.6\ 10^{-12}$	

The results of the first 40 integration steps closely resemble the results listed in table 3.2. After that, however, the strongly stable scheme quickly reaches the region where stability controls the step length.

Although the stabilized Taylor methods enable us to integrate equations of type (3.1) more efficiently than the standard Taylor methods, there are far more efficient integration methods. We mention the two- and three-cluster methods given in [4], section 4.4. In section 6 of the present paper an ALGOL 60 version of these methods will be given and applied to problem (3.1).

4. Hyperbolic differential equations

In this section we study the numerical solution of the Cauchy problem for some simple hyperbolic differential equations by the modified Taylor method.

4.1 The equation $U_t = \frac{1}{2}U_x$

Consider the Cauchy problem

$$(4.1)\quad \begin{cases} U_t = \frac{1}{2}U_x, & -\infty \leq x \leq \infty, \quad 0 \leq t < \infty, \\ U = \exp(-x^2), & -\infty \leq x \leq \infty, \quad t = 0. \end{cases}$$

By discretizing the variable x, that is replacing x by the discrete variable $j\xi$, $j = 0, \pm 1, \pm 2, \ldots$, where ξ is the mesh size, we may approximate equation (4.1) by an infinite set of ordinary differential equations:

$$(4.2)\qquad \frac{dU}{dt} = \frac{1}{4\xi}(X_+ - X_-)U.$$

Here, U denotes a vector with an infinite number of components corresponding to the grid points $j\xi$ and $X_\pm$ are shift operators with respect to the index j of the components of U. The operator occurring in the right hand side of (4.2) may be represented by a matrix D of infinite order i.e.

$$D = \frac{1}{4\xi}\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & & \\ \ldots, & 0, & -1, & 0, & +1, & 0, & \ldots & & \\ & \ldots, & 0, & -1, & 0, & +1, & 0, & \ldots & \\ & & \ldots, & 0, & -1, & 0, & +1, & 0, & \ldots \\ & & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & & \cdot & \cdot & \cdot & \cdot \end{bmatrix}.$$

It is easily verified that D has eigenfunctions E_ω of which the j-th component is given by

$$E_\omega^{(j)} = \exp(i\omega j\xi),$$

where ω is an arbitrary real number. The corresponding eigenvalues δ of D are given by

$$\delta = \frac{i}{2\xi} \sin \omega\xi. \tag{4.3}$$

Thus, D has purely imaginary eigenvalues with a spectral radius

$$\sigma(D) = \frac{1}{2\xi} . \tag{4.4}$$

In [3], section 5 the polynomials are given which generate difference schemes suitable for the integration of this type of differential equations. For instance, the polynomials

$$P_3(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{4}z^3, \qquad \beta(3) = 2, \tag{4.5}$$

$$P_4(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4, \ \beta(4) = 2\sqrt{2}. \tag{4.6}$$

These polynomials are second and fourth order exact, respectively. Hence, as can be easily verified, the analytical solution of (4.1) will locally satisfy the difference schemes generated by (4.5) and (4.6) apart from a term

$$O(\tau^3) + O(\tau\xi^2) \quad \text{and} \quad O(\tau^5) + O(\tau\xi^2), \tag{4.7}$$

respectively. Since the stability condition, which is of the form

$$\tau \leq \frac{\beta(n)}{\sigma(D)} = 2\ \beta(n)\xi, \tag{4.8}$$

allows time steps of order ξ we shall have an approximation error of at least order ξ^3. Note that the second degree polynomial given in [3], formula (5.1), which is only first order accurate, will yield a residual term $O(\tau^2) + O(\tau\xi^2)$, i.e. of order ξ^2.

In the actual application of the modified Taylor method to equation (4.2) we are faced with the fact that the number of components of U is infinite. However, when we wish the difference solution at the point $(j\xi, t_k)$ we only need the values of the difference solution at the points $(j\xi, t_{k-1})$, $((j\pm1)\xi, t_{k-1})$, ..., $((j\pm n)\xi, t_{k-1})$. This is illustrated in figure 4.1 for n = 4.

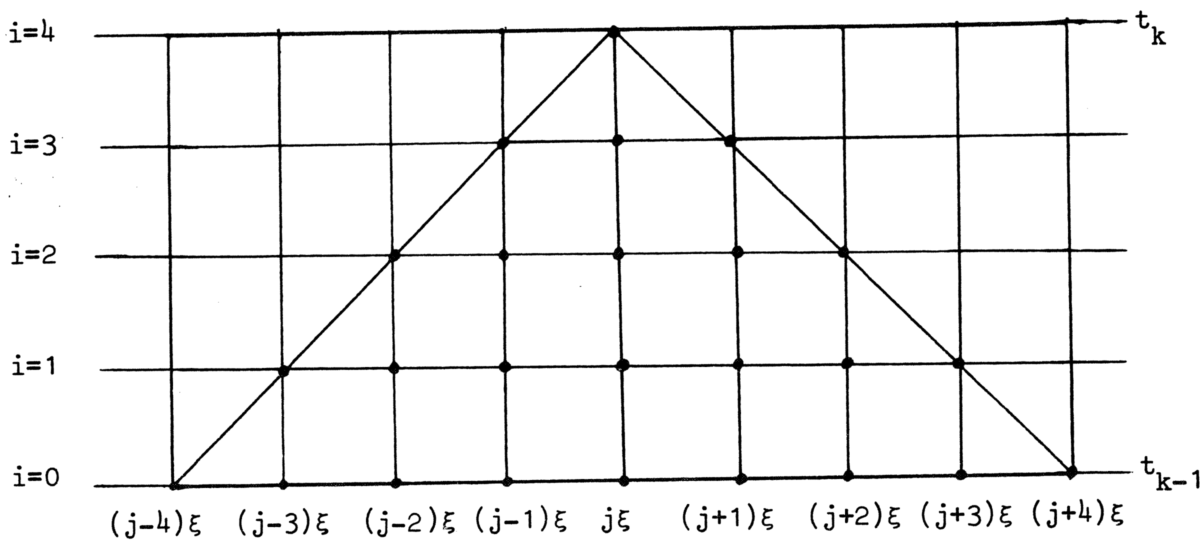

fig. 4.1 Domain of dependence for a fourth degree generating polynomial applied to equation (4.2)

Therefore, when we roughly know the number, say K, of steps necessary to integrate the initial value problem and when the solution is required at, say J points we have to start with a system of J + 2Kn differential equations. Moreover, during the integration process the number of relevant equations decreases, so that it is efficient to change the index m_0 of the first equation and the index m of the last equation. Let the initial function U_0 be specified at the points $x = j\xi$, $j = g_0, g_0+1, \ldots, g$. Then the indices m_0 and m should be defined by the procedure

```
integer procedure m0;
begin integer decrement;
      if k = 0 then decrement:= i else decrement:= i + (k-1) * data[-2];
      m0:= $g_0$ + decrement; m:= g - decrement
end;
```

Note that the use of this procedure avoids the repeated calculation of the value of m in the for statements occurring in modified taylor.

We are now in a position to give the ALGOL 60 version of procedure derivative.

```
procedure derivative (i,a); integer i; array a;
begin real ajm1, aj;
      ajm1:= a[m0-1];
      for j:= m0 step 1 until m do
      begin aj:= a[j];
            a[j]:= (a[j+1]-ajm1)/4/ksi;
            ajm1:= aj
      end
end;
```

Here, ksi denotes the mesh size ξ and must be specified before modified taylor is called.

Suppose that the solution is required in the region

$$R: [-.05 \leq x \leq .05] * [0 \leq t \leq .45].$$

Furthermore, let us take a mesh size

$$\xi = .0025,$$

then the solution is required at 40 gridpoints when t = .45. Condition (4.7) prescribes a maximal time step .010 and $.010\sqrt{2}$ for the polynomials (4.5) and (4.6). Therefore, by the argument given above we have to start with at least $40 + 2 * (.45/ .010) * 3 = 310$ and $40 + 2 * (.45/ .010\sqrt{2}) * 4 \simeq 296$ differential equations, respectively. However, when the accuracy condition is more restrictive than the stability condition we shall need a larger set of equations. In order to have a safety margin we took $g = -g_0 = 200$ for both polynomials. The values of the remaining parameters, such as t, te, u, data and sigma, need no further explanation.

Having specified all parameters of procedure modified taylor which characterize the problem to be solved and the method to be used, we arrive at the actual call of this procedure:

```
k:= 0;
modified taylor (t, if k ≥ 45 then t else .45, m0, m, u, 1/2/ksi,
                 i, derivative, k, data, 1.5, 2, 10^-3, 10^-2, eta,
                 rho, output);
```

In table 4.1 and 4.2 the results are listed for k = 5, 10, 15, ...

Table 4.1 Generating polynomial (4.5)

k	t_k	$\eta_k/\|\|\rho_k'\|\|_2$	$\|\|\varepsilon_k\|\|_R/\|\|\tilde{U}_k\|\|_R$
0	0	$3.3\ 10^5$	0
5	.05	$3.4\ 10^5$	$3.3\ 10^{-8}$
10	.10	$3.6\ 10^5$	$1.0\ 10^{-7}$
15	.15	$3.8\ 10^5$	$2.1\ 10^{-7}$
20	.20	$4.0\ 10^5$	$3.7\ 10^{-7}$
25	.25	$4.1\ 10^5$	$5.7\ 10^{-7}$
30	.30	$4.1\ 10^5$	$8.2\ 10^{-7}$
35	.35	$4.0\ 10^5$	$1.1\ 10^{-6}$
40	.40	$3.8\ 10^5$	$1.4\ 10^{-6}$
45	.45	$3.6\ 10^5$	$1.8\ 10^{-6}$

In these tables $|| \; ||_2$ denotes the Euclidean norm over the grid points $(j\xi, t_k)$, $m_0 \leq j \leq m$ and $|| \; ||_R$ denotes the Euclidean norm over the grid points $(j\xi, t_k)$, $-20 \leq j \leq 20$. The error ε_k is defined by the difference of the analytical solution $\tilde{U}(t) = \exp(-(x+\frac{1}{2}t)^2)$ of (4.1) and the numerical solution

$$\varepsilon_k^{(j)} = \exp[-(j\xi+\frac{1}{2}t_k)^2] - u[j]$$

where u[j] is the numerical solution at $(j\xi, t_k)$.

Table 4.2 Generating polynomial (4.6)

k	t_k	$\eta_k/\|\|\rho_k'\|\|_2$	$\|\|\varepsilon_k\|\|_R/\|\|\tilde{U}_k\|\|_R$
0	0	$1.1\ 10^7$	0
5	.07	$1.1\ 10^7$	$2.0\ 10^{-8}$
10	.14	$1.0\ 10^7$	$6.6\ 10^{-8}$
15	.21	$1.0\ 10^7$	$1.4\ 10^{-7}$
20	.28	$1.0\ 10^7$	$2.5\ 10^{-7}$
25	.35	$1.0\ 10^7$	$3.8\ 10^{-7}$
30	.42	$1.0\ 10^7$	$5.4\ 10^{-7}$
32	.448	$1.1\ 10^7$	$5.7\ 10^{-7}$

An analysis of table 4.1 and 4.2 reveals that the step size is completely governed by the stability condition (4.8). Ideally, the steps should be such that the errors due to discretization of x and t are approximately equal. From (4.7) it follows that in the case of the second order method (polynomial (4.5)) the approximation errors are comparable as $\tau, \xi \to 0$, hence we may expect the same for the discretization errors ε_k. In order to verify this consider the errors ε_k produced by the fourth order method (with respect to τ). These errors consist for the greater part of errors introduced by the x-discretization. Since the second order method uses the same x-discretization we may conclude for this method that the discretization error due to the t-discretization is slightly larger than the error due to the x-discretization. Thus the stability condition is not a restriction for the first integration process. The second integration process can be economized by using a more accurate discretization of the operator $\partial/\partial x$ in order to make the approximation errors of comparable order as $\tau, \xi \to 0$.

4.2 The equation $U_t = tU_x$

Consider the Cauchy problem

$$(4.10) \quad \begin{cases} U_t = tU_x, & -\infty \leq x \leq \infty,\ 0 \leq t \leq \infty, \\ U = \exp(x), & -\infty \leq x \leq \infty,\ t = 0, \end{cases}$$

with the analytical solution

$$(4.11) \quad \tilde{U} = \exp(x + \tfrac{1}{2}t^2).$$

This problem may be approximated by the infinite set of differential equations

$$(4.12) \quad \frac{dU}{dt} = \frac{t}{2\xi}(X_+ - X_-)U,$$

where U, ξ and $X_\pm$ are defined in the same manner as in the preceding subsection.

The eigenvalues of the Jacobian D of (4.12) are given by

$$(4.13) \quad \delta = i\,\frac{t}{\xi}\sin \omega\xi,$$

so that

$$(4.14) \quad \sigma(D) = \frac{t}{\xi}\ .$$

This results in the (local) stability condition

$$(4.15) \quad \tau \leq \frac{\beta(n)}{\sigma(D)} = \frac{\beta(n)}{t}\,\xi$$

For small values of t this condition allows large step sizes τ_k. However, for large values of τ_k the linearization on which (4.14) is based is not valid. Therefore, we must apply (4.15) very carefully. For instance, we could replace $\sigma(D)$ by the spectral radius of D at the point $t + \tau$, i.e.

$$(4.15') \qquad \tau \leq \frac{\beta(n)}{t+\tau}\, \xi .$$

Or equivalently,

$$(4.15'') \qquad \tau \leq \frac{1}{2}\left(\sqrt{t^2+4\,\beta(n)\xi} - t\right) \sim \begin{cases} \sqrt{\beta(n)\xi} & \text{for } t^2 \ll 4\,\beta(n)\xi \\ \frac{\beta(n)}{t}\xi & \text{for } t^2 \gg 4\,\beta(n)\xi . \end{cases}$$

From (4.7) and (4.15") it may be concluded that for small values of t the approximation error of the difference schemes generated by the polynomials (4.5) and (4.6) is of order $\xi^{3/2}$ and $\xi^{5/2}$, respectively. For larger values of t both polynomials yield an error of order ξ^3. Hence, it is expected that polynomial (4.6) shall yield more accurate results.

Finally, we have to discuss the derivatives of U defined by equation (4.12). Let us write it in the form

$$(4.12') \qquad \dot{U} = tD\,U, \quad D = \frac{1}{2\xi}(X_+ - X_1)$$

then it is easily verified that

$$\ddot{U} = tD\,\dot{U} + D\,U,$$

$$\dddot{U} = tD\,\ddot{U} + 2D\,\dot{U},$$

$$\ddddot{U} = tD\,\dddot{U} + 3D\,\ddot{U}.$$

The corresponding ALGOL 60 version is given by

```
procedure derivative (i,a); integer i; array a;
begin real vjm1, vj;
      v[m0-1]:= a[m0-1];
      if i = 1 then for j:= m0 step 1 until m do
      begin v[j]:= a[j]; a[j]:= (t/2/ksi) * (a[j+1]-v[j-1]) end;
      if i > 1 then
      begin vjm1:= v[m0-1];
            for j:= m0 step 1 until m do
            begin vj:= v[j];
                  v[j]:= a[j];
                  a[j]:= (t/2/ksi) * (a[j+1]-v[j-1]) +
                  (i-1)* (1/2/ksi) * (v[j+1]-vjm1);
                  vjm1:= vj
            end
      end
end;
```

Here, it is supposed that m0 and m are integer procedures defined by (4.9) and that v is a one dimensional array v[g0:g] declared at the beginning of the program. It may be remarked that this procedure can be written in a more efficient form, but we have preferred the formulation given above in order to keep things as simple as possible.

Suppose that the solution is required in the region

$$R: \{-.6 \leq x \leq .6\} * \{0 \leq t \leq 1.2\},$$

and let the mesh size ξ be given the value .02. Then, if the step size is completely determined by the stability conditions we would at least need $1.2 * 30 = 36$ and $1.2 * 30 * \frac{1}{2}\sqrt{2} \simeq 26$ steps for the polynomials (4.5) and (4.6), respectively (cf. condition (4.15')). However, if the values of η_a and η_r are small, the accuracy conditions will prescribe much smaller steps. For reasons of security we took, in all cases, the maximum number of points needed to accomplish 60 steps, i.e. $g = -g0 = 30 + 3 * 60 = 210$ and $30 + 4 * 60 = 270$, respectively. This completes the definition of the parameters characterizing the initial value problem and the method to be used.

We have done some experiments with increasing values of η_a and η_r:

$$(\eta_a,\eta_r) = (10^{-5},10^{-6}),\ (10-^{4},10^{-5}),\ (10^{-2},10^{-3}).$$

The actual call of modified taylor was as follows:

```
k:= 0;
modified taylor (t, if k > 60 then t else 1.2, m0, m, u,
                 2 * data[0]/(sqrt(t*t+4*data[0]*ksi)-t),
                 derivative, k, data, 1.5, 2, aeta, reta,
                 eta, rho, output);
```

In table 4.3 through 4.8 the results of these experiments are listed.

Table 4.3

Generating polynomial (4.5)

$(\eta_a,\eta_r) = (10^{-5},10^{-6})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$3.4\ 10^{+11}$	0
5	.11	1.0	$8.7\ 10^{-7}$
10	.26	1.0	$7.0\ 10^{-6}$
15	.38	1.0	$1.5\ 10^{-5}$
20	.48	1.0	$2.3\ 10^{-5}$
25	.58	1.0	$3.2\ 10^{-5}$
30	.68	1.0	$4.1\ 10^{-5}$
35	.77	1.0	$5.1\ 10^{-5}$
40	.85	1.0	$6.1\ 10^{-5}$
45	.93	1.0	$7.2\ 10^{-5}$
50	1.01	1.0	$8.2\ 10^{-5}$
55	1.08	1.0	$9.3\ 10^{-5}$
60	1.15	-	$1.1\ 10^{-4}$

Table 4.4

Generating polynomial (4.6)

$(\eta_a,\eta_r) = (10^{-5},10^{-6})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$1.3\ 10^{+16}$	0
5	.18	.9	$1.0\ 10^{-6}$
10	.44	1.0	$6.4\ 10^{-6}$
15	.69	1.0	$1.6\ 10^{-5}$
20	.91	1.0	$2.7\ 10^{-5}$
25	1.12	1.0	$4.1\ 10^{-5}$
27	1.20	-	$4.7\ 10^{-5}$

Table 4.5

Generating polynomial (4.5)

$(\eta_a,\eta_r) = (10^{-4},10^{-5})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$3.4\ 10^{+9}$	0
5	.25	1.1	$1.1\ 10^{-5}$
10	.50	1.0	$6.6\ 10^{-5}$
15	.70	1.0	$1.3\ 10^{-4}$
20	.88	1.0	$1.9\ 10^{-4}$
25	1.05	1.0	$2.5\ 10^{-4}$
30	1.20	-	$3.3\ 10^{-4}$

Table 4.6

Generating polynomial (4.6)

$(\eta_a,\eta_r) = (10^{-4},10^{-5})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$1.3\ 10^{+13}$	0
5	.31	3.0	$.3\ 1'^{-5}$
10	.75	1.5	$1.6\ 10^{-5}$
15	1.05	3.5	$3.4\ 10^{-5}$
18	1.20	-	$4.5\ 10^{-5}$

Table 4.7

Generating polynomial (4.5)

$(\eta_a,\eta_r) = (10^{-2},10^{-3})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$3.4\ 10^{+5}$	0
5	.50	$1.9\ 10^{+1}$	$.26\ 10^{-5}$
10	.80	$4.0\ 10^{+1}$	$1.7\ 10^{-4}$
15	1.01	$5.7\ 10^{+1}$	$2.8\ 10^{-4}$
20	1.20	-	$3.7\ 10^{-4}$

Table 4.8

Generating polynomial (4.6)

$(\eta_a,\eta_r) = (10^{-2},10^{-3})$

k	t_k	$\eta_k/\|\|\rho_k\|\|_2$	$\|\|\varepsilon_k\|\|_R$
0	0	$1.3\ 10^{+7}$	0
5	.59	$7.8\ 10^{+1}$	$.4\ 10^{-5}$
10	.94	$2.7\ 10^{+2}$	$2.0\ 10^{-4}$
15	1.20	-	$3.8\ 10^{-4}$

5. Parabolic differential equations

In the preceding section partial differential equations were considered of which the Jacobian possessed purely imaginary eigenvalues. We now concentrate on a class of equations which have negative eigenvalues. We shall illustrate the application of modified taylor to such equations by two simple diffusion problems.

5.1 The equation $U_t = U_{xx} + e^{-t}(x^{10}+90x^8-x)$

Consider the initial boundary value problem

$$(5.1)\quad \begin{cases} U_t = U_{xx} + e^{-t}(x^{10}+90x^8-x), & 0 \le x \le 1, \quad t \ge 0, \\ U = 1 + x(1-x^9), & 0 \le x \le 1, \quad t = 0, \\ U = 1, & x = 0\ ,\ x = 1, \quad t \ge 0. \end{cases}$$

In reference [7] some results are given obtained by applying the modified Taylor method. Here, a more detailed discussion will be given how these results were obtained. Problem (5.1) is solved by the function

$$(5.2)\qquad \tilde{U} = 1 + e^{-t}x(1-x^9).$$

Let us define the operator

$$(5.3)\qquad D^{(i)} = a(X_+^2 + X_-^2) + b(X_+ + X_-) + c,$$

where $X_\pm$ are the usual shift operators with respect to the mesh size ξ and a, b and c are weight parameters to be determined in such a way that D approximates the operator $\partial^2/\partial x^2$ as $\xi \to 0$. A simple calculation yields the following expansion for the operator D:

$$(5.3')\qquad D^{(i)} = (2a+2b+c) + (4a+b)\ \xi^2 \frac{\partial^2}{\partial x^2} + \frac{1}{12}(16a+b)\ \xi^4 \frac{\partial^4}{\partial x^4} + $$
$$+ \frac{1}{360}(64a+b)\ \xi^6 \frac{\partial^6}{\partial x^6} + \dots .$$

From this representation of $D^{(i)}$ it follows that we have a first order exact approximation of $\partial^2/\partial x^2$ if

$$(5.4)\qquad \begin{cases} 2a + 2b + c = 0, \\ (4a+b)\,\xi^2 \;= 1 \end{cases}$$

and a third order exact approximation if, in addition,

$$(5.5)\qquad 16a + b = 0.$$

The operator $D^{(i)}$ can be applied in the grid points $(j\xi, t_k)$, $j = 2, 3, \ldots, \xi^{-1} - 2$. At the point (ξ, t_k) we define the operator

$$(5.6)\qquad D^{(b)} = a'X_- + b' + c'X_+ + d'X_+^2 + e'X_+^3 + f'X_+^4$$

and at the point $(1-\xi, t_k)$ a similar operator, which is obtained when $X_\pm$ is replaced by $X_\mp$. This operator may be represented by the series

$$(5.6')\qquad D^{(b)} = (a'+b'+c'+d'+e'+f') + (-a'+b'+2d'+3e'+4f')\xi\,\frac{\partial}{\partial x} +$$

$$+ \frac{1}{2}(a'+c'+4d'+9e'+16f')\xi^2\,\frac{\partial^2}{\partial x^2} +$$

$$+ \frac{1}{6}(-a'+c'+8d'+27e'+64f')\xi^3\,\frac{\partial^3}{\partial x^3} +$$

$$+ \frac{1}{24}(a'+c'+16d'+81e'+256f')\xi^4\,\frac{\partial^4}{\partial x^4} +$$

$$+ \frac{1}{120}(-a'+c'+32d'+243e'+1024f')\xi^5\,\frac{\partial^5}{\partial x^5} + \ldots .$$

We have a first order exact approximation of $\partial^2/\partial x^2$ at the boundary points if

$$(5.7)\quad \begin{cases} a' + b' + c' + d' + e' + f' = 0. \\ -a' + c' + 2d' + 3e' + 4f' = 0 \\ (a'+c'+4d'+9e'+16f')\xi^2 = 2, \\ -a' + c' + 8d' + 27e' + 64f' = 0 \end{cases}$$

and a third order exact approximation if, in addition,

$$(5.8)\quad \begin{cases} a' + c' + 16d' + 81e' + 256f' = 0, \\ -a' + c' + 32d' + 243e' + 1024f' = 0. \end{cases}$$

Problem (5.1) can now be approximated by an initial value problem for the system of ordinary differential equations

$$(5.9)\qquad \dot{U} = DU + F,$$

where

$$D = \begin{bmatrix} b' & c' & d' & e' & f' & 0 & \cdot & \cdot & \cdot & 0 \\ b & c & b & a & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ a & b & c & b & a & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & a & b & c & b & a & 0 & \cdot & \cdot & 0 \\ \vdots & & \ddots & & \ddots & & \ddots & & & \vdots \\ 0 & \cdot & \cdot & a & b & c & b & a & 0 & 0 \\ \vdots & & & & \ddots & & \ddots & & \ddots & \vdots \\ 0 & \cdot & \cdot & \cdot & 0 & a & b & c & b & a \\ 0 & \cdot & \cdot & \cdot & \cdot & 0 & a & b & c & b \\ 0 & \cdot & \cdot & \cdot & 0 & f' & e' & d' & c' & b' \end{bmatrix}$$

and

$$U = \begin{bmatrix} U^{(1)} \\ U^{(2)} \\ \vdots \\ U^{(j)} \\ \vdots \\ U^{(m-1)} \\ U^{(m)} \end{bmatrix}, \quad F = e^{-t} \begin{bmatrix} \xi^{10}+90\xi^{8} & -\xi & +a'e^{t} \\ (2\xi)^{10}+90(2\xi)^{8} & -2\xi & +a\ e^{t} \\ (3\xi)^{10}+90(3\xi)^{8} & -3\xi & \\ \vdots & & \\ (j\xi)^{10}+90(j\xi)^{8} & -j\xi & \\ \vdots & & \\ ((m-1)\xi)^{10}+90((m-1)\xi)^{8} & -(m-1)\xi & +a\ e^{t} \\ (m\xi)^{10}+90(m\xi)^{8} & -m\xi & +a'e^{t} \end{bmatrix}$$

A simple calculation reveals that the set of values

$$(5.10) \qquad \begin{cases} a = 0, \quad b = \xi^{-2}, \ c = -2\xi^{-2}, \\ \\ a'=\xi^{-2}, \ b'=2\xi^{-2}, \ c'=\xi^{-2}, \ d'=e'=f'=0, \end{cases}$$

gives rise to a first order exact approximation and

$$(5.11) \qquad \begin{cases} a = -\frac{1}{12}\xi^{-2}, \ b = \frac{4}{3}\xi^{-2}, \quad c = -\frac{5}{2}\xi^{-2}, \\ \\ a'=\frac{5}{6}\xi^{-2}, \quad b'=-\frac{5}{4}\xi^{-2}, \ c'=-\frac{1}{3}\xi^{-2}, \ d'=\frac{7}{6}\xi^{-2}, \ e'=-\frac{1}{2}\xi^{-2}, \ f'=\frac{1}{12}\xi^{-2}, \end{cases}$$

to a third order exact approximation.

In the first order case it is easily verified that the matrix D has eigenfunctions of the form $\exp(\omega j\xi)$ with eigenvalues

$$(5.12) \qquad \delta_{\omega} = 2(\cos\ \omega\xi-1)\xi^{-2}.$$

Hence, D has negative eigenvalues with a spectral radius

(5.13) $\sigma(D) = 4\xi^{-2}$.

In the third order case the eigenfunctions and eigenvalues are not so easily found. As an estimate of $\sigma(D)$ we shall use the spectral radius of the matrix D approximating the operator $D^{(i)}$ within third order accuracy, that is we neglect the boundary conditions. The eigenvalues δ of this approximation are of the form

(5.14) $\delta_\omega = (-\frac{1}{3}\cos^2\omega\xi + \frac{8}{3}\cos\omega\xi - \frac{7}{3})\xi^{-2}$

with the spectral radius

(5.15) $\sigma(D) = \frac{16}{3}\xi^{-2}$.

Formulae (5.13) and (5.14) lead to the stability conditions

(5.16) $\tau \leq \frac{1}{4}\beta(n)\xi^2, \quad \tau \leq \frac{3}{16}\beta(n)\xi^2$,

respectively.

Clearly, equation (5.9) should be solved by polynomials which exploits the fact that D has negative eigenvalues. Such polynomials are discussed in reference [3]. The order p of the polynomial to be used is determined by the approximation error of the difference scheme. In the cases (5.10) and (5.11) we have respectively errors

(5.17) $O(\tau^{p+1}) + O(\tau\xi^2), \quad O(\tau^{p+1}) + O(\tau\xi^4)$,

so that, by (5.16), p should be chosen 1 and 2. For p = 1 we have chosen the polynomials

(5.18) $T_4(1+z/16), \quad T_{10}(1+z/100)$

with $\beta(4) = 32$ and $\beta(10) = 200$.

For p = 2 we have chosen the polynomials

(5.19) $P_2(z) = 1 + z + \frac{1}{2}z^2$, $\beta(2) = 2$,

(5.20) $P_3(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{16}z^3$, $\beta(3) = 6.27$,

(5.21) $P_4(z) = 1 + z + \frac{1}{2}z^2 + .0786845\ z^2 + .00360845\ z^4$, $\beta(4) = 12$.

We solved problem (5.1) for $0 \leq t \leq .3$. In table 5.1 the relative accuracy ε_{rel} and an estimate for the computational labour required, are listed (the computational labour was measured by the quantity $Kcn/100\Delta x$, where K is the number of integration steps, n the degree of the polynomial and c has the values 1 and 2 in the first and third order case respectively).

Table 5.1 Values of computational labour and relative accuracy obtained in solving problem (5.1)

ε_{rel}	$T_4(1+z/16)$	$T_{10}(1+z/100)$	$P_2(z)$	$P_3(z)$	$P_4(z)$
.10%	130	80	30	15	11
.06%	290	160	40	20	15
.02%	1070	700	67	33	24

These results were obtained by neglecting accuracy conditions (aeta and reta negative), because it was expected that the stability condition is the most stringent one.

5.2 The equation $U_t = (aU+b)\ (U_{rr} + \frac{1}{r}U_r)$

The next initial boundary value problem arose in a physics problem:

$$(5.22)\quad \begin{cases} U_t = (aU+b)(U_{rr} + \frac{1}{r}U_r), & 0 \leq r \leq \infty, \quad t \geq 0, \\ U = 1, & 0 \leq r \leq 1, \quad t = 0, \\ U = 0, & 1 \leq r \leq \infty, \quad t = 0, \\ U_r = 0, & r = 0,\ \infty, \quad t \geq 0. \end{cases}$$

Here, -a and b are given positive constants.

In [6] this problem was solved by replacing the discontinuous initial function by the continuous function

$$(5.23)\qquad U = \begin{cases} 1, & r \leq 1-\Delta r, \\ \frac{1}{2}(1+\cos(\frac{(r-1)+\Delta r}{2\Delta r}\pi)), & 1-\Delta r \leq r \leq 1+\Delta r,\ t = 0, \\ 0, & x \geq 1+\Delta r. \end{cases}$$

Furthermore, the independent variable r was replaced by a variable x which amplifies the region $1-\Delta r \leq r \leq 1+\Delta r$ in such a way that the initial function in this new variable behaves sufficiently smooth. Here, we shall not use such a transformation. Moreover, we replace the right hand boundary condition simply by

$$(5.23)\qquad U = 0, \quad r = r_0, \quad t = 0.$$

Problem (5.22) can be approximated by an initial value problem for the differential equation (compare [6], section 4)

$$(5.24)\qquad \dot{U} = DU,$$

where U is a vector with components $U^{(j)}$ corresponding to the grid points $j\xi$, $j = 0, 1, 2, \ldots, m$; $m = \frac{x_0}{\xi}$ and D is the matrix

$$D = \frac{1}{\xi^2}\begin{bmatrix}
-4d^{(0)}, & 4d^{(0)}, & 0\ , & 0\ , & \cdots\cdots & & & 0 \\
\frac{1}{2}d^{(1)}, & -2d^{(1)}, & \frac{3}{2}d^{(1)}, & 0\ , & \cdots\cdots & & & 0 \\
0\ , & \frac{3}{4}d^{(2)}, & -2d^{(2)}, & \frac{5}{4}d^{(2)}, & 0\ , & \cdots\cdots & & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & & & \vdots \\
0 \cdots & 0\ , & \frac{2j-1}{2j}d^{(j)}, & -2d^{(j)}, & \frac{2j+1}{2j}d^{(j)}, & 0, & \cdots & 0 \\
\vdots & & \ddots & \ddots & \ddots & \ddots & & \vdots \\
0 & \cdots\cdots & & & \cdots & 0\ , & \frac{2m-1}{2m}d^{(m)}, & -2d^{(m)}
\end{bmatrix}.$$

The grid function $d^{(j)}$ is, in fact, the diffusion coefficient

$$d^{(j)} = aU^{(j)} + b.$$

As was pointed out in [6], D has negative eigenvalues with

$$(5.25) \qquad \sigma(D) \leq 4\xi^{-2} \underset{j}{\mathrm{Max}}(2.d^{(j)}).$$

Equation (5.24) is a first order exact approximation of the partial differential equation and, therefore, should be solved by first order exact generating polynomials. Since D has negative eigenvalues we choose the polynomials $T_n(1+z/n^2)$. In our experiments we took:

$$(5.26)\quad \begin{cases} a = -.263, & b = .291, \\ \Delta r = .08, & \xi = .02, \quad r_0 = 2. \end{cases}$$

From (5.25) and the stability condition associated to the generating polynomials $T_n(1+z/n^2)$ it follows that we certainly have stability for

$$\tau = \frac{\beta(n)}{\sigma(D)} = \frac{1}{4}n^2\ \xi^{-2} = 10^{-4}n^2.$$

In table 5.2 some results are listed for n = 10.

Table 5.2 Numerical solution in the neighbourhood of the transition point r = 1

r \ t	0	.10	.20	.30	.40	.50	.60	.70	.80	.90	1.00
.90	1.00	.84	.77	.74	.72	.70	.68	.67	.66	.65	.64
.92	1.00	.79	.74	.71	.69	.67	.66	.65	.64	.63	.62
.94	.95	.74	.70	.68	.66	.65	.64	.63	.62	.61	.60
.96	.84	.69	.66	.65	.64	.63	.62	.61	.60	.59	.59
.98	.68	.64	.63	.62	.61	.60	.59	.59	.58	.58	.57
1.00	.50	.59	.59	.59	.58	.58	.57	.57	.56	.56	.55
1.02	.33	.54	.55	.56	.56	.55	.55	.55	.54	.54	.54
1.04	.18	.49	.52	.53	.53	.53	.53	.53	.53	.52	.52
1.06	.09	.44	.48	.50	.51	.51	.51	.51	.51	.51	.50
1.08	.03	.40	.45	.47	.48	.49	.49	.49	.49	.49	.49
1.10	.00	.36	.42	.44	.46	.47	.47	.47	.47	.47	.47

6. The exponential fitted Taylor method

The second numerical integration method presented in this paper is the procedure exponential fitted taylor. This procedure is based on the three-cluster method described in reference [4], section 4.4. It is appropriate for the integration of those problems of type (2.1) in which the Jacobian has eigenvalues δ which can be reasonably placed into three clusters as illustrated in figure 6.1. Such equations are said to be stiff.

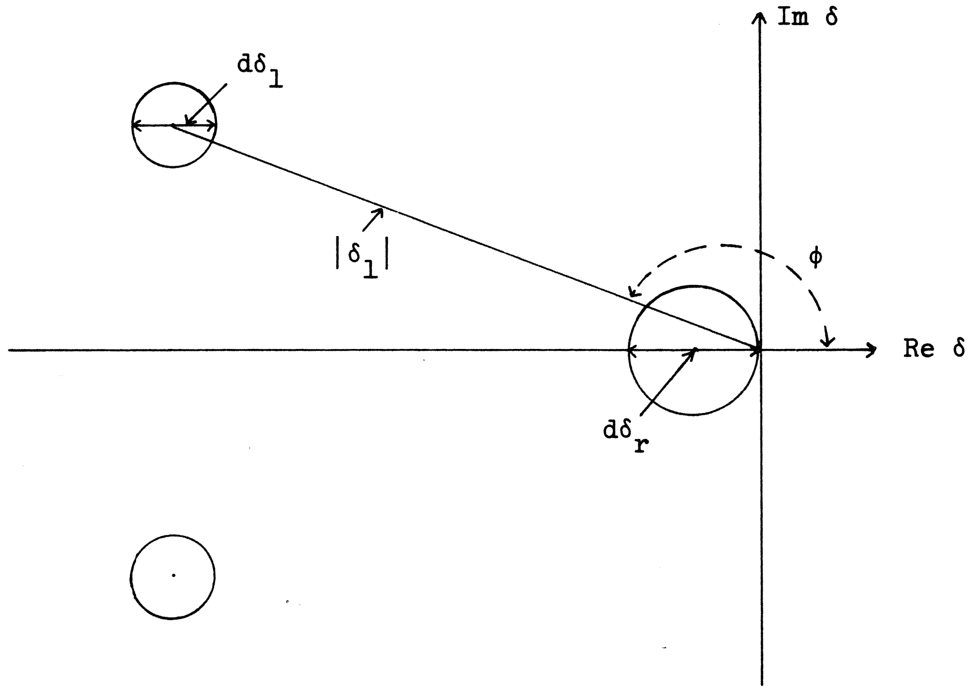

fig. 6.1 Eigenvalues δ situated in three clusters with $d\delta_r << |\delta_l|$, $d\delta_l << |\delta_l|$

In an actual application of the method it is necessary to give explicitly the first three derivatives of the function U(t) in terms of t and U.

In the following subsections the ALGOL 60 version of the three-cluster method is discussed.

6.1 Procedure exponential fitted taylor

The heading of this procedure reads as follows:

```
procedure exponential fitted taylor (t, te, m0, m, u, sigma, phi, diameter,
                                     derivative, i, k, alfa, norm, aeta,
                                     reta, eta, rho, output);
integer m0, m, i, k, norm;
real t, te, sigma, alfa, aeta, reta, eta, rho, phi, diameter;
array u;
procedure derivative, output;
```

The actual parameters corresponding to the formal parameters, as far as not identical to the parameters defined in section 2.1, are:

sigma:	<expression>; approximation of the modulus of the center δ_1 of the left hand cluster;
phi:	<expression>; argument ϕ of the center of a left hand cluster; phi should be given in radians by the user of the procedure;
diameter:	<expression>; diameter $d\delta_1$ of the left hand clusters; diameter should be given by the user;
aeta, reta:	<expression>; desired absolute and relative local accuracy η_a and η_r; aeta and reta should be positive;

Next the procedure body is given:

```
begin integer kl;
    own real ec1,ec2,tau0,tau1,tau2;
    real q,ec0,tau,taui,betan,t2,sigmal,phil;
    real array c,ro[m0:m],beta,betha[1:3];
```

```
procedure coefficient;
begin real b,b1,b2,bb,e,beta2,beta3;
    b:=tau×sigmal; b1:=b×cos(phil); bb:=b×b;
    if abs(b)<_10-3 then
    begin beta2:=.5-bb/24;
          beta3:=1/6+b1/12;
          betha[3]:=.5+b1/3
    end else
    begin e:=exp(b1)/bb; b2:=b×sin(phil);
        beta2:=(-2×b1-4×b1×b1/bb+1)/bb;
        beta3:=(1+2×b1/bb)/bb;
        if abs(b2)<_10-9 then
        begin beta2:=beta2-e×(b-3);
              beta3:=beta3+e×(b-2)/b;
              betha[3]:=1/bb+e×(b-1)
        end else
        begin beta2:=beta2-e×sin(b2-3×phil)/b2×b;
              beta3:=beta3+e×sin(b2-2×phil)/b2;
              betha[3]:=1/bb+e×sin(b2-phil)/b2×b;
        end
    end;
    beta[1]:=betha[1]:=1;
    beta[2]:=beta2; beta[3]:=beta3;
    betha[2]:=1-bb×beta3; b:=abs(b);
    q:=if b<1.5 then 4-2×b/3 else if b<6 then (30-2×b)/9 else 2;
end;
```

```
real procedure normfunction(norm,w);
integer norm; array w;
begin integer j; real s,x;
    s:=0;
    if norm=1 then
    begin for j:=m0 step 1 until m do
        begin x:=abs(w[j]); if x>s then s:=x end
    end else
    s:=sqrt(vecvec(m0,m,0,w,w));
    normfunction:=s;
end;



procedure local error bound;
eta:=aeta+reta × normfunction(norm,u);



procedure local error construction(i); integer i;
begin integer j; real b;
    if i=1 then for j:=m0 step 1 until m do ro[j]:=0;
    if i<4 then
    begin b:=betha[i]×taui;
        for j:=m0 step 1 until m do ro[j]:=ro[j]+b×c[j]
    end;
    if i=4 then
```

```
     begin for j:=m0 step 1 until m do ro[j]:=ro[j]-tau×c[j];
          rho:=normfunction(norm,ro);
          ec0:=ec1;ec1:=ec2;ec2:=rho/tau↑q;
     end
end;



procedure stepsize;
begin real tauacc,taustab,taucr,aa,bb,cc;
     if k=0∧kl≠0 then begin k:=kl;tau:=tau2;goto eos end;
     local error bound;
     aa:=2×abs(sigmal/diameter);bb:=.5×abs(1/sin(phil));
     betan:=aa×(if aa<bb then aa else bb);
     if k=0 then tauacc:=(eta/normfunction(norm,c)) else
     if kl=0 then
     begin tauacc:=(eta/rho)↑(1/q)×tau2;
          if tauacc>10×tau2 then tauacc:=10×tau2 else kl:=2
     end else
     if kl<4 then
     begin tauacc:=(eta/rho)↑(1/q)×tau2; kl:=kl+1 end else
     begin aa:=(tau0×(ec2-ec1)-tau1×(ec1-ec0))/
                    (tau2×tau0-tau1×tau1);
          bb:=(ec2-ec1-aa×(tau2-tau1))/tau1;
          if aa>0 then
          begin cc:=ec2-aa×tau2-bb×t2;tauacc:=0;tau:=alfa×tau2;
```

```
                if ¬zeroin(tauacc,tau,aa×tauacc−eta/tauacc↑q+bb×t
                          +cc,₁₀−3×tau2) then tauacc:=tau2×alfa;
            end else tauacc:=(eta/rho)↑(1/q)×tau2;
            tau:=tau2×(if eta<rho then (eta/rho)↑(1/q) else alfa);
            if tauacc>tau then tauacc:=tau;
            if tauacc<.5×tau2 then tauacc:=.5×tau2;
        end;
        taustab:=abs(betan/sigmal); taucr:=tau2×tau2/tau1;
        tau:=if tauacc>taustab then taustab else tauacc;
        if taustab<₁₀−12×t then goto end of eft;
        if tauacc<₁₀−12×t then tauacc:=₁₀−12×t;
        if k>1∧tau>taucr×(1−₁₀−6)∧tau<taucr×(1+₁₀−6) then
        tau:=if tau<taucr then taucr×(1−₁₀−6) else taucr×(1+₁₀−6);
        tau0:=tau1;tau1:=tau2;tau2:=tau;
    eos: if t+tau>te then tau:=te−t
    end;


    procedure difference scheme;
    begin integer i,j; real b;
        i:=0;taui:=1;sigmal:=sigma;phil:=phi;
        for j:=m0 step 1 until m do c[j]:=u[j];
    next term:  i:=i+1; derivative(i,c);
        if i=1 then
        begin if k≠0 then
```

```
            begin local error construction(4);output end;
            stepsize;coefficient;k:=k+1
        end;
        taui:=taui×tau; b:=beta[i]×taui;
        local error construction(i);
        for j:=m0 step 1 until m do u[j]:=u[j]+b×c[j];
        if i<3 then goto next term;
        t2:=t; t:=t+tau
    end;


    kl:=k; k:=0;
next level:
    difference scheme; if t<te then goto next level;
    output;
end of eft:
end of exponential fitted taylor;
```

6.2 Procedure difference scheme

When procedure difference scheme is completed the array u[j] contains the components of the numerical solution at the next grid point $t_{k+1} = t_k + \tau_k$.

In principle, this procedure is equivalent to procedure difference scheme declared in modified taylor, but the organization is different. The reason is that the construction of the local error is completely different (see section 6.3). For instance, the error associated with the step τ_k not only uses the successive derivatives $c_k^{(i)}$, but also the first derivative at the next point (t_{k+1}, u_{k+1}). Therefore, procedure difference scheme must proceed until the first correction term of the next step, otherwise the step length τ_{k+1}, which is based on the error produced in the step τ_k, cannot be predicted.

6.3 Procedure local error construction

As already observed in section 2.3 it may be inconvenient to measure the local error by the first neglected terms of the local Taylor expansion in those cases where the differential equation has a stiff behaviour. There are two reasons: firstly, the discrepancy may differ considerably from the local error; secondly, the first neglected terms may be a poor approximation to the discrepancy. In order to illustrate this we consider once again the third order stabilized Taylor method applied to problem (3.1) for $t \geq 3$ (cf. section 3). We have

$$(6.1) \qquad \rho_k(\tau_k) \sim -.14\left(\frac{\tau_k}{t_k}\right)^4 + \frac{1}{5}\left(\frac{\tau_k}{t_k}\right)^5 - \frac{1}{6}\left(\frac{\tau_k}{t_k}\right)^6 + \ldots$$

and

$$(6.2) \quad \rho_k'(\tau_k) \sim \rho_k(\tau_k) + \varepsilon_k\left[-.023(\tau_k e^{t_k})^4 + \frac{1}{5!}(\tau_k e^{t_k})^5 - \frac{1}{6!}(\tau_k e^{t_k})^6 + \ldots\right].$$

Let us elaborate these expressions for the maximal step size allowed by stability, i.e.

$$\tau_k \sim 6e^{-t_k}.$$

Then we have

$$(6.1') \qquad \rho_k(\tau_k) \sim -.14\,\frac{1290}{(t_k e^{t_k})^4} > -.14\ 10^{-8} \qquad\qquad \text{if } t_k > 3$$

and

(6.2') $\rho'_k(\tau_k) \sim \rho_k(\tau_k) - .26\varepsilon_k \sim - .26\varepsilon_k$ if $\varepsilon_k > 10^{-8}$.

In actual computation, however, we approximate the discrepancy by its first Taylor term, so that we find

(6.2") $\rho'_k(\tau_k) \sim \rho_k(\tau_k) - 29.67\varepsilon_k \sim - 29.67\varepsilon_k$ if $\varepsilon_k > 10^{-10}$.

Our conclusion is the following: firstly, the discrepancy is a reasonable approximation to the local discretization error provided that the global discretization error is at least less than 10^{-9}; secondly, estimating the discrepancy by its first Taylor term we are led to values which are a factor 100 too large.

This example clearly shows the danger of estimating the local error by the first Taylor terms of the discrepancy.

We will describe a completely different approach for controlling the accuracy of numerical calculations. Our starting point is simple: instead of substituting the local analytical solution into the difference scheme, which gives rise to a discrepancy $\rho'_k(\tau)$, we substitute the numerical solution valid between two points t_k and t_{k+1} into the differential equation. This gives rise to a residual $\zeta_k(\tau)$. Let $U_p(t)$ be the numerical solution between the points $P = (t_k,u_k)$ and $Q = (t_{k+1},u_{k+1})$ (see figure 6.2). Substitution of $U_p(t)$ into equation (2.1) leads to

(6.3) $\zeta_k(\tau) = \dot{U}_p(t) - H(t,U_p(t)) = \dot{U}_p(t_k+\tau) - H(t_k+\tau,U_p(t_k+\tau))$.

We now have the following theorem.

<u>Theorem 6.1</u>

The discrepancy $\rho'_k(\tau_k)$ is given by

(6.4) $\rho'_k(\tau_k) \sim \int_0^{\tau_k} \zeta_k(\tau)d\tau$

provided that the integral

$$I_k(\tau_k) = \int_{t_k}^{t_{k+1}} [H(t,\tilde{U}'_p(t)) - H(t,U_p(t))]dt$$

is negligible with repsect to the right hand side of (6.4). Furthermore, we have

$$I_k(\tau) = O(\tau^{p+2}) \quad \text{as } \tau \to 0,$$

where p is the order of the method.

Proof

It is easily verified that

$$\rho'_k(\tau_k) = \tilde{U}'_p(t_{k+1}) - U_p(t_{k+1}) = \tilde{U}'_p(t_{k+1}) - \tilde{U}'_p(t_k) + \tilde{U}'_p(t_k) - U_p(t_{k+1})$$

$$= \int_{t_k}^{t_{k+1}} \dot{\tilde{U}}'_p(t)dt - \int_{t_k}^{t_{k+1}} \dot{U}_p(t)dt$$

$$= \int_{t_k}^{t_{k+1}} [H(t,\tilde{U}'_p(t)) - H(t,U_p(t))]dt + \int_{t_k}^{t_{k+1}} [H(t,U_p(t)) - \dot{U}_p(t)]dt$$

$$= \int_0^{\tau_k} \zeta_k(\tau)d\tau + I_k(\tau_k).$$

The second statement of the theorem is trivial.

From this theorem we derive for $\tau_k \to 0$ a simple upper bound for the discrepancy, i.e.

$$(6.5) \qquad ||\rho'_k(\tau_k)|| \leq \tau_k||\zeta_k(\tau_k)||.$$

For a given value of τ_k the correctness of (6.5) depends on the value of $||I_k||$. Nevertheless, it seems reasonable to base the accuracy of numerical calculations on the value of

(6.6) $\tau_k||\zeta_k(\tau_k)||.$

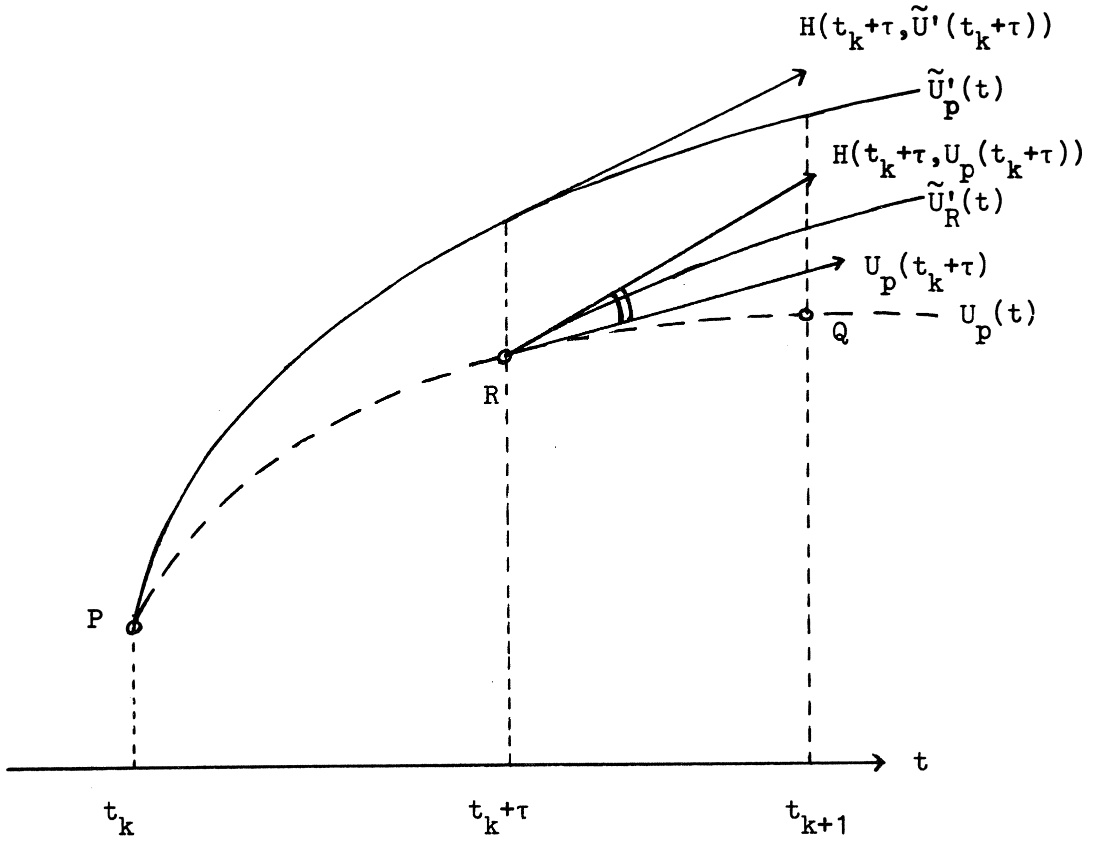

fig. 6.2 Geometrical illustration of the residual function $\zeta_k(\tau)$

Procedure local error construction calculates the value of expression (6.6) and uses it as an estimate of the discrepancy.
From the definition of the three-cluster method it follows that

(6.7) $U_p(t) = U_p(t_k+\tau) = u_k + \tau c_k^{(1)} + \beta_2\tau^2 c_k^{(2)} + \beta_3(\tau)\tau^3 c_k^{(3)},$

where β_2 and β_3 are given functions of τ (see [4], formula (4.14)). This yields

(6.8) $\dot{U}_p(t_{k+1}) = c_k^{(1)} + \beta_2'(\tau_k)\tau_k c_k^{(2)} + \beta_3'(\tau_k)\tau_k^2 c_k^{(3)},$

where

$$(6.9)\quad \begin{cases} \beta_2' = -\frac{1}{b}\left[2\cos\phi + e^{b\cos\phi}\,\frac{\sin(b\sin\phi - 2\phi)}{\sin\phi}\right], \\ \beta_3' = \frac{1}{b^2}\left[1 + e^{b\cos\phi}\,\frac{\sin(b\sin\phi - \phi)}{\sin\phi}\right], \\ b = \tau_k|\delta_1|. \end{cases}$$

Note that $\beta_2' = 1-\beta_3 b^2$.

From (6.6) and (6.7) we finally derive

$$(6.6')\quad \tau_k||\zeta_k(\tau_k)|| = ||\tau_k c_{k+1}^{(1)} - \tau_k c_k^{(1)} - \beta_2'\tau_k^2 c_k^{(2)} - \beta_3'\tau_k^3 c_k^{(3)}||.$$

In the exponential fitted Taylor method we have adopted the right hand side of (6.6') as a measure for the discrepancy. The array betha[i], occurring in procedure local error construction, corresponds to the coefficients β_i'.

6.4 Procedure local error bound

See section 2.4.

6.5 Procedure stepsize

As in section 2.5 the prediction of the step τ_k is based on an extrapolation of the error constants, which are known at the points $t = t_j$, $j < k$, to the next point $t = t_k$. We have used the extrapolation process represented by formula (2.8d), i.e.

$$(6.10)\quad ||\rho'|| = (A\tau+Bt+C)\tau^q.$$

For q the following function was taken (see figure 6.3).

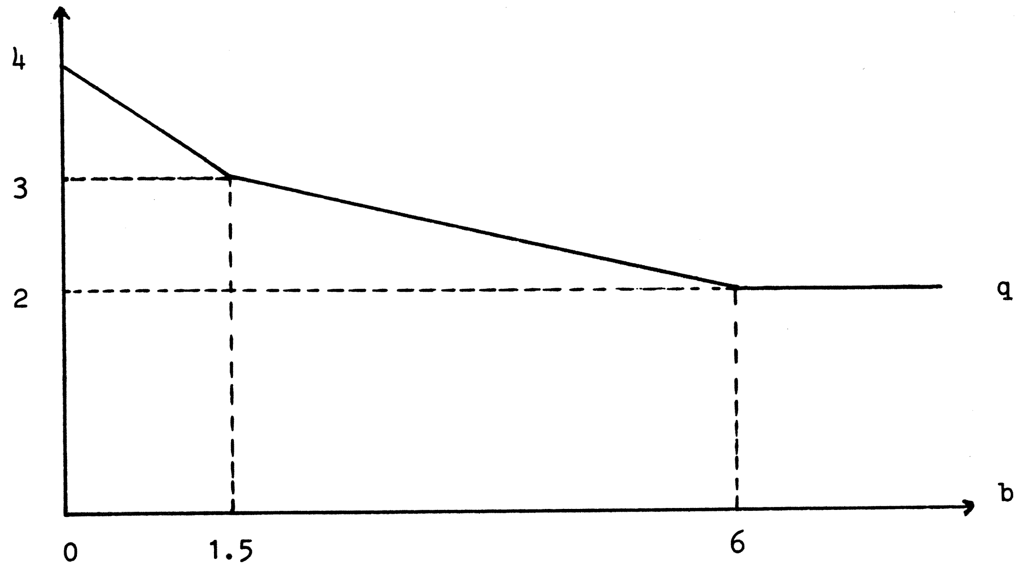

fig. 6.3 Order q of the discrepancy as a function of $b = \tau_k|\delta_1|$

Representation (6.6) gave rise to some difficulties. Firstly, when the step size τ_k equals the critical value

$$\tau_{cr} = \frac{\tau_{k-1}^2}{\tau_{k-2}},$$

then, in the next step, the value of A becomes infinite. Therefore, procedure stepsize excludes the new step τ_k from a small neighbourhood of τ_{cr}. Secondly, the constant A may turn out to be negative; thus when τ increases the error constant decreases. In order to avoid this undesired behaviour of the error constant we have used representation (2.8a) in cases where $A < 0$. Finally, since the extrapolation formula (6.10) is only linear with respect to τ and t the extrapolation process is more sensitive for variations of τ and t than in modified taylor which uses a parabolic formula. In order to guard against rapid variations we have put

$$(6.11) \qquad \alpha = \sqrt[q]{\frac{\eta_k}{||\rho_k'(\tau_k)||}}$$

as soon as $||\rho_k'(\tau_k)|| > \eta_k$. This means that the step length decreases each time it is found that the discrepancy exceeds the tolerance. Of course, the user of procedure exponential fitted taylor may prescribe more severe expressions α than (6.11).

When a prediction of the new step length τ_k is found, this step is checked for stability. The following stability condition is used ([4], formulae (4.8') and (4.18)):

$$(6.12) \qquad \tau_k \leq \frac{\mathrm{Min}\left(\dfrac{|\delta_1|}{d\delta_1|\sin\phi|},\ 4\dfrac{|\delta_1|^2}{(d\delta_1)^2}\right)}{|\delta_1|}.$$

This condition guarantees stability with respect to the components corresponding to eigenvalues in the left hand clusters. The stability properties with respect to the right hand cluster are identical to those of the third order Taylor method as $\tau_k \to 0$ and Euler's method as $\tau_k|\delta_1| \to \infty$ (compare the regions given in [3], fig. 3.1),

6.6 Procedure coefficient

This procedure calculates the values of the coéfficients β_2, β_3, β_2' and β_3' according to [4], formula (4.14) and formula (6.9) of the present paper. Furthermore, the value of q is calculated according to figure 6.3. Since these coefficients depend on the step length, procedure coefficient is called in every integration step.

6.7 Real procedure normfunction (norm,w)

See section 2.7.

7. Numerical solution of stiff equations

We shall employ procedure exponential fitted taylor for the numerical solution of some simple ordinary differential equations with a stiff behaviour.

7.1 The equation $\dot{U} = -e^{t}U + e^{t}\ln t + 1/t$

Let us again consider problem (3.1). We recall that for $t > 3$ the differential equation becomes increasingly stiff and, as we have seen in section 3.5, in this region the modified Taylor method is seriously limited in its step sizes by stability conditions. We have applied the three-cluster method to problem (3.1). Since $\phi = \pi$ the stability condition for this problem becomes (compare (6.7))

$$(7.1)\qquad \tau_k \le 4\,\frac{|\delta_1|}{(d\delta_1)^2}\,.$$

For $|\delta_1|$ and $d\delta_1$ we have chosen the approximations

$$(7.2)\qquad |\delta_1| = e^{t_k},\quad d\delta_1 = 2e^{t_k}(e^{\tau_k}-1) \sim 2\tau_k e^{t_k}.$$

Substitution into (7.1) yields

$$(7.1')\qquad \tau_k \le 4\,\frac{|\delta_1|}{4|\delta_1|^{4/3}} = e^{-1/3t_k}\,.$$

From this it follows that the parameters sigma and diameter respectively are

$$(7.2')\qquad \text{sigma} = e^{+t},\quad \text{diameter} = 2e^{2/3t}.$$

In table 7.1 the results are listed of the following call of procedure exponential fitted taylor (compare the call of modified taylor in section (3.2)):

```
k:= 0;
exponential fitted taylor (t, if k < 200 then t else 8, 0, 0, u, exp(t),
                       3.141592, 2 * exp(2*t/3), derivative, i, k, 1.5, 1,
                       10^-5, 10^-4, eta, rho, output);
```

Table 7.1 Three cluster method applied to problem (3.1) with $\eta_a = 10^{-5}$, $\eta_r = 10^{-4}$

k	t_k	τ_k	$\tau_{stab} = \exp(-\frac{1}{3}t_k)$	$\eta_k/\lvert\rho_k'\rvert$	$\lvert\varepsilon_k\rvert = \lvert\tilde{U}_k - u_k\rvert$	$\lVert\varepsilon\rVert_\infty$
0	.010	$5\ 10^{-6}$	.997	10^{10}	0	$1.3\ 10^{-3}$
10	.034	.004	.989	1.74	$7.4\ 10^{-4}$	
20	.113	.013	.963	1.63	$1.3\ 10^{-3}$	
30	.339	.035	.893	1.61	$1.3\ 10^{-3}$	
40	.888	.076	.744	1.39	$6.1\ 10^{-4}$	
50	1.766	.092	.555	.88	$5.2\ 10^{-5}$	
60	2.619	.078	.418	.92	$3.7\ 10^{-5}$	
70	3.336	.065	.329	.95	$3.1\ 10^{-5}$	
80	3.942	.056	.269	.95	$2.6\ 10^{-5}$	
90	4.466	.049	.226	.96	$2.3\ 10^{-5}$	
100	4.928	.043	.192	.97	$2.0\ 10^{-5}$	
...	...	...	...	...	...	
200	7.915	.022	.071	1.00	$3.5\ 10^{-6}$	

At first sight the superiority of the three-cluster method is not clear from these results. In the first part of the integration process it is even considerably less efficient than the stabilized third order Taylor method generated by polynomial (3.4) (compare table 3.2). This may be explained by the fact that the present method has third order accuracy only as $\tau \to 0$, whereas the Taylor method is "uniformly" of third order. In the second part of the integration the three-cluster method is the more efficient one, as it is not retarded by stability. This means that we can accelerate the integration by relaxing the accuracy condition. Note that in the case of the modified Taylor method a less severe tolerance function will hardly diminish the number of integration steps as the greater part of the steps is governed by stability.

In table 7.2 some results are given for lower values η_a and η_r. As can be seen a large number of integration steps are saved while a reasonable accuracy is retained.

Table 7.2 Three-cluster method applied to problem (3.1)

$\eta_a=\eta_r$	k	t_k	τ_k	$\tau_{stab} = \exp(-\frac{1}{3}t_k)$	$\eta_k/\lvert\rho_k'\rvert$	$\lvert\varepsilon_k\rvert=\lvert\tilde{U}_k-u_k\rvert$	$\lvert\varepsilon\rvert_\infty$
10^{-1}	10	.772	.389	.773	10.3	5.3 10^{-2}	9.7 10^{-2}
	20	4.592	.216	.216	7.7	1.5 10^{-3}	
	30	6.256	.124	.124	17.8	2.4 10^{-4}	
	40	7.312	.087	.087	27.2	7.6 10^{-5}	
	49	8.000		.069		2.3 10^{-5}	
10^{-2}	10	.238	.080	.927	3.2	4.3 10^{-2}	4.4 10^{-2}
	20	3.098	.283	.356	.84	2.8 10^{-3}	
	30	5.336	.169	.169	1.15	6.4 10^{-4}	
	40	6.701	.107	.107	2.2	1.5 10^{-4}	
	50	7.629	.079	.079	3.1	6.1 10^{-5}	
	55	8.000		.069		3.9 10^{-5}	
10^{-3}	10	.075	.021	.975	.82	6.4 10^{-3}	8.6 10^{-3}
	20	.596	.122	.820	2.4	6.3 10^{-3}	
	30	2.644	.167	.414	.93	4.2 10^{-4}	
	40	4.080	.119	.257	.94	2.6 10^{-4}	
	50	5.138	.093	.180	1.00	1.4 10^{-4}	
	60	5.995	.077	.136	1.00	8.3 10^{-5}	
	70	6.718	.066	.107	1.00	5.0 10^{-5}	
	80	7.342	.058	.087	1.00	3.3 10^{-5}	
	90	7.888	.051	.072	1.00	2.2 10^{-5}	
	93	8.000		.069		1.1 10^{-6}	

The efficiency of our integration method can be further improved by exploiting the fact that in all our experiments with a constant tolerance it turned out that the accumulated discretization error ε_k is relatively small for larger values of t_k. This may be explained by recalling the fact that the differential equation becomes increasingly stiff for larger values of t_k so that the discrepancy becomes an increasingly more pessimistic estimate of the local discretization error. This suggests to use a tolerance function which is larger as the equation becomes more stiff. In the present case we expect the discrepancy to behave like $\tau_k^2 e^{2t_k}$ in the stiff region, while the local discretization error is certainly not expected to increase with t_k. This implies that the tolerance function η_k should be such that the step sizes prescribed by accuracy are at least non-decreasing in the stiff region, i.e.

$$\eta_k \sim e^{2t_k}.$$

Let us take $3 \leq t \leq 8$ as the stiff region of the problem. Then we may define

$$(7.3) \qquad \eta_k = (\eta_a + \eta_r ||u_k||) f_k,$$

where $f_k = 1$ for $t_k \leq 3$, $f_k = e^{2t_k - 6}$ for $t_k \geq 3$ and η_a, η_r are constants. With this tolerance function we found the results listed in table 7.3 (see page 71).

We have omitted the case $\eta_a = \eta_r = 10^{-1}$ since the results would be equal to the ones listed in table 7.2. This can be concluded from the fact that in table 7.2 stability already controls the integration in the stiff region.

This section is concluded with a comparison of the computational labour involved when applying the modified and the exponential fitted Taylor method in the stiff region, respectively. Let $3 \leq t \leq 8$ be the stiff region then a modified Taylor method with stability parameter $\beta(n)$ chooses step sizes which do not exceed the value

$$\tau_k = \beta(n) e^{-t_k}.$$

Table 7.3 Three-cluster method with the tolerance function (7.3)

$\eta_a=\eta_r$	k	t_k	τ_k	$\tau_{stab} = \exp(-\frac{1}{3}t_k)$	$\eta_k/\lvert\rho'_k\rvert$	$\lvert\varepsilon_k\rvert=\lvert\tilde{U}_k-u_k\rvert$	$\lvert\lvert\varepsilon\rvert\rvert_\infty$
10^{-2}	10	.238	.080	.924	3.2	4.3 10^{-2}	4.4 10^{-2}
	20	3.098	.310	.356	.76	2.8 10^{-3}	
	30	5.490	.160	.160	1.8 10^{2}	5.4 10^{-4}	
	40	6.798	.104	.104	4.5 10^{3}	1.4 10^{-4}	
	50	7.700	.077	.077	3.8 10^{4}	5.7 10^{-5}	
	55	8.000		.069		1.8 10^{-7}	
10^{-3}	10	.075	.021	.975	.82	6.4 10^{-3}	8.6 10^{-3}
	20	.596	.122	.820	2.4	6.3 10^{-3}	
	30	2.644	.167	.414	.93	4.2 10^{-4}	
	40	4.548	.220	.220	1.7	1.4 10^{-3}	
	50	6.232	.125	.125	1.1 10^{2}	2.5 10^{-4}	
	60	7.294	.088	.088	1.5 10^{3}	8.4 10^{-5}	
	69	8.000		.069		3.7 10^{-5}	
$\eta_a=10^{-5}$ $\eta_r=10^{-4}$	10	.034	.004	.989	1.74	7.4 10^{-4}	1.3 10^{-3}
	20	.113	.013	.963	1.63	1.2 10^{-3}	
	30	.338	.035	.893	1.61	1.2 10^{-3}	
	40	.884	.076	.745	1.39	5.1 10^{-4}	
	50	1.767	.092	.555	.88	5.2 10^{-5}	
	60	2.619	.078	.418	.92	3.7 10^{-5}	
	70	3.365	.080	.325	.90	5.1 10^{-5}	
	80	4.285	.108	.240	.94	2.1 10^{-4}	
	90	5.630	.153	.153	1.73	4.1 10^{-4}	
	100	6.888	.101	.101	3.8 10^{1}	1.3 10^{-4}	
	110	7.767	.075	.075	3.1 10^{2}	5.3 10^{-5}	
	114	8.000		.069		1.7 10^{-6}	

For $t_k \geq 3$ we may approximate this relation by

$$\frac{dt_k}{dk} = \beta(n)e^{-t_k},$$

so that

$$k = \frac{1}{\beta(n)} e^{t_k} + \text{const.}$$

From this relation we deduce that the number of steps required by the modified Taylor method to integrate the interval $3 \leq t \leq 8$ is at least

$$\frac{e^8-e^3}{\beta(n)} = \frac{2981-20}{\beta(n)} = \frac{2961}{\beta(n)} .$$

Hence the number of evaluations of a derivative of the right hand side of equation (3.1) is at least

$$(7.4) \qquad \frac{2961n}{\beta(n)} .$$

This lower bound does not depend on the required accuracy. For the methods discussed in section 3 expression (7.4) assumes the appropriate values 4000, 1940, 970 and 360. The exponential fitted Taylor method, however, requires a number of evaluations which varies from 100 to 150 for the cases listed in table 7.3.

7.2 Two coupled differential equations

In reference [1] the following initial value problem was discussed as an illustration of a stiff differential equation:

$$(7.5) \qquad \dot{U} = DU + F, \quad U(0) = \tilde{U}_0,$$

where

$$D = \begin{pmatrix} -500.5 & 499.5 \\ 499.5 & -500.5 \end{pmatrix}, \quad F = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \quad \tilde{U}_0 = \begin{pmatrix} -.1 \\ .1 \end{pmatrix}.$$

The matrix D has the eigenvalues $\delta_l = -1000$ and $\delta_r = -1$ with eigenvectors $(-1,1)^T$ and $(1,1)^T$, respectively. In the initial phase the solution is for the greater part composed of the eigenvector $(-1,1)^T$ since the initial vector is composed of this vector. However, this component will vanish rapidly, as it corresponds to the eigenvalue $\delta_l = -1000$, and the inhomogeneous term F only introduces the slowly varying component corresponding to $\delta_r = -1$. In fact, the analytical solution of (7.5) is given by

$$(7.6) \qquad U = 2(1-e^{-t})\begin{pmatrix} 1 \\ 1 \end{pmatrix} + \frac{1}{10} e^{-1000t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}.$$

We have done experiments with

$$(7.7) \qquad \eta_a = \eta_r = 1, 10^{-1}, 3 10^{-2}, 10^{-2}, 3 10^{-3}, 10^{-3}, 3 10^{-4}, 10^{-4}, 3 10^{-5}, 10^{-5}.$$

The actual call of exponential fitted taylor reads:

```
k:= 0;
for te:= .1 step .1 until 1 do
exponential fitted taylor (t, te, 0, 1, u, 1000, 3.1416, 0, derivative, i,
                          k, 1.5, 2, aeta, reta, eta, rho, output);
```

In table 7.4 we have listed the values of 3K, K being the total number of integration steps, and

$$||\varepsilon||_\infty = \underset{t_k=j/10, j=0,1,\ldots,10}{\text{Maximum}} ||\tilde{U}_{(t_k)} - u_k||_2 ,$$

where $||\ ||_2$ denoted the maximum norm.

Note that the standard third order Taylor method, which is of comparable order as $\tau \to 0$, requires at least 400 integration steps, thus $3K \geq 1200$.

Table 7.4 Three-cluster method applied to problem (7.5)

$\eta_a = \eta_r$	3K	$\|\|\varepsilon\|\|_\infty$
10^0	36	$3.6\ 10^{-2}$
10^{-1}	39	$3.6\ 10^{-2}$
$3\ 10^{-2}$	39	$3.6\ 10^{-2}$
10^{-2}	51	$3.1\ 10^{-2}$
$3\ 10^{-3}$	81	$1.8\ 10^{-2}$
10^{-3}	117	$1.1\ 10^{-2}$
$3\ 10^{-4}$	189	$5.8\ 10^{-3}$
10^{-4}	288	$3.1\ 10^{-3}$
$3\ 10^{-5}$	468	$1.5\ 10^{-3}$
10^{-5}	711	$6.9\ 10^{-4}$

In order to investigate the consequences of an inaccurate estimate of the left eigenvalue we have done experiments with intentionally wrong chosen $|\delta_1|$ or ϕ. Furthermore, we have done the corresponding experiments in which $d\delta_1$ was chosen in such a way that the exact position of δ_1, i.e. $\delta_1 = -1000$, was just in the left hand cluster determined by $|\delta_1|$, ϕ and $d\delta_1$.

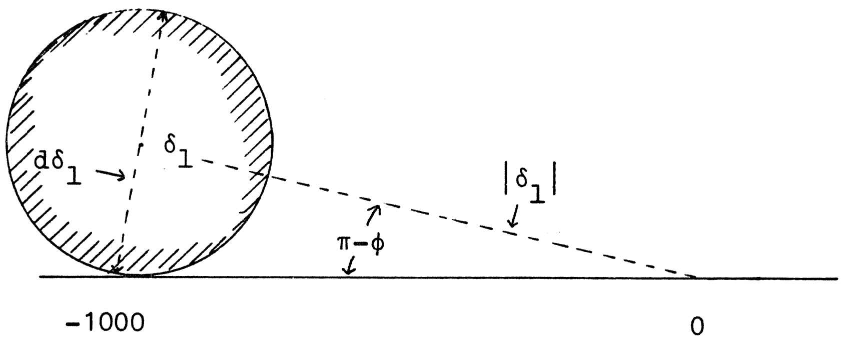

fig. 7.1 Simulation of a cluster with non-zero diameter $d\delta_1$

From figure 7.1 it follows that

(7.8) $$d\delta_1 = 2\sqrt{|\delta_1|^2 + 2\ 10^3|\delta_1|\cos\phi + 10^6}\ .$$

The results are shown in table 7.4 and illustrated in figure 7.2 and 7.3.

Table 7.4 Three-cluster method applied to problem (7.5) with $\eta_a = \eta_r = 10^{-3}$

		$d\delta_1 = 0$		$d\delta_1$ according to (7.8)		
$d\|\delta_1\|/\|\delta_1\|$	$d\phi/\pi$	3K	$\|\|\varepsilon\|\|_\infty$	3K	$\|\|\varepsilon\|\|_\infty$	τ_{stab}
0	0	117	$1.1\ 10^{-2}$	117	$1.1\ 10^{-2}$	∞
	$.05/\pi$	120	$1.1\ 10^{-2}$	120	$1.1\ 10^{-2}$	.2
	$.10/\pi$	126	$1.1\ 10^{-2}$	126	$1.1\ 10^{-2}$	.05
	$.15/\pi$	132	$1.1\ 10^{-2}$	174	$6.3\ 10^{-3}$	.0222
	$.20/\pi$	135	$1.1\ 10^{-2}$	264	$3.3\ 10^{-3}$	.0125
	$.25/\pi$	135	$1.1\ 10^{-2}$	411	$1.7\ 10^{-3}$	.0080
	$.30/\pi$	192	$7.6\ 10^{-3}$	561	$1.0\ 10^{-3}$	.0056
	$.35/\pi$	246	$5.5\ 10^{-3}$	741	$5.9\ 10^{-4}$	.0041
	$.40/\pi$	303	$3.9\ 10^{-3}$	948	$3.6\ 10^{-4}$	.0032
	$.45/\pi$	369	$3.0\ 10^{-3}$	1185	$2.2\ 10^{-4}$	.0026
	$.50/\pi$	336	$1.0\ 10^{-2}$	1452	$1.4\ 10^{-4}$	.0021
-.5	0	462	$8.3\ 10^{-3}$	1536	$4.2\ 10^{-5}$	.0020
-.4		330	$6.8\ 10^{-3}$	828	$2.4\ 10^{-4}$	.0038
-.3		273	$4.9\ 10^{-3}$	414	$1.3\ 10^{-3}$	.0078
-.2		147	$9.1\ 10^{-3}$	177	$5.5\ 10^{-3}$	.0200
-.1		126	$1.1\ 10^{-2}$	126	$1.1\ 10^{-2}$	.0900
0		117	$1.1\ 10^{-2}$	117	$1.1\ 10^{-2}$	∞
.1		126	$1.1\ 10^{-2}$	126	$1.1\ 10^{-2}$	.1100
.2		132	$1.1\ 10^{-2}$	147	$8.7\ 10^{-3}$	.0300
.3		138	$1.1\ 10^{-2}$	237	$4.1\ 10^{-3}$	.0144
.4		171	$8.8\ 10^{-3}$	384	$2.2\ 10^{-3}$	.0088
.5		222	$7.1\ 10^{-3}$	534	$1.3\ 10^{-3}$	.0060

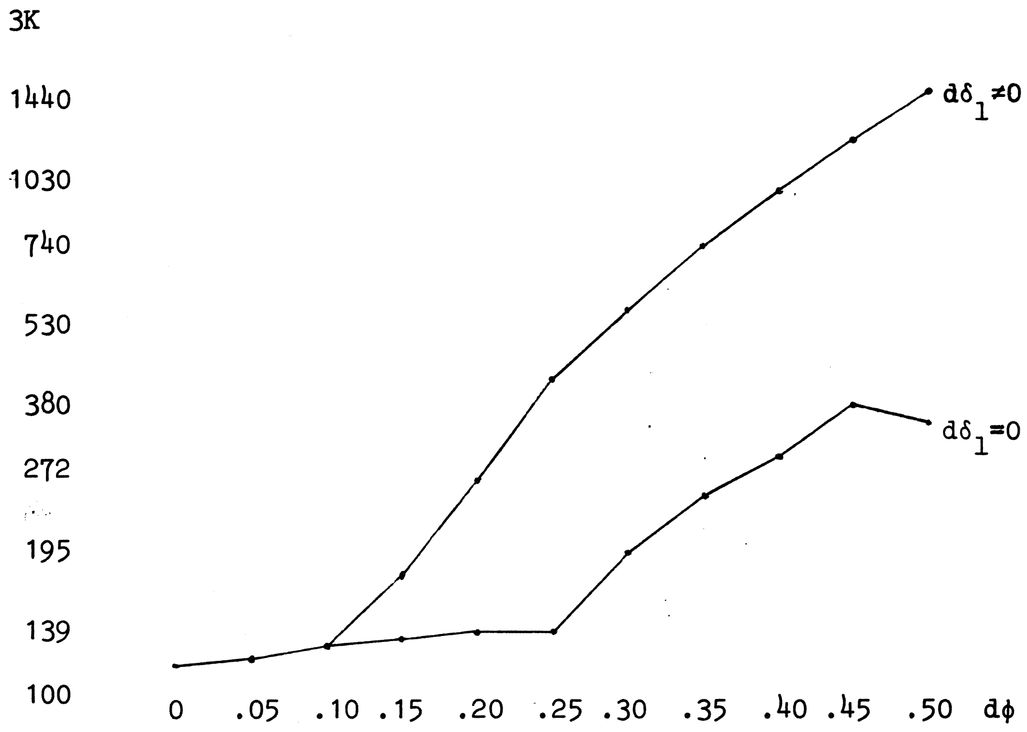

fig. 7.2 The computational labour as a function of $d\phi/\pi$

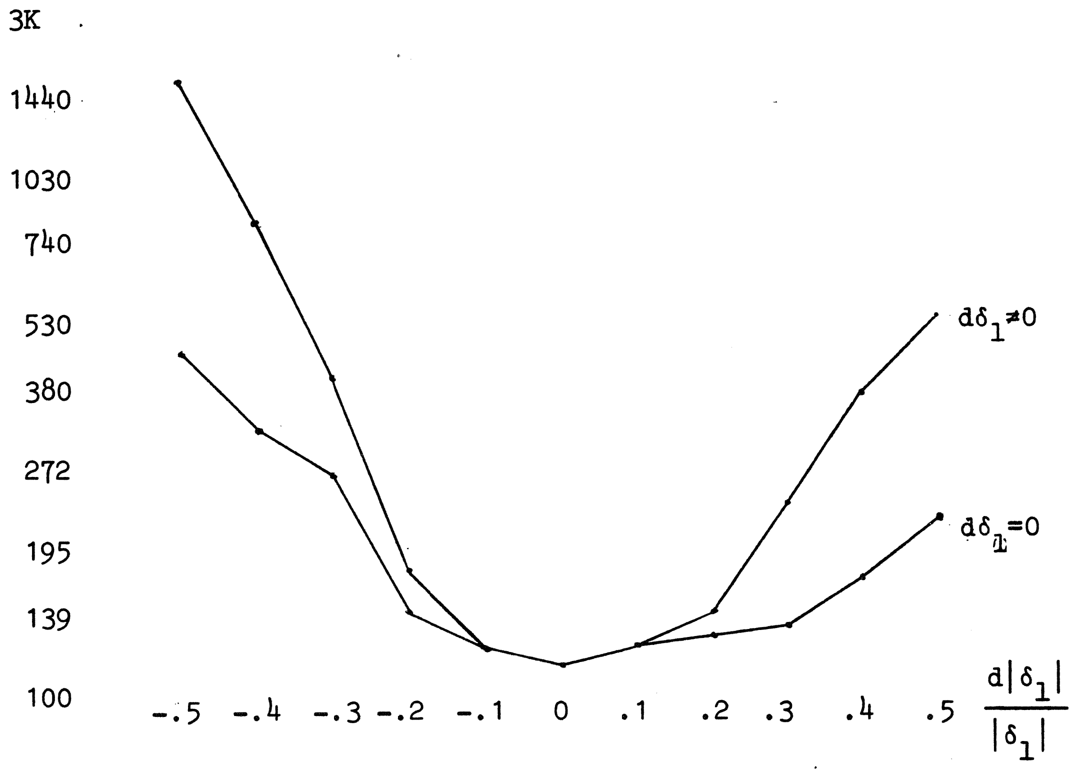

fig. 7.3 The computational labour as a function of $d|\delta_1|/|\delta_1|$

First, we consider the results corresponding to $d\delta_1 = 0$, the case of an inaccurately estimated eigenvalue δ_1. As can be seen from the figures we have an increase of approximately 15% of the computational labour (function evaluations) provided that ϕ and $|\delta_1|$ are accurate within ±8% and -15%, +30%, respectively. For higher inaccuracies the computational labour rapidly increases. Furthermore, we observe that it is better to estimate $|\delta_1|$ too large than too small.
Next, we consider the cases in which a cluster with non-zero diameter is simulated (figure 7.1). From the figures 7.2 and 7.3 it may be concluded that the integration becomes increasingly more laborious when the diameter increases. Only for relatively small clusters the accuracy condition $\eta_a = \eta_r = 10^{-5}$ is more severe than the stability condition.

7.3 A third order differential equation

Consider the initial value problem

$$(7.9)\quad \begin{cases} \dddot{U} + (1-2r\cos\phi)\,\ddot{U} + r(r-2\cos\phi)\,\dot{U} + r^2 U = 0, \\ \\ U(0) = 1,\ \dot{U}(0) = 0,\ \ddot{U}(0) = 0 \end{cases}$$

where r and ϕ are given parameters.
This problem can be written in the equivalent form

$$(7.9')\quad \begin{cases} \dot{\vec{U}} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -r^2 & -r(r-2\cos\phi) & 2r\cos\phi-1 \end{pmatrix} \vec{U}, \\ \\ \vec{U}(0) = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \end{cases}$$

where $\vec{U}$ has the components U, $\dot{U}$ and $\ddot{U}$.

It is easily verified that the Jacobian matrix of (7.9') has the eigenvalues

$-1,\ re^{i\phi},\ re^{-i\phi}.$

Hence, for large values of r and $\pi/2 \leq \phi \leq \pi$ the three-cluster method is an appropriate integration method for problem (7.9').

This example was chosen to illustrate the superiority of the residual formula for the evaluation of the discrepancy over the usual formula based on the first neglected Taylor terms. In order to see this we consider the general solution of equation (7.9'):

$$(7.10)\qquad \vec{\tilde{U}} = c_1 \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix} e^t + c_2 \begin{pmatrix} 1 \\ -re^{i\phi} \\ r^2e^{2i\phi} \end{pmatrix} e^{-rte^{i\phi}} + c_3 \begin{pmatrix} 1 \\ -re^{-i\phi} \\ r^2e^{-2i\phi} \end{pmatrix} e^{-rte^{-i\phi}},$$

where c_1, c_2 and c_3 are integration constants determined by the initial conditions. In general, the constants c_2 and c_3 are $O(r^{-1})$ so that initially the third derivative of the vector $\vec{U}$ is $O(r^4)$. Hence, the discrepancy, approximated by its first neglected Taylor terms, is at least $\tau^3 O(r^4)$. This implies initial steps of order $(\eta/r^4)^{1/3}$. To be more concrete, let

$$(7.11)\qquad r = 1000,\ \phi = \frac{2\pi}{3},\ \eta = 10^{-3}.$$

Then we have to expect initial steps $\tau \sim 10^{-5}$.

When using the residual formula for the discrepancy the three-cluster method needs in the case (7.11), 8 trial steps whereafter the step sizes vary from .025 (at t~0) to .039 (at t~1). In table 7.5 some results are listed obtained for alfa = 1.5 and norm = 2.

It may be interesting to compare these results with the results obtained when the initial values are changed to

$$(7.12)\qquad U(0) = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}.$$

which leads to the analytical solution (7.10) with $c_1 = 1$ and $c_2 = c_3 = 0$. In this case procedure exponential fitted taylor only needs 2 trial steps, since the initial derivatives do not contain the fastly decaying components. In table 7.6 some results are shown.

Table 7.5 Three-cluster method applied to problem (7.9') with $r = 1000$, $\phi = 2\pi/3$, $\eta_a = 10^{-3}$ and $\eta_r = 0$

k	t_k	τ_k	u_k	$\dot{u}_k$	$\ddot{u}_k$	$\varepsilon_k = u_k - \tilde{U}_k$
0	0	10^{-10}	1.000	.000	.000	0
5	.00001	.0001	1.000	$-.6\ 10^{-4}$	-11	$1.1\ 10^{-5}$
10	.0611	.025	.942	-.942	.942	$9.1\ 10^{-4}$
15	.190	.027	.827	-.827	.827	$4.2\ 10^{-4}$
20	.327	.029	.719	-.719	.719	$1.6\ 10^{-3}$
25	.475	.031	.619	-.619	.619	$2.5\ 10^{-3}$
30	.634	.033	.527	-.527	.527	$3.4\ 10^{-3}$
35	.806	.037	.443	-.443	.443	$4.0\ 10^{-3}$
40	.994	.006	.367	-.367	.367	$4.5\ 10^{-3}$
41	1.0000		.363	-.363	-.363	$4.5\ 10^{-3}$

Table 7.6 Three-cluster method applied to equation (7.9') with initial conditions (7.12) and $r = 1000$, $\phi = 2\pi/3$, $\eta_a = 10^{-3}$, $\eta_r = 0$

k	t_k	τ_k	u_k	$\dot{u}_k$	$\ddot{u}_k$	$\varepsilon_k = u_k - \tilde{U}_k$
0	0	$6\ 10^{-5}$	1	-1	1	0
5	.081	.026	.921	-.921	+.921	$8.2\ 10^{-4}$
10	.212	.027	.807	-.807	+.807	$2.1\ 10^{-3}$
15	.351	.029	.701	-.701	.701	$3.0\ 10^{-3}$
20	.500	.031	.603	-.603	.603	$3.9\ 10^{-3}$
25	.660	.034	.512	-.512	.512	$4.6\ 10^{-3}$
30	.836	.037	.428	-.428	.428	$5.1\ 10^{-3}$
35	1.000		.362	-.326	.326	$5.4\ 10^{-3}$

7.4 A stiff equation from biochemistry

In biochemistry the following initial value problem is of interest:

$$(7.12)\quad \begin{cases} \dot{S} = (-1+C)S + .99C, \\ \dot{C} = 10^3(-C+(1-C)S), \\ S(0) = 1,\ C(0) = 0. \end{cases}$$

The solution is required at t = 50.

The Jacobian matrix of (7.12) is given by

$$(7.13)\quad D = \begin{pmatrix} C-1 & .99+S \\ 10^3(1-C) & -10^3(1+S) \end{pmatrix}.$$

According to the theorem of Gerschgerin the eigenvalues are situated in two circles of radius .99 + S and centered at C - 1 and $-10^3(1+S)$, respectively. Thus, initially the eigenvalues are in two circles of radius 1.99 and centered at -1 and -2000, while in the stationary state ($\dot{S}=\dot{C}\sim 0$) the eigenvalues are near -1 and -1000. From this it follows that (7.12) is an example of a stiff equation, so that the three-cluster method should be used. In table 7.7 the results obtained at t = 50 are shown. In the actual call of exponential fitted taylor we have put alfa = 1.5 and norm = 2.

Table 7.7 Three-cluster method applied to problem (7.12)

$\eta_a = \eta_r$	k	t_k	s_k	c_k
10^{-1}	18	50	.764859	.433404
10^{-2}	36	50	.765405	.433561
10^{-3}	82	50	.765671	.433644
10^{-4}	170	50	.765781	.433679

8. Summary of integration formulae using at most four derivatives

This paper is concluded with a survey of possible integration formulae using at most four derivatives.

We distinguish four classes of initial value problems.

Class A: Jacobian D of the given system of differential equations is known to have negative or "almost negative" eigenvalues. Many parabolic equations describing diffusion processes belong to this class.

Class B: Jacobian D is known to have imaginary eigenvalues. To this class belong symmetric hyperbolic differential equations.

Class C: Eigenvalues of D cannot be located.

Class D: Eigenvalues of D can be placed in two or three widely spaced clusters.

Table 8.1 The array data for some modified Taylor methods

class	data[-2]	data[-1]	data[0]	data[1]	data[2]	data[3]	data[4]
A	1	1	2	1			
A	2	1	8	1	1/8		
B,C	2	1	1	1	1		
A	2	2	2	1	1/2		
A	3	1	18	1	4/27	4/729	
A	3	2	6.26	1	1/2	1/16	
B	3	2	2	1	1/2	1/4	
A,C	3	3	2.52	1	1/2	1/6	
A	4	1	32	1	5/32	1/128	1/8192
A	4	2	12	1	1/2	(-9)78684485	(-10)36084541
A	4	3	6	1	1/2	1/6	(-9) 18455702
A,B,C	4	4	$2\sqrt{2}$	1	1/2	1/6	1/24

If the given initial value problem belongs to class A, B or C it is recommended to use modified taylor with an array data as given in table 8.1. For methods using more than four derivatives we refer to the references [3] and [4].

If the problem is of type D the two- or three-cluster method, i.e. exponential fitted taylor, is most convenient.

References

[1] Fowler, M.E., R.M. Warten, A numerical integration technique for ordinary differential equations with widely separated eigenvalues, I.B.M. Journal, (1967).

[2] Directions for use of the milli-system for the EL X8, LR1.1, Mathematisch Centrum, Amsterdam, (1971).

[3] Houwen, P.J. van der, One-step methods for linear initial value problems I. Polynomial methods, TW report 119, Mathematisch Centrum, Amsterdam, (1970).

[4] Houwen, P.J. van der, One-step methods for linear initial value problems II. Applications to stiff equations, TW report 122/70, Mathematisch Centrum, Amsterdam, (1970).

[5] Houwen, P.J. van der, J. Kok, Numerical solution of a minimax problem, TW report 123/71, Mathematisch Centrum, Amsterdam, (1971).

[6] Houwen, P.J. van der, C. de Vreugd, A diffusion problem with a discontinuous initial condition, Report TN 58/70, Mathematisch Centrum, Amsterdam, (1970).

[7] Houwen, P.J. van der, One-step methods for linear initial value problems, ZAMM 51, T58, (1971).