NEANDC (E) 231 «L»
INDC (FR) 61/L

Centre d'Etudes de Bruyères-le-Châtel

OSIRIS : A RUNGE KUTTA SOLVER OF SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS

by

M. COLLIN and A. SCHETT

- Décembre 1983 -

Centre d'Etudes de Bruyères-le-Châtel

OSIRIS : A RUNGE KUTTA SOLVER OF SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS

by

M. COLLIN and A. SCHETT

ABSTRACT: The Code OSIRIS (Order and Step Idently Adjusting Runge-Kutta Integrator of Systems) has been developed on the basis of both explicit as well as implicit Runge-Kutta processes of various orders: 4(5), 7(8), 8(9), 1o for explicit processes and 4 and 6 for implicit processes of the Rosenbrock type. This permits an optimization of the integration procedure by choosing the appropriate type of Runge-Kutta methods(explicit or implicit) and by adjusting dynamically the order of the process as well as the step-size.
The performance of the Code OSIRIS is demonstrated by some representative examples and is compared with the Code GEAR which is applying multistep methods.

OSIRIS : un code de résolution de système d'équations différentielles ordinaires à haute dynamique.

Le code Osiris (Order and Stepsize Idently Adjusting Runge-Kutta Integrator of System) comporte différents algorithmes de Runge-Kutta tant explicites (d'ordres 4(5), 7(8), 8(9), 10) qu'implicites (ordres 4 et 6 de type Rosenbrock). Ceci permet d'optimiser la procédure d'intégration par un ajustement permanent des méthodes utilisées et du pas d'intégration.
Quelques exemples représentatifs permettent de comparer les performances du code OSIRIS à celle du code GEAR utilisant des méthodes à pas liés.

TABLE OF CONTENTS

INTRODUCTION

Considerable progress has been made during the last decades in the field of numerical methods for solving initial value problems of systems of ordinary differential equations. Even recent publications on classical numerical methods as RUNGE-KUTTA and multistep are numerous.

Naturally, several computer codes have been developed on the basis of these methods.

Perhaps, the most performant code in this field is the Code GEAR /HI74/ which is based on multistep methods. This Code, since its first release in 1971, has continually benefited from basic mathematical progress as well as from considerable user experience.

However, we have met problems which could not be solved successfully using the Code GEAR. This has obliged us to look for other codes, particularly those based on RUNGE-KUTTA methods, because they are a natural alternative to multistep methods.

Surprisingly, all those codes which we found in current mathematical program libraries, have major deficiencies:

- Some use RUNGE-KUTTA procedures which are clearly superseded;
- others apply up-to-date, but not optimal RUNGE-KUTTA processes.
- Finally, to our great annoyance, none of these codes is powerful enough to solve problems being modestly to extremely stiff which are the kind of systems with which we deal.

This situation has motivated the present work which is primarely devoted to RUNGE-KUTTA methods.

Approximately speaking, this report is divided into three main parts:

The first is devoted to a concise survey of modern RUNGE-KUTTA processes.

The second is concerned with the choice of those processes which have been used in our ODE-solver code.

The third is centered on applications of our code.

Essentially, the code which we have developed is based on modern explicit as well as implicit RUNGE-KUTTA processes of both low and high order. It permits, therefore, an efficient solution of systems, which are low to extremely stiff, to accuracies lying in the range 10^{-1} to 10^{-7}.

All in all, we have incorporated six RUNGE-KUTTA processes into our code (see TABLE 4.2 in CHAPTER 4); four of them are explicit and two are implicit of the Rosenbrock type.

An economical execution is guaranteed by an automatic choice of the locally most efficient RUNGE-KUTTA process and, of course, of an automatic choice of optimal step-size during the integration process.

Concerning the performance of our RUNGE-KUTTA solver in comparison with the Code GEAR the following can be concluded:

- The RUNGE-KUTTA code has successfully solved all problems which the Code GEAR failed to do.
- The RUNGE-KUTTA code is competitive with the Code GEAR as far as other problems which we treated are concerned.

In ANNEX 1 we present a new algorithm to solve systems of non-linear coupled ordinary differential equations of the type (1.1).

1.0 FORMULATION OF THE PROBLEM.

Let us consider the general initial value problem

$$dY_i(t)/dt = f_i(t, Y_1(t), \ldots, Y_I(t)) \ , \tag{1.1}$$

where $i=1,2,3,\ldots,I$.

A subclass of (1.1) is the system

$$dY_i(t)/dt = \sum_{s=1}^{S} g_{is}(t) \prod_{j=1}^{I} Y_j^{r_{isj}}(t) \ , \tag{1.2}$$

where g_{is} are either constants or depending on t .

r_{isj} are integers.

Eq.(1.2) comprises many applications in natural science in general and in physics in particular.

In what follows we shall mention just some of the applications which are somehow relevant in the context of our study.

Application 1: A quantitative description of the nuclear fusion processes:

D + T $\rightarrow$ He-4 + n ,

D + D $\rightarrow$ He-3 + n ,

D + D $\rightarrow$ T + p ,

D + He-3 $\rightarrow$ He-4 + p ,

is given by the following equations

$$dY(t)/dt = \begin{pmatrix} -1 & -1 & -1 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} b_1 Y_1 Y_2 \\ b_2 Y_1 Y_1 \\ b_3 Y_1 Y_1 \\ b_4 Y_1 Y_3 \end{pmatrix} \ , \tag{1.3}$$

where b_i are interaction parameters; Y_1, Y_2, Y_3 dencte the amount of D, T, He-3, respectively; dY(t)/dt is a one column matrix.

Application 2: Matter which is irradiated by means of a particle beam is changing its composition with time. Such a transformation is described by the following differential equations

$$dY_i(t)/dt = \sum_{j=1}^{I} g_{ij} Y_j(t) \quad , \tag{1.4}$$

where Y_i denotes the quantity of the nucleus i .

If g_{ij} are constants, then either the particle flux is constant or it is absent (decay). The particle flux depends on time otherwise.

Application 3: Let us consider in abstracto the following chemical reactions

$$Y_1 \rightarrow Y_2 + Y_3 \ ,$$
$$Y_1 + Y_3 \rightarrow Y_4 \ ,$$
$$Y_4 \rightarrow Y_3 + Y_5 \ ,$$
$$Y_2 + Y_3 \rightarrow Y_6 \ ,$$

where Y_i denotes the concentration of the chemical species i .

The equations describing these processes quantitatively read

$$dY(t)/dt = \begin{pmatrix} -1 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 \\ 1 & -1 & 1 & -1 \\ 0 & 1 & -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} b_1 Y_1 \\ b_2 Y_1 Y_3 \\ b_3 Y_4 \\ b_4 Y_2 Y_3 \end{pmatrix} \tag{1.5}$$

where $dY(t)/dt$ is a one column matrix.

In many applications one has to deal with stiff non-linear ordinary differential equations.

THE NUMERICAL METHODS WHICH WE SHALL DISCUSS IN THIS WORK ARE QUITE GENERALLY APPROPRIATE TO SOLVE THE INITIAL VALUE PROBLEM (1.1).

However, our main numerical effort is focused on the solution of (1.4). We integrate this system for various g_{ij}.

2.0 SOLUTION OF INITIAL VALUE PROBLEMS FOR SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS.

Eq.(1.1) is manifestly complicated. Analytical solutions exist only in rare cases. Usually, to solve it one has to have recourse to numerical methods.

2.1 ANALYTICAL SOLUTIONS.

If (1.1) is non-linear, the analytical solution is only known for some cases, for example, for EULER's equation of motion in a force-free field and for equations resulting from the YANG-MILL theory /RA82/.

If (1.1) reduces to a linear system as (1.3), then two cases have to be distinguished:

Case 1: g_{ij} are constants. Then, the analytical solution of (1.3) is known. It reads in matrix form

$$Y(t) = P^T E C \ , \tag{2.1}$$

where P^T is the transpose of the eigenvector-matrix P,

$$E = \begin{pmatrix} e^{\lambda_1 t} & 0 & \cdots & 0 \\ 0 & e^{\lambda_2 t} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & e^{\lambda_I t} \end{pmatrix} \ ,$$

λ_i $(i=1,2,3,\ldots,I)$ are the eigenvalues of the matrix g_{ij}, and C is a one column matrix containing arbitrary constants which can be determined with the help of (2.1) using the initial values $Y(t=t_0)$. Eq.(2.1) becomes then

$$Y(t) = P^T E(P^T E(t=t_0))^{-1} Y(t=t_0) \ . \tag{2.2}$$

Observe that the solution of (2.2) can be obtained without computing explicitely the inverse of a matrix. It can in fact be achieved with the aid of a GAUSSIAN elimination process together with a complete pivoting followed by a back substitution. One should bear in mind that the solution via direct matrix inversion REQUIRES ABOUT THREE TIMES MORE computational effort than via GAUSSIAN elimination.

An equivalent representation of the analytical solution (2.1) can be obtained by means of LAPLACE-transformation. This solution method offers for not too large systems (1.3) several advantages because:

- No eigenvalue has to be computed.
- No eigenvector has to be computed.
- Only the solution of one linear algebraic system is necessary.

Here, we will only sketch this method to exhibit its underlying principles.

The transformation $\mathcal{L}$ of (1.3) into the image space yields

$$\mathcal{L}(dY_i(t)/dt - g_{ij}Y_j(t)) = sZ_i - Y_i(t=t_o) - g_{ij}Z_j = 0 \quad (2.3)$$

The determination of the unknowns Z_i by solving (2.3) and the transformation of Z_i back to the original space, i.e.,

$$\mathcal{L}^{-1} Z_i = Y_i(t) \quad , \qquad (2.4)$$

yield directly the solution functions $Y_i(t)$.

Finally, we wish to emphasize that an analytical solution, despite its mathematical beauty is not necessarely THE MOST

APPROPRIATE METHOD FOR PRACTICAL COMPUTATION. This is especially true, if the input data are only known within a certain accuracy and, therefore, the solution do not need to be computed exactly either. Integration using numerical methods might in such a case be more economical than the use of the analytical representation of the functions, because the former permits a flexible adaptation to the accuracy requirements and avoids thus wasteful computational effort.

Case 2: g_{ij} are not constants. Then, for some specific functions $g_{ij}(t)$, analytical solutions of (1.3) are known. In general, however, the solution of (1.3) and of course of (1.1) can only be obtained using numerical methods. To solve such a system several numerical methods are at our disposal. ALL OF THEM ARE ESSENTIALLY EXTRAPOLATORY IN A SENSE THAT THEY START FROM A POINT AND DEVELOP THE SOLUTION FOR A NEIGHBOURING POINT.

We are now going to enumerate the common numerical methods and to outline their essential features.

2.2 NUMERICAL METHODS.

2.2.1 Categories and characteristics.

For obvious reasons, a great deal of attention has in literature been given to this subject and comprehensive books have been published on it /GE71/, /HA76/, /LA73/.

It is of course not our aim to repeat here details on these methods, but to exhibit their fundamental features and to select among the various methods those which seem to be

most appropriate and performant to sove problems with which we have to deal. Five categories of numerical methods are of particular interest: the Taylor series method, the RUNGE-KUTTA methods, the multistep methods, the hybrid method and the OBRECHKOFF method.

- *The Taylor series method.*

This is conceptually the most attractive numerical method. Essentially, one uses there all information of a function (value plus derivatives) at a point to determine the function at another point. The LIE-series method /GR6o/ can be regarded as a more elegant form of it. The Taylor series method permits in principle an automatic choice of both the order of approximation as well as the step size. It has the obvious demerit of requiring derivatives of functions.

- The RUNGE-KUTTA methods.

These methods can be considered as approximations to the Taylor series method. However, it uses only the knowledge of the value of the function at a point (none or only one derivative) to determine another function value in its vicinity. This simplification is gained at the price of the determination of the socalled RUNGE-KUTTA schemes for each order of approximation separately. That involves tedious and laborious calculations, especially for RUNGE-KUTTA processes of high order.

- The multistep methods.

Essentially, these methods use the knowledge of the functions at several points (multistep) to determine the value of the function at a further point. Also for these methods one has

to determine coefficients prior to computation.

- The hybrid method.

This method is a combination of the RUNGE-KUTTA and the multistep formalism.

- The OBRECHKOFF method.

This method is a combination of the Taylor series and the multistep formalism.

Each of these methods has its merits and drawbacks. They are complementary. For most purposes one of the methods mentioned above should prove sufficient. As for our applications, we need first of all methods which do not require derivatives of functions involved. Therefore, only two basically different methods remain for our aim: The RUNGE-KUTTA and the multistep methods.

2.2.2 The multistep methods.

These are main numerical methods. They have seen a great deal of attention in literature. A code based on these methods is available. For these reasons we will confine ourselves only to some practical remarks on them.
We have already mentioned above that the essential feature of these methods is the use of several function values to determine a function value at another point. Several formalisms were proposed for that process. Well known are the ADAM-BASHFORTH-MOULTON- and GEAR's methods. Both have been incorporated in the program package called GEAR /HI74/.
This Code has a relatively long history. The first version of it was published 1971 under the name DIFSUB. Since then this Code has benefited continually from the basic mathe-

matical progress and the considerable user experience.
The present version (4-th version) of the Code GEAR applies the ADAM-BASHFORTH-MOULTON method up to the order 12 and the GEAR's method up to the order 5. The Code is self-starting and changes automatically step-size and order of method. It is claimed to be particularly well suited to solve stiff problems.
The Code GEAR has since its release been widely used as reference code to test newly developed methods relative to its performance.
PERHAPS, THIS CODE CAN AT PRESENT BE REGARDED AS THE MOST PERFORMANT ODE-SOLVER.
However, as we shall see later, we have treated problems which could not successfully be solved by means of the Code GEAR. More precisely, the Code could either not start or yielded erroneous results.

2.2.3 The RUNGE-KUTTA methods.

We will treat these methods in more detail, because we have implemented a RUNGE-KUTTA solver of systems of ordinary differential equations. The following reasons have prompted the implementation of such a code :

- The Code GEAR could not successfully solve some problems with which we had to deal.
- RUNGE-KUTTA methods are an attractive alternative to multistep methods, because they do not require derivatives of functions involved either.
- Comparison studies of the performance of numerical methods showed that the RUNGE-KUTTA methods are competitive in general and even superior in certain cases /EN75/, /EN76/,

/HU72/, /SH76/, /GO81/, /KA81/ and /GO82/.

- One possibility of controlling the total accumulated error is the comparison of the solution functions obtained with the aid of two different codes each of them based on a different mathematical algorithm , for example, based on RUNGE-KUTTA- or on multistep methods.

2.2.3.1 Basic notions of RUNGE-KUTTA methods.

2.2.3.1.1 The algorithm of these methods.

Let us consider the initial value problem

$$dY_i(x)/dx = f_i(x,Y_1,Y_2,\ldots,Y_n) \quad , \tag{2.5}$$

where $i=1,2,3,\ldots,n$.

The formal RUNGE-KUTTA approximation of the solution functions reads in vector form

$$Y(x+h) = Y(x)+h.\sum_{j=o}^{s} b_j g_j$$

and the next higher approximation can be written

$$\overline{Y}(x+h) = Y(x)+h.\sum_{j=o}^{\overline{s}} \overline{b}_j \overline{g}_j \quad , \tag{2.6}$$

where

h is the increment or step-size,

s is the stage(number of function evaluations) of the method,

$$g_o = f(x_o,\ Y(x=x_o)) \ ,$$

$$g_i = f(x_o+c_i h, Y(x=x_o)+h \sum_{j=o}^{s} a_{ij} g_j) \quad , \tag{2.7}$$

The determination of these coefficients is in fact the central problem of a RUNGE-KUTTA process of a given order. They are derived in two main steps:

- establishment of the algebraic equations for them,
- solution of these equations with a view to an optimal numerical performance of the RUNGE-KUTTA process.

2.2.3.1.2 Definitions.

If $a_{ij}=o$ for $i \leq j$, then the method is said to be explicit.
If $a_{ij}=o$ for $i<j$, then the method is said to be semi-implicit.
The method is said to be implicit otherwise.

2.2.3.1.3 Applicability of explicit and implicit methods.

Explicit methods are much simpler than implicit methods, because in this case the g_i in (2.7) can be computed in a straightforward manner, i.e. , without solving an algebraic system. HOWEVER, SUCH METHODS HAVE TOO LIMITED STABILITY REGIONS (see Chapter 3). The integration of socalled stiff systems of ordinary differential equations with the help of such methods is thus risky, expensive or even impossible, because the step-size is limited to a much too small value. Whereas, THE IMPLICIT METHODS PERMIT A LARGE STEP-SIZE EVEN FOR STIFF PROBLEMS. However, when applying implicit methods, the g_i in (2.7) must be determined via the solution of algebraic equations, which can be non-linear. That requires not only more computational effort in comparison with explicit methods, but introduces also new error sources in the computational procedure.

2.2.3.2 Brief history of RUNGE-KUTTA methods.

2.2.3.2.1 Generalities.

These methods were invented by the mathematicians C.D.T.RUNGE (1856-1927)/RU95/ and M.W.KUTTA(1867-1944)/KU01/. The basic theory of the remainder of the approximative solution, resulting from these methods, has been developed by L.BIEBERBACH/BI51/. The formal theory of RUNGE-KUTTA processes of any order was published by J.C.BUTCHER /BU63/,/BU64a/,/BU64/.

2.2.3.2.2 Explicit methods.

A central problem in numerical analysis is the error estimation. Concerning RUNGE-KUTTA methods, it was R.H.MERSON/ME57/, who made a pioneering work in this field with a view to practical calculations. Although his error estimation was not optimal and led generally to wasteful computation, it provided a fresh impetus to investigate this item. It was R.ENGLAND /EN67/, /EN69/, who proposed a more realistic error estimation along with optimal explicit RUNGE-KUTTA processes. Both are naturally related. Almost at the same time, E.FEHLBERG /FE68/, /FE69/ and /FE70/ succeeded in deriving still more optimal RUNGE-KUTTA schemata together with economical error estimations. These error estimations fail, however, for problems of the form $Y'(x)=f(x)$ and a RUNGE-KUTTA process of an order greater than 4 /EN76/. It was J.H.VERNER/VE78/ who developed explicit RUNGE-KUTTA schemata along with error estimations which are claimed to work for any system of ordinary differential equations to be integrated. Recently, J.R.CASH /CA83/ has published a RK-algorithm permitting an automatic change of step-size and order of method.

2.2.3.2.3 Implicit methods.

These methods evoked interest at a later stage. One reason for that is that implicit methods are more cumbersome algorithms. Furthermore, one has realized rather late that for the solution of certain systems of ordinary differential equations, more precisely, of the so-called stiff systems, methods having large stability regions are required. The integration of stiff systems over a longer path can only be carried out by means of implicit methods, because only they permit an appropriate integration step-size.

The general theory of implicit RUNGE-KUTTA methods of any order has been developed by J.C.BUTCHER/BU64/. An important breakthrough was made by B.L.EHLE/EH69/. HE SHOWED THAT ONE CAN OBTAIN A-STABLE IMPLICIT RUNGE-KUTTA FORMULAE OF ARBITRARY ORDER.

Note that multistep methods are only A-stable for orders not larger than 2 /DA63/.

In principle, the implicit RUNGE-KUTTA methods permit thus integration of any system of ordinary differential equations. Unfortunately, to integrate a system of m differential equations, implicit RUNGE-KUTTA methods with a full matrix require the solution of m.q (q-stages) simultaneous implicit (in general non-linear) equations at each integration step. This problem was simplified by S.P.NORSETT/NO74/ in that he reduced the full matrix a_{ij} to a triangular matrix, $a_{ij}=o$ for $i<j$. M.CROUZEIX/CR75/ and R.ALEXANDER/AL77/ continued investigations in this direction. Another important simplification has been proposed by H.H.ROSENBROCK/RO63/.

HIS METHOD CAN BE REGARDED AS A LINEARISATION OF SEMI-IMPLICIT RUNGE_KUTTA METHODS. More precisely, whenever semi-implicit RUNGE-KUTTA methods would require the solution of q non-linear equations, the Rosenbrock-type method calls for the solution of q linear equations. This renders implicit methods more economical and numerically more reliable. C.F.HAINES/HA69/ undertook further researches concerning this method. Later B.A.GOTTWALD and G.WANNER/GO81/ and then P.KAPS together with G.WANNER /KA81/ have published RUNGE-KUTTA processes of the Rosenbrock type of orders 4 and 5, 6, respectively. Recently, J.R.CASH/CA83/ and G.J.COOPER together with A.SAYFY/CO83/ have published additional implicit RUNGE-KUTTA processes.

2.2.3.3 Status of coefficients of RUNGE-KUTTA methods.

In the subsequent TABLE 2.1 we give a selection of RUNGE-KUTTA processes.

TABLE 2.1: A selection of RUNGE-KUTTA methods for which coefficients are available.

Explicit		Implicit	
Order	References	Order	References
1-9	/FE68/,/FE69/,/FE7o/	3	/RO63/,/HA69/
5-9	/VE78/	4	/GO81/
1o	/CU75/,/HA78/	5-6	/KA81/

Observe that there exist additional schemata of RUNGE-KUTTA coefficients, for example, those by ENGLAND /EN67/,/EN69/,

by SHANK/SH66/, by SARAFYAN /SA72/ and by others. However, it seems that they are not so efficient as those given in TABLE 2.1.

From this Table one can also see that the highest order of explicit RUNGE-KUTTA methods for which coefficients are available in literature is equal to 1o, and that the highest order of implicit methods for which coefficients are published is equal to 6.

IT SHOULD BE EMPHASIZED THAT COEFFICIENTS OF STILL HIGHER ORDER RUNGE-KUTTA PROCESSES MIGHT NOT BRING ANY SUBSTANTIAL GAIN ANYMORE.

Besides the question of the usefulness of still higher order RUNGE-KUTTA methods for practical computation, the effort required for the determination of the corresponding coefficients is quite considerable. This is well illustrated in TABLE 2.2, where we list the number of non-linear algebraic equations which must be solved for the computation of the coefficients of an explicit RUNGE-KUTTA process of order p.

TABLE 2.2: The number of non-linear algebraic equations to be solved for the determination of the coefficients of an explicit RUNGE-KUTTA process of order p/HA78/.

RK-order p	1	2	3	4	5	6	7	8	9	1o	11
Number of non-linear equations to be solved.	1	2	4	8	17	37	85	2oo	486	12o5	3o47

Moreover, the solution of these equations is not unique and a judicious choice with a view to an economical RUNGE-KUTTA process requires skill and experience.

A measure of such an optimal choice is the number of function evaluations (or stages) for a process of a given order. In TABLE 2.3 we give theoretical lower and upper bounds together with the actual values of the smallest number of function evaluations which are needed for an explicit RUNGE-KUTTA method of a given order. Formulae for the estimation of the bounds of the stages are published in /CO72/ and /BU75/.

TABLE 2.3: Bounds of the minimum number of stages s_{min} of explicit RUNGE-KUTTA processes and the corresponding actual values.

RK-order / s_{min}	1	2	3	4	5	6	7	8	9	1o	11	12
Lower	1	2	3	4	6	7	9	1o	11	12	13	14
Actual	3[1)]	4[1)]	5[1)]	6[1)]	8[2)]	1o[2)]	13[2)]	17[2)]		18[4)]		
					8[3)]	1o[3)]	13[3)]	16[3)]		17[5)]		
upper								11	16	22	29	37

1) /FE68/, 2) /FE69/, 3) /VE78/, 4) /CU75/, 5) /HA78/.

2.2.3.4 Tables of coefficients of RUNGE-KUTTA methods.

In what follows we give for convenience a selection of tables of RUNGE-KUTTA processes. We list only coefficients which deserve particular attention. The notation is that used in (2.6) and (2.7).

2.2.3.4.1 Coefficients of explicit methods.

TABLE 2.4: RK-coefficients of order 1(2) by FEHLBERG/FE7o/.

i \ j	c_i	a_{ij}		b_i	$\bar{b}_i$
		0	1		
0	0			$\frac{1}{256}$	$\frac{1}{512}$
1	$\frac{1}{2}$	$\frac{1}{2}$		$\frac{255}{256}$	$\frac{255}{256}$
2	1	$\frac{1}{256}$	$\frac{255}{256}$		$\frac{1}{512}$

$$TE = \frac{1}{512}(g_o - g_1)\,h,$$

where TE is the truncation error.

TABLE 2.5: RK-coefficients of order 2(3) by FEHLBERG/FE7o/.

i \ j	c_i	a_{ij}			b_i	$\bar{b}_i$
		0	1	2		
0	0	0			$\frac{214}{891}$	$\frac{533}{2\,106}$
1	$\frac{1}{4}$	$\frac{1}{4}$			$\frac{1}{33}$	0
2	$\frac{27}{40}$	$-\frac{189}{800}$	$\frac{729}{800}$		$\frac{650}{891}$	$\frac{800}{1\,053}$
3	1	$\frac{214}{891}$	$\frac{1}{33}$	$\frac{650}{891}$		$-\frac{1}{78}$

$$TE = \left(-\frac{23}{1\,782}g_o + \frac{1}{33}g_1 - \frac{350}{11\,583}g_2 + \frac{1}{78}g_3\right)h,$$

where TE is the truncation error.

TABLE 2.6: RK-coefficients of order 3(4) by FEHLBERG/FE7o/.

i \ j	c_i	a_{ij}				b_i	$\bar{b}_i$
		0	1	2	3		
0	0	0				$\frac{79}{490}$	$\frac{229}{1\,470}$
1	$\frac{2}{7}$	$\frac{2}{7}$				0	0
2	$\frac{7}{15}$	$\frac{77}{900}$	$\frac{343}{900}$			$\frac{2\,175}{3\,626}$	$\frac{1\,125}{1\,813}$
3	$\frac{35}{38}$	$\frac{805}{1\,444}$	$-\frac{77\,175}{54\,872}$	$\frac{97\,125}{54\,872}$		$\frac{2\,166}{9\,065}$	$\frac{13\,718}{81\,585}$
4	1	$\frac{79}{490}$	0	$\frac{2\,175}{3\,626}$	$\frac{2\,166}{9\,065}$		$\frac{1}{18}$

$$TE = \left(\frac{4}{735}g_0 - \frac{75}{3\,626}g_2 + \frac{5\,776}{81\,585}g_3 - \frac{1}{18}g_4\right)h.$$

where TE is the truncation error.

TABLE 2.7: RK-coefficients of order 4(5) by FEHLBERG/FE7o/.

i \ j	c_i	a_{ij}					b_i	$\bar{b}_i$
		0	1	2	3	4		
0	0	0					$\frac{25}{216}$	$\frac{16}{135}$
1	$\frac{1}{4}$	$\frac{1}{4}$					0	0
2	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				$\frac{1\,408}{2\,565}$	$\frac{6\,656}{12\,825}$
3	$\frac{12}{13}$	$\frac{1\,932}{2\,197}$	$-\frac{7\,200}{2\,197}$	$\frac{7\,296}{2\,197}$			$\frac{2\,197}{4\,104}$	$\frac{28\,561}{56\,430}$
4	1	$\frac{439}{216}$	-8	$\frac{3\,680}{513}$	$-\frac{845}{4\,104}$		$-\frac{1}{5}$	$-\frac{9}{50}$
5	$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3\,544}{2\,565}$	$\frac{1\,859}{4\,104}$	$-\frac{11}{40}$		$\frac{2}{55}$

$$TE = \left(-\frac{1}{360}g_0 + \frac{128}{4\,275}g_2 + \frac{2\,197}{75\,240}g_3 - \frac{1}{50}g_4 - \frac{2}{55}g_5\right)h,$$

where TE is the truncation error.

TABLE 2.8: RK-coefficients of order 5(6) by FEHLBERG/FE68/ and /FE69/.

$i \backslash j$	c_i	a_{ij}							b_i	$\bar{b}_i$
		0	1	2	3	4	5	6		
0	0	0							$\frac{31}{384}$	$\frac{7}{1408}$
1	$\frac{1}{6}$	$\frac{1}{6}$							0	
2	$\frac{4}{15}$	$\frac{4}{75}$	$\frac{16}{75}$						$\frac{1125}{2816}$	
3	$\frac{2}{3}$	$\frac{5}{6}$	$-\frac{8}{3}$	$\frac{5}{2}$					$\frac{9}{32}$	
4	$\frac{4}{5}$	$-\frac{3}{5}$	$\frac{144}{25}$	-4	$\frac{16}{25}$				$\frac{125}{768}$	
5	1	$\frac{361}{320}$	$-\frac{18}{5}$	$\frac{407}{128}$	$-\frac{11}{80}$	$\frac{55}{128}$			$\frac{5}{66}$	0
6	0	$-\frac{11}{640}$	0	$\frac{11}{256}$	$-\frac{11}{160}$	$\frac{11}{256}$	0			$\frac{5}{66}$
7	1	$\frac{93}{640}$	$-\frac{18}{5}$	$\frac{803}{256}$	$-\frac{11}{160}$	$\frac{99}{256}$	0	1		$\frac{5}{66}$

The truncation TE reads

$$TE = \frac{5}{66}(g_o+g_5-g_6-g_7)h \ .$$

TABLE 2.9: RK-coefficients of order 5(6) by VERNER /VE78/.

i \ j	c_i	a_{ij} 1	2	3	4	5	6	7	8
1	0								
2	$\frac{1}{18}$	$\frac{1}{18}$							
3	$\frac{1}{6}$	$\frac{-1}{12}$	$\frac{1}{4}$						
4	$\frac{2}{9}$	$\frac{-2}{81}$	$\frac{4}{27}$	$\frac{8}{81}$					
5	$\frac{2}{3}$	$\frac{40}{33}$	$\frac{-4}{11}$	$\frac{-56}{11}$	$\frac{54}{11}$				
6	1	$\frac{-369}{73}$	$\frac{72}{73}$	$\frac{5380}{219}$	$\frac{-12285}{584}$	$\frac{2695}{1752}$			
7	$\frac{8}{9}$	$\frac{-8716}{891}$	$\frac{656}{297}$	$\frac{39520}{891}$	$\frac{-416}{11}$	$\frac{52}{27}$	0		
8	1	$\frac{3015}{256}$	$\frac{-9}{4}$	$\frac{-4219}{78}$	$\frac{5985}{128}$	$\frac{-539}{384}$	0	$\frac{693}{3328}$	
$\bar{b}_i$		$\frac{1}{80}$	0	$\frac{4}{25}$	$\frac{243}{1120}$	$\frac{77}{160}$	$\frac{73}{700}$		
b_i		$\frac{57}{640}$	0	$\frac{-16}{65}$	$\frac{1377}{2240}$	$\frac{121}{320}$	0	$\frac{891}{8320}$	$\frac{2}{35}$

with the truncation error

$$TE = \left(\frac{33}{640}g_1 - \frac{132}{325}g_3 + \frac{891}{2240}g_4 - \frac{33}{320}g_5 - \frac{73}{700}g_6 + \frac{891}{8320}g_7 + \frac{2}{35}g_8\right)h \ .$$

TABLE 2.1o: RK-coefficients of order 6(7) by VERNER /VE78/.

i \ j	c_i	a_{ij} 1	2	3	4	5	6	7	8	9	10
1	0										
2	$\frac{1}{12}$	$\frac{1}{12}$									
3	$\frac{1}{6}$	0	$\frac{1}{6}$								
4	$\frac{1}{4}$	$\frac{1}{16}$	0	$\frac{3}{16}$							
5	$\frac{3}{4}$	$\frac{21}{16}$	0	$\frac{-81}{16}$	$\frac{9}{2}$						
6	$\frac{16}{17}$	$\frac{13446888}{250563}$	0	$\frac{-1709184}{83521}$	$\frac{1365632}{83521}$	$\frac{-78208}{250563}$					
7	$\frac{1}{2}$	$\frac{-559}{384}$	0	6	$\frac{-204}{47}$	$\frac{14}{39}$	$\frac{-4913}{78208}$				
8	1	$\frac{-625}{224}$	0	12	$\frac{-456}{47}$	$\frac{48}{91}$	$\frac{14739}{136864}$	$\frac{6}{7}$			
9	$\frac{2}{3}$	$\frac{-12253}{99144}$	0	$\frac{16}{27}$	$\frac{16}{459}$	$\frac{29072}{161109}$	$\frac{-2023}{75816}$	$\frac{-112}{12393}$	0		
10	1	$\frac{30517}{2512}$	0	$\frac{-7296}{157}$	$\frac{268728}{7379}$	$\frac{2472}{2041}$	$\frac{-3522621}{10743824}$	$\frac{132}{157}$	0	$\frac{-12393}{4396}$	
$\bar{b}_i$		$\frac{7}{90}$	0	0	$\frac{16}{45}$	$\frac{16}{45}$	0	$\frac{2}{15}$	$\frac{7}{90}$		
b_i		$\frac{2881}{40320}$	0	0	$\frac{1216}{2961}$	$\frac{-2624}{4095}$	$\frac{24137569}{57482880}$	$\frac{-4}{21}$	0	$\frac{4131}{3920}$	$\frac{-157}{1260}$

with the truncation error

$$TE = \left(\frac{-17}{2688}g_1 + \frac{272}{4935}g_4 - \frac{272}{273}g_5 + \frac{24137569}{57482880}g_6 - \frac{34}{105}g_7 - \frac{7}{90}g_8 + \frac{4131}{3920}g_9 - \frac{157}{1260}g_{10}\right)h \ .$$

TABLE 2.11: RK-coefficients of order 6(7) by FEHLBERG /FE68/.

$i \backslash j$	c_i	a_{ij} 0	1	2	3	4	5	6	7	8	b_i	$\bar{b}_i$
0	0	0									$\frac{77}{1440}$	$\frac{11}{864}$
1	$\frac{2}{33}$	$\frac{2}{33}$									0	
2	$\frac{4}{33}$	0	$\frac{4}{33}$								0	
3	$\frac{2}{11}$	$\frac{1}{22}$	0	$\frac{3}{22}$							$\frac{1771561}{6289920}$	
4	$\frac{1}{2}$	$\frac{43}{64}$	0	$-\frac{165}{64}$	$\frac{77}{32}$						$\frac{32}{105}$	
5	$\frac{2}{3}$	$-\frac{2383}{486}$	0	$\frac{1067}{54}$	$-\frac{26312}{1701}$	$\frac{2176}{1701}$					$\frac{243}{2560}$	
6	$\frac{6}{7}$	$\frac{10077}{4802}$	0	$-\frac{5643}{686}$	$\frac{116259}{16807}$	$-\frac{6240}{16807}$	$\frac{1053}{2401}$				$\frac{16807}{74880}$	
7	1	$-\frac{733}{176}$	0	$\frac{141}{8}$	$-\frac{335763}{23296}$	$\frac{216}{77}$	$-\frac{4617}{2816}$	$\frac{7203}{9152}$			$\frac{11}{270}$	0
8	0	$\frac{15}{352}$	0	0	$-\frac{5445}{46592}$	$\frac{18}{77}$	$-\frac{1215}{5632}$	$\frac{1029}{18304}$	0			$\frac{11}{270}$
9	1	$-\frac{1833}{352}$	0	$\frac{141}{8}$	$-\frac{51237}{3584}$	$\frac{18}{7}$	$-\frac{729}{512}$	$\frac{1029}{1408}$	0	1		$\frac{11}{270}$

with the truncation error TE

$$TE = \frac{11}{270}(g_0 + g_7 - g_8 - g_9)h \ .$$

TABLE 2.12: RK-coefficients of order 7(8) by FEHLBERG /FE68/.

$i \backslash j$	c_i	a_{ij} 0	1	2	3	4	5	6	7	8	9	10	11	b_i	$\bar{b}_i$
0	0	0												$\frac{41}{840}$	0
1	$\frac{2}{27}$	$\frac{2}{27}$												0	
2	$\frac{1}{9}$	$\frac{1}{36}$	$\frac{1}{12}$											0	
3	$\frac{1}{6}$	$\frac{1}{24}$	0	$\frac{1}{8}$										0	
4	$\frac{5}{12}$	$\frac{5}{12}$	0	$-\frac{25}{16}$	$\frac{25}{16}$									0	
5	$\frac{1}{2}$	$\frac{1}{20}$	0	0	$\frac{1}{4}$	$\frac{1}{5}$								$\frac{34}{105}$	
6	$\frac{5}{6}$	$-\frac{25}{108}$	0	0	$\frac{125}{108}$	$-\frac{65}{27}$	$\frac{125}{54}$							$\frac{9}{35}$	
7	$\frac{1}{6}$	$\frac{31}{300}$	0	0	0	$\frac{61}{225}$	$-\frac{2}{9}$	$\frac{13}{900}$						$\frac{9}{35}$	
8	$\frac{2}{3}$	2	0	0	$-\frac{53}{6}$	$\frac{704}{45}$	$-\frac{107}{9}$	$\frac{67}{90}$	3					$\frac{9}{280}$	
9	$\frac{1}{3}$	$-\frac{91}{108}$	0	0	$\frac{23}{108}$	$-\frac{976}{135}$	$\frac{311}{54}$	$-\frac{19}{60}$	$\frac{17}{6}$	$-\frac{1}{12}$				$\frac{9}{280}$	
10	1	$\frac{2383}{4100}$	0	0	$-\frac{341}{164}$	$\frac{4496}{1025}$	$-\frac{301}{82}$	$\frac{2133}{4100}$	$\frac{45}{82}$	$\frac{45}{164}$	$\frac{18}{41}$			$\frac{41}{840}$	0
11	0	$\frac{3}{205}$	0	0	0	0	$-\frac{6}{41}$	$-\frac{3}{205}$	$-\frac{3}{41}$	$\frac{3}{41}$	$\frac{6}{41}$	0			$\frac{41}{840}$
12	1	$-\frac{1777}{4100}$	0	0	$-\frac{341}{164}$	$\frac{4496}{1025}$	$-\frac{289}{82}$	$\frac{2193}{4100}$	$\frac{51}{82}$	$\frac{33}{164}$	$\frac{12}{41}$	0	1		$\frac{41}{840}$

with the truncation error

$$TE = \frac{41}{840}(g_0 + g_{10} - g_{11} - g_{12})h.$$

TABLE 2.13: RK-coefficients of order 7(8) by VERNER /VE78/.

i \ j	c_i	1	2	3	4	5	6	7	8	9	10	11	12	13
		a_{ij}												
1	0													
2	$\frac{1}{4}$	$\frac{1}{4}$												
3	$\frac{1}{12}$	$\frac{5}{72}$	$\frac{1}{72}$											
4	$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$										
5	$\frac{2}{5}$	$\frac{106}{125}$	0	$\frac{-408}{125}$	$\frac{352}{125}$									
6	$\frac{1}{2}$	$\frac{1}{48}$	0	0	$\frac{8}{33}$	$\frac{125}{528}$								
7	$\frac{6}{7}$	$\frac{-1263}{2401}$	0	0	$\frac{39936}{26411}$	$\frac{-64125}{26411}$	$\frac{5520}{2401}$							
8	$\frac{1}{7}$	$\frac{37}{392}$	0	0	0	$\frac{1625}{9408}$	$\frac{-2}{15}$	$\frac{61}{6720}$						
9	$\frac{2}{3}$	$\frac{17176}{25515}$	0	0	$\frac{-47104}{25515}$	$\frac{1325}{504}$	$\frac{-41792}{25515}$	$\frac{20237}{145800}$	$\frac{4312}{6075}$					
10	$\frac{2}{7}$	$\frac{-23834}{180075}$	0	0	$\frac{-77824}{1980825}$	$\frac{-636635}{633864}$	$\frac{254048}{300125}$	$\frac{-183}{7000}$	$\frac{8}{11}$	$\frac{-324}{3773}$				
11	1	$\frac{12733}{7600}$	0	0	$\frac{-20032}{3225}$	$\frac{456485}{80256}$	$\frac{-42599}{7125}$	$\frac{339227}{912000}$	$\frac{-1029}{4180}$	$\frac{1701}{1408}$	$\frac{5145}{2432}$			
12	$\frac{1}{3}$	$\frac{-27061}{204120}$	0	0	$\frac{40448}{280665}$	$\frac{-1353775}{1197504}$	$\frac{17662}{25515}$	$\frac{-71687}{1166400}$	$\frac{98}{225}$	$\frac{1}{16}$	$\frac{3773}{11664}$	0		
13	1	$\frac{11203}{8680}$	0	0	$\frac{-38144}{11935}$	$\frac{2354425}{458304}$	$\frac{-84046}{16275}$	$\frac{673309}{1636800}$	$\frac{4704}{8525}$	$\frac{9477}{10912}$	$\frac{-1029}{992}$	0	$\frac{729}{341}$	
$\bar{b}_i$		$\frac{13}{288}$	0	0	0	0	$\frac{32}{125}$	$\frac{31213}{144000}$	$\frac{2401}{12375}$	$\frac{1701}{14080}$	$\frac{2401}{19200}$	$\frac{19}{450}$		
b_i		$\frac{31}{720}$	0	0	0	0	$\frac{16}{75}$	$\frac{16807}{79200}$	$\frac{16807}{79200}$	$\frac{243}{1760}$	0	0	$\frac{243}{1760}$	$\frac{31}{720}$

with the truncation error

$$TE = \left(\frac{-1}{480}g_1 - \frac{16}{375}g_6 - \frac{2401}{528000}g_7 + \frac{2401}{132000}g_8 + \frac{243}{14080}g_9 - \frac{2401}{19200}g_{10} - \frac{19}{450}g_{11} + \frac{243}{1760}g_{12} + \frac{31}{720}g_{13}\right)h \ .$$

TABLE 2.14: RK-coefficients of order 8(9) by FEHLBERG /FE68/.

c(1)	=	0.4436 8940 3764 9818 3109 5994 0428 1370
c(2)	=	0.6655 3410 5647 4727 4664 3991 0642 2055
c(3)	=	0.9983 0115 8471 2091 1996 5986 5963 3083
c(4)	=	0.3155 0000 0000 0000 0000 0000 0000 0000
c(5)	=	0.5054 4100 9481 6906 8626 5161 2673 7384
c(6)	=	0.1714 2857 1428 5714 2857 1428 5714 2857
c(7)	=	0.8285 7142 8571 4285 7142 8571 4285 7143
c(8)	=	0.6654 3968 1210 1156 2534 9537 6925 5586
c(9)	=	0.2487 8317 9680 6265 2069 7222 7456 0771
c(10)	=	0.1090 0000 0000 0000 0000 0000 0000 0000
c(11)	=	0.8910 0000 0000 0000 0000 0000 0000 0000
c(12)	=	0.3995 0000 0000 0000 0000 0000 0000 0000
c(13)	=	0.6005 0000 0000 0000 0000 0000 0000 0000
c(14)	=	1
c(15)	=	0
c(16)	=	1

a(1,o)	=	0.4436 8940 3764 9818 3109 5994 0428 1370
a(2,o)	=	0.1663 8352 6411 8681 8666 0997 7660 5514
a(2,1)	=	0.4991 5057 9235 6045 5998 2993 2981 6541
a(3,o)	=	0.2495 7528 9617 8022 7999 1496 6490 8271
a(3,2)	=	0.7487 2586 8853 4068 3997 4489 9472 4812
a(4,o)	=	0.2066 1891 1634 0060 2426 5567 1039 3185
a(4,2)	=	0.1770 7880 3779 8634 7040 3809 9728 8319
a(4,3)	=	$-0.6819\ 7715\ 4138\ 6949\ 4669\ 3770\ 7681\ 5048 \cdot 10^{-1}$
a(5,o)	=	0.1092 7823 1526 6640 8227 9038 9092 6157
a(5,3)	=	$0.4021\ 5962\ 6423\ 6799\ 5421\ 9905\ 6369\ 0087 \cdot 10^{-2}$
a(5,4)	=	0.3921 4118 1690 7898 0444 3923 3017 4325
a(6,o)	=	$0.9889\ 9281\ 4091\ 6466\ 5304\ 8447\ 6543\ 4355 \cdot 10^{-1}$
a(6,3)	=	$0.3513\ 8370\ 2279\ 6396\ 6951\ 2044\ 8735\ 6703 \cdot 10^{-2}$
a(6,4)	=	0.1247 6099 9831 6001 6621 5206 2587 2489
a(6,5)	=	$-0.5574\ 5546\ 8349\ 8979\ 9643\ 7429\ 0146\ 6348 \cdot 10^{-1}$
a(7,o)	=	-0.3680 6865 2862 4220 3724 1531 0108 0691
a(7,4)	=	$-0.2227\ 3897\ 4694\ 7600\ 7645\ 0240\ 2094\ 4166 \cdot 10^{+1}$
a(7,5)	=	$0.1374\ 2908\ 2567\ 0291\ 0729\ 5656\ 9124\ 5744 \cdot 10^{+1}$
a(7,6)	=	$0.2049\ 7390\ 0271\ 1160\ 3002\ 1593\ 5409\ 2206 \cdot 10^{+1}$
a(8,o)	=	$0.4546\ 7962\ 6413\ 4715\ 0077\ 3519\ 5060\ 3349 \cdot 10^{-1}$

TABLE 2.14 (cont.).

a(8,5)	= 0.3254 2131 7015 8914 7114 6774 6964 8853
a(8,6)	= 0.2847 6660 1385 2790 8888 1824 2057 3667
a(8,7)	= 0.9783 7801 6759 7915 2435 8683 9727 1099 · 10^{-2}
a(9,o)	= 0.6084 2071 0626 2205 7051 0941 4520 5182 · 10^{-1}
a(9,5)	=-0.2118 4565 7440 3700 7526 3252 7525 1206 · 10^{-1}
a(9,6)	= 0.1959 6557 2661 7083 1957 4644 9006 2983
a(9,7)	=-0.4274 2640 3648 1760 3675 1448 3534 2899 · 10^{-2}
a(9,8)	= 0.1743 4365 7368 1491 1955 3234 5255 8189 · 10^{-1}
a(1o,o)	= 0.5405 9783 2959 3191 7355 7857 2411 1182 · 10^{-1}
a(1o,6)	= 0.1102 9325 5978 2892 6539 2831 2764 8228
a(1o,7)	=-0.1256 5008 5200 7255 6414 1477 6378 2250 · 10^{-2}
a(1o,8)	= 0.3679 0043 4775 8146 0136 3840 4356 6339 · 10^{-2}
a(1o,9)	=-0.5778 0542 7703 7207 3940 8406 2857 1866 · 10^{-1}
a(11,o)	= 0.1273 2477 0686 8711 4546 6451 8179 9160
a(11,7)	= 0.1144 8805 0063 9310 5323 6588 7572 1817
a(11,8)	= 0.2877 3020 7036 9799 2776 2022 0184 9198
a(11,9)	= 0.5094 5379 4596 1136 3153 7358 8507 9465
a(11,1o)	=-0.1479 9682 2443 7257 5900 2421 4444 9640
a(12,o)	=-0.3652 6793 8766 1674 0535 8485 4439 4333 · 10^{-2}
a(12,5)	= 0.8162 9896 0123 1891 9777 8194 2124 7030 · 10^{-1}
a(12,6)	=-0.3860 7735 6356 9350 6490 5176 9434 3215
a(12,7)	= 0.3086 2242 9246 0510 6450 4741 6602 5206 · 10^{-1}
a(12,8)	=-0.5807 7254 5283 2060 2815 8293 7473 3518 · 10^{-1}
a(12,9)	= 0.3359 8659 3289 8497 1493 1434 5136 2322
a(12,1o)	= 0.4108 6880 4019 4995 8613 5496 2278 6417
a(12,11)	=-0.1184 0245 9723 5598 5520 6331 5615 4536 · 10^{-1}
a(13,o)	=-0.1237 5357 9212 4514 3254 9790 9613 5669 · 10^{+1}
a(13,5)	=-0.2443 0768 5513 5478 5358 7348 6136 6763 · 10^{+2}
a(13,6)	= 0.5477 9568 9327 7865 6050 4365 2899 1173
a(13,7)	=-0.4441 3863 5334 1324 6374 9598 9656 9346 · 10^{+1}
a(13,8)	= 0.1001 3104 8137 1326 6094 7926 1785 1022 · 10^{+2}
a(13,9)	=-0.1499 5773 1020 5175 8447 1709 8507 3142 · 10^{+2}
a(13,1o)	= 0.5894 6948 5232 1701 3620 8245 3965 1427 · 10^{+1}
a(13,11)	= 0.1738 0377 5034 2898 4877 6168 5744 0542 · 10^{+1}
a(13,12)	= 0.2751 2330 6931 6673 0263 7586 2286 0276 · 10^{+2}
a(14,o)	=-0.3526 0859 3883 3452 2700 5029 5887 5588
a(14,5)	=-0.1839 6103 1448 4827 0375 0441 9898 8231

TABLE 2.14 (cont.).

a(14,6) =-0.6557 0189 4497 4164 5138 0068 7998 5251
a(14,7) =-0.3908 6144 8804 3986 3435 0255 2024 1310
a(14,8) = 0.2679 4646 7128 5002 2936 5844 2327 1209
a(14,9) =-0.1038 3022 9913 8249 0865 7698 5850 7427 $\cdot 10^{+1}$
a(14,1o) = 0.1667 2327 3242 5867 1664 7273 4616 8501 $\cdot 10^{+1}$
a(14,11) = 0.4955 1925 8553 1597 7067 7329 6707 1441
a(14,12) = 0.1139 4001 1323 9706 3228 5867 3814 1784 $\cdot 10^{+1}$
a(14,13) = 0.5133 6696 4246 5861 3688 1990 9719 1534 $\cdot 10^{-1}$
a(15,o) = 0.1046 4847 3406 1481 0391 8730 0240 6755 $\cdot 10^{-2}$
a(15,8) =-0.6716 3886 8449 9028 2237 7784 4617 8020 $\cdot 10^{-2}$
a(15,9) = 0.8182 8762 1894 2502 1265 3300 6524 8999 $\cdot 10^{-2}$
a(15,1o) =-0.4264 0342 8644 8334 7277 1421 3808 7581 $\cdot 10^{-2}$
a(15,11) = 0.2800 9029 4741 6893 6545 9763 3115 3703 $\cdot 10^{-2}$
a(15,12) =-0.8783 5333 8762 3867 6639 0578 1314 5633 $\cdot 10^{-2}$
a(15,13) = 0.1025 4505 1108 2555 8084 2177 6966 4009 $\cdot 10^{-1}$
a(16,o) =-0.1353 6550 7861 7406 7080 4421 6888 9966 $\cdot 10^{+1}$
a(16,5) =-0.1839 6103 1448 4827 0375 0441 9898 8231
a(16,6) =-0.6557 0189 4497 4164 5138 0068 7998 5251
a(16,7) =-0.3908 6144 8804 3986 3435 0255 2024 1310
a(16,8) = 0.2746 6285 5812 9992 5758 9622 0773 2989
a(16,9) =-0.1046 4851 7535 7191 5887 0351 8857 2678 $\cdot 10^{+1}$
a(16,1o) = 0.1671 4967 6671 2315 5012 0044 8830 6588 $\cdot 10^{+1}$
a(16,11) = 0.4952 3916 8258 4180 8131 1869 9074 0287
a(16,12) = 0.1148 1836 4662 7330 1905 2257 9595 4930 $\cdot 10^{+1}$
a(16,13) = 0.4108 2191 3138 3305 5603 9813 2752 7525 $\cdot 10^{-1}$
a(16,15) = 1

b(o) = 0.3225 6083 5002 1624 9913 6129 0096 0247 $\cdot 10^{-1}$
b(8) = 0.2598 3725 2837 1540 3018 8870 2317 1963
b(9) = 0.9284 7805 9965 7702 7788 0637 1430 2190 $\cdot 10^{-1}$
b(1o) = 0.1645 2339 5147 6434 2891 6477 3184 2800
b(11) = 0.1766 5951 6378 6007 4367 0842 9839 7547
b(12) = 0.2392 0102 3203 5275 9374 1089 3332 0941
b(13) = 0.3948 4274 6042 0285 3746 7521 1882 9325 $\cdot 10^{-2}$
b(14) = 0.3072 6495 4758 6064 0406 3683 0552 2124 $\cdot 10^{-1}$

with the truncation error

$$TE = b(14)(g_o+g_{14}-g_{15}-g_{16})h \ .$$

TABLE 2.15: RK-coefficients of order 8(9) by VERNER /VE78/.

i \ j	c_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0																
2	$\frac{1}{12}$	$\frac{1}{12}$															
3	$\frac{1}{9}$	$\frac{1}{27}$	$\frac{2}{27}$														
4	$\frac{1}{6}$	$\frac{1}{24}$	0	$\frac{1}{8}$								$\frac{a,b}{c} \Delta \frac{a+b\sqrt{6}}{c}$					
5	$\frac{2,2}{15}$	$\frac{4,94}{375}$	0	$\frac{-94,-84}{135}$	$\frac{318,308}{375}$												
6	$\frac{6,1}{15}$	$\frac{9,-1}{150}$	0	0	$\frac{117,32}{1425}$	$\frac{69,28}{570}$											
7	$\frac{6,-1}{15}$	$\frac{927,-147}{1250}$	0	0	$\frac{-16248,7728}{9375}$	$\frac{-489,179}{3750}$	$\frac{14268,-5798}{9375}$										
8	$\frac{2}{3}$	$\frac{2}{27}$	0	0	0	0	$\frac{16,-1}{54}$	$\frac{16,1}{54}$									
9	$\frac{1}{3}$	$\frac{19}{256}$	0	0	0	0	$\frac{118,-23}{512}$	$\frac{118,23}{512}$	$\frac{-9}{256}$								
10	$\frac{1}{3}$	$\frac{11}{144}$	0	0	0	0	$\frac{268,-1}{864}$	$\frac{268,1}{864}$	$\frac{-1}{16}$	$\frac{-8}{27}$							
11	$\frac{1}{4}$	$\frac{5034,-271}{61440}$	0	0	0	0	0	$\frac{7859,-1626}{10240}$	$\frac{-2232,813}{20480}$	$\frac{-594,271}{960}$	$\frac{657,-813}{3120}$						
12	$\frac{4}{5}$	$\frac{5996,-3794}{405}$	0	0	0	0	$\frac{-4342,-338}{9}$	$\frac{154732,-40538}{135}$	$\frac{-6136,3794}{45}$	$\frac{-340864,242816}{405}$	$\frac{26304,-13176}{45}$	$\frac{-26624}{81}$					
13	$\frac{5}{6}$	$\frac{3793,2168}{103680}$	0	0	0	0	$\frac{4047,2263}{13824}$	$\frac{-232128,40717}{69120}$	$\frac{7947,-2168}{11520}$	$\frac{1048,-542}{405}$	$\frac{-1383,542}{720}$	$\frac{2624}{1053}$	$\frac{3}{1664}$				
14	1	$\frac{-137}{1296}$	0	0	0	0	$\frac{5642,-337}{864}$	$\frac{5642,337}{864}$	$\frac{-299}{48}$	$\frac{184}{81}$	$\frac{-44}{9}$	$\frac{-5120}{1053}$	$\frac{-11}{468}$	$\frac{16}{9}$			
15	$\frac{1}{6}$	$\frac{33617,-2168}{518400}$	0	0	0	0	$\frac{-3846,31}{13824}$	$\frac{155328,-37507}{345600}$	$\frac{-12173,2168}{37800}$	$\frac{27,542}{3024}$	$\frac{-1372,-542}{2400}$	$\frac{320}{567}$	$\frac{-1}{1920}$	$\frac{4}{105}$	0		
16	1	$\frac{-36487,-30352}{279600}$	0	0	0	0	$\frac{-29666,-4499}{7456}$	$\frac{2779182,-615973}{186400}$	$\frac{-94329,91056}{93200}$	$\frac{-232192,121408}{17475}$	$\frac{101376,-22784}{2835}$	$\frac{-169984}{9087}$	$\frac{-87}{30240}$	$\frac{492}{1165}$	0	$\frac{1260}{337}$	
$\hat{b}_i$		$\frac{103}{1680}$	0	0	0	0	0	0	$\frac{-27}{140}$	$\frac{76}{105}$	$\frac{-201}{280}$	$\frac{1024}{1365}$	$\frac{3}{7280}$	$\frac{12}{35}$	$\frac{9}{280}$		
b_i		$\frac{11}{522}$	0	0	0	0	0	0	$\frac{171}{400}$	$\frac{86}{525}$	$\frac{93}{280}$	$\frac{-2048}{6825}$	$\frac{-3}{18200}$	$\frac{19}{175}$	0	$\frac{9}{35}$	$\frac{243}{4200}$

with the truncation error

$$TE = \left(\frac{-7}{409} g_1 + \frac{63}{200} g_8 - \frac{14}{25} g_9 + \frac{11}{10} g_{10} - \frac{1024}{975} g_{11} - \frac{21}{3640} g_{12} - \frac{3}{13} g_{13} - \frac{9}{280} g_{14} + \frac{9}{23} g_{15} + \frac{243}{4200} g_{16} \right) h \quad .$$

TABLE 2.16: RK-coefficients of order 10 by CURTIS /CU75/.

$a_{2,1}$ = +0·14525 18960 31615 05176
c_2 = +0·14525 18960 31615 05176
$a_{3,1}$ = +0·07262 59480 15807 52588
$a_{3,2}$ = +0·07262 59480 15807 52588
c_3 = +0·14525 18960 31615 05176
$a_{4,1}$ = +0·05446 94610 11855 64441
$a_{4,2}$ = 0
$a_{4,3}$ = +0·16340 83830 35566 93323
c_4 = +0·21787 78440 47422 57764
$a_{5,1}$ = +0·54469 46101 18556 44411
$a_{5,2}$ = 0
$a_{5,3}$ = −2·04260 47879 44586 66540
$a_{5,4}$ = +2·04260 47879 44586 66540
c_5 = +0·54469 46101 18556 44411
$a_{6,1}$ = +0·06536 33532 14226 77329
$a_{6,2}$ = 0
$a_{6,3}$ = 0
$a_{6,4}$ = +0·32681 67660 71133 86646
$a_{6,5}$ = +0·26145 34128 56907 09317
c_6 = +0·65363 35321 42267 73293
$a_{7,1}$ = +0·08233 70775 74827 16585
$a_{7,2}$ = 0
$a_{7,3}$ = 0
$a_{7,4}$ = +0·21191 71963 20280 35617
$a_{7,5}$ = −0·03997 34350 80542 18312
$a_{7,6}$ = +0·02037 [illegible]531 75960 06198
c_7 = +0·27465 94919 90525 40088
$a_{8,1}$ = +0·08595 30577 90073 43832
$a_{8,2}$ = 0
$a_{8,3}$ = 0
$a_{8,4}$ = 0
$a_{8,5}$ = 0
$a_{8,6}$ = +0·29117 69478 05885 09603
$a_{8,7}$ = +0·39644 75145 14702 41049
c_8 = +0·77357 75201 10660 94484
$a_{9,1}$ = +0·08612 09348 56069 67550
$a_{9,2}$ = 0
$a_{9,3}$ = 0
$a_{9,4}$ = 0
$a_{9,5}$ = 0
$a_{9,6}$ = +0·13974 64826 82444 20890
$a_{9,7}$ = +0·39510 98495 81567 45999
$a_{9,8}$ = −0·04079 41270 37085 63576
c_9 = +0·58018 31400 82995 70863
$a_{10,1}$ = +0·07233 14442 23379 48078
$a_{10,2}$ = 0
$a_{10,3}$ = 0
$a_{10,4}$ = 0
$a_{10,5}$ = 0
$a_{10,6}$ = +0·22002 76284 68999 81021
$a_{10,7}$ = +0·08789 53342 54367 34013
$a_{10,8}$ = −0·04445 38399 62603 50864
$a_{10,9}$ = −0·21832 82289 48875 46891
c_{10} = +0·11747 23380 35267 65357
$a_{11,1}$ = +0·08947 10093 67311 14229
$a_{11,2}$ = 0
$a_{11,3}$ = 0
$a_{11,4}$ = 0
$a_{11,5}$ = 0
$a_{11,6}$ = +0·39460 08170 28556 18607
$a_{11,7}$ = +0·34430 11367 96333 34877
$a_{11,8}$ = −0·07946 68266 42926 61291
$a_{11,9}$ = −0·39152 18947 89596 61238
$a_{11,10}$ = 0
c_{11} = +0·35738 42417 59677 45184
$a_{12,1}$ = +0·03210 00687 79632 09213
$a_{12,2}$ = 0
$a_{12,3}$ = 0
$a_{12,4}$ = 0
$a_{12,5}$ = 0
$a_{12,6}$ = 0
$a_{12,7}$ = 0
$a_{12,8}$ = −0·00018 46375 99751 20501
$a_{12,9}$ = +0·15608 94025 31321 98608
$a_{12,10}$ = +0·19344 96857 65456 02528
$a_{12,11}$ = +0·26116 12387 63663 64969
c_{12} = +0·64261 57582 40322 54816
$a_{13,1}$ = +0·04423 74932 85249 96327
$a_{13,2}$ = 0
$a_{13,3}$ = 0
$a_{13,4}$ = 0
$a_{13,5}$ = 0
$a_{13,6}$ = 0
$a_{13,7}$ = 0
$a_{13,8}$ = +0·00464 07744 34539 03964
$a_{13,9}$ = +0·04704 66028 26151 36532
$a_{13,10}$ = +0·08620 74994 80114 88160
$a_{13,11}$ = −0·02607 98302 46821 38093
$a_{13,12}$ = −0·03858 02017 43966 21532
c_{13} = +0·11747 23380 35267 65357
$a_{14,1}$ = +0·02318 04671 74294 11567
$a_{14,2}$ = 0
$a_{14,3}$ = 0
$a_{14,4}$ = 0
$a_{14,5}$ = 0
$a_{14,6}$ = 0
$a_{14,7}$ = 0
$a_{14,8}$ = +0·31978 56784 11636 70673
$a_{14,9}$ = +0·59332 33331 84189 86861
$a_{14,10}$ = −0·11171 27832 23452 87347
$a_{14,11}$ = +0·18039 50557 03050 23573
$a_{14,12}$ = −0·45540 14298 85722 07269
$a_{14,13}$ = +0·33295 73406 00736 36584
c_{14} = +0·88252 76619 64732 34643
$a_{15,1}$ = +0·02624 36432 57981 05892
$a_{15,2}$ = 0
$a_{15,3}$ = 0
$a_{15,4}$ = 0
$a_{15,5}$ = 0
$a_{15,6}$ = 0
$a_{15,7}$ = 0

TABLE 2.16 (cont.).

$a_{15,8}$ = +0·04863 13942 38672 66107
$a_{15,9}$ = +0·04274 38253 83464 78868
$a_{15,10}$ = −0·15575 14733 74349 19785
$a_{15,11}$ = +0·13260 47194 91765 23318
$a_{15,12}$ = −0·09402 96215 29465 15652
$a_{15,13}$ = +0·36891 19544 21896 67414
$a_{15,14}$ = −0·01197 02001 30288 60976
c_{15} = +0·35738 42417 59677 45184
$a_{16,1}$ = +0·10284 18616 86822 30957
$a_{16,2}$ = 0
$a_{16,3}$ = 0
$a_{16,4}$ = 0
$a_{16,5}$ = 0
$a_{16,6}$ = 0
$a_{16,7}$ = 0
$a_{16,8}$ = −1·29708 67206 53005 11984
$a_{16,9}$ = −3·30609 69033 14255 58655
$a_{16,10}$ = −0·06747 08496 92334 33385
$a_{16,11}$ = −3·59679 15480 63726 08732
$a_{16,12}$ = +4·00369 56199 26740 87775
$a_{16,13}$ = +0·04100 57488 26127 81166
$a_{16,14}$ = +0·26978 32108 90450 67539
$a_{16,15}$ = +4·49273 53386 33502 00133
c_{16} = +6·64261 57582 40322 5481$\overline{6}$
$a_{17,1}$ = +0·00876 60933 97736 46361
$a_{17,2}$ = 0
$a_{17,3}$ = 0
$a_{17,4}$ = 0
$a_{17,5}$ = 0
$a_{17,6}$ = 0
$a_{17,7}$ = 0
$a_{17,8}$ = +0·58404 29244 96019 59632
$a_{17,9}$ = +1·29877 96899 66251 57393
$a_{17,10}$ = −0·05854 75188 23066 37183
$a_{17,11}$ = +0·96649 29364 08446 17558
$a_{17,12}$ = −1·41862 48359 70972 65394
$a_{17,13}$ = +0·32460 19941 17747 10444
$a_{17,14}$ = −0·05497 11264 10616 89342
$a_{17,15}$ = −0·91544 14072 48110 49093
$a_{17,16}$ = +0·14742 89120 31297 84267
c_{17} = +0·88252 76619 64732 34643
$a_{18,1}$ = +0·10173 66974 11157 66388
$a_{18,2}$ = 0
$a_{18,3}$ = 0
$a_{18,4}$ = 0
$a_{18,5}$ = 0
$a_{18,6}$ = 0
$a_{18,7}$ = 0
$a_{18,8}$ = −1·69621 75532 09432 81071
$a_{18,9}$ = −3·82523 58462 11624 25452
$a_{18,10}$ = −0·05854 75188 23066 37183
$a_{18,11}$ = −2·52076 77892 27152 29120
$a_{18,12}$ = +5·06895 15710 79828 19146
$a_{18,13}$ = +0·03221 83852 50196 89969
$a_{18,14}$ = +0·09661 35792 25014 27296
$a_{18,15}$ = +3·44722 70365 27756 71816
$a_{18,16}$ = −0·20173 27870 73975 92774
$a_{18,17}$ = +0·55575 42250 51297 90987
c_{18} = +1·00000 00000 00000 00000
b_1 = +0·03333 33333 33333 33333
b_2 = 0
b_3 = 0
b_4 = 0
b_5 = 0
b_6 = 0
b_7 = 0
b_8 = 0
b_9 = 0
b_{10} = 0
b_{11} = 0
b_{12} = +0·23119 09904 31452 64709
b_{13} = +0·18923 74781 48923 49016
b_{14} = +0·03153 95796 91487 24836
b_{15} = +0·27742 91885 17743 17651
b_{16} = +0·04623 81980 86290 52942
b_{17} = +0·15769 78984 57436 24180
b_{18} = +0·03333 33333 33333 33333

TABLE 2.17: RK-coefficients of order 1o by HAIRER /HA78/.

C(1) =	0·0000000000000000000+000	A(11, 1) =	2·3406573691335449371 7−002
C(2) =	5·0000000000000000000−001	A(11, 2) =	0·0000000000000000000+000
C(3) =	5·2650910014161216275 1−001	A(11, 3) =	0·0000000000000000000+000
C(4) =	7·8976365021241824412 6−001	A(11, 4) =	0·0000000000000000000+000
C(5) =	3·9392357012567161128 3−001	A(11, 5) =	0·0000000000000000000+000
C(6) =	7·6665398625354880000 0−001	A(11, 6) =	9·4493130189496180224 0−002
C(7) =	2·8976365021241824412 6−001	A(11, 7) =	−2·7287205590195641900 6−001
C(8) =	1·0847768921956621294 0−001	A(11, 8) =	2·2402204611559220741 0−001
C(9) =	3·5738424175967745184 3−001	A(11, 9) =	6·0438144107513509541 6−001
C(10) =	8·8252766196473246642 6−001	A(11,10) =	−3·0815376929279965258 6−002
C(11) =	6·4261575824032254815 7−001	A(12, 1) =	4·5443775310176369940 8−002
C(12) =	1.1747233803526765357 4−001	A(12, 2) =	0·0000000000000000000+000
C(13) =	7·6665398625354880000 0−001	A(12, 3) =	0·0000000000000000000+000
C(14) =	2·8976365021241824412 6−001	A(12, 4) =	0·0000000000000000000+000
C(15) =	5·2650910014161216275 1−001	A(12, 5) =	0·0000000000000000000+000
C(16) =	5·0000000000000000000−001	A(12, 6) =	−1·1879966718644156772 3−003
C(17) =	1·0000000000000000000+000	A(12, 7) =	1·2035654990928113480 3−002
A(2, 1) =	5·0000000000000000000−001	A(12, 8) =	7·5126902987647924059 1−002
A(3, 1) =	2·4929726760968197801 3−001	A(12, 9) =	−1·8220924098884569041 2−002
A(3, 2) =	2·7721183253193018473 8−001	A(12,10) =	−2·5715285408406504285 5−004
A(4, 1) =	1·9744091255310456103 2−001	A(12,11) =	4·5320783713482958550 6−003
A(4, 2) =	0·0000000000000000000+000	A(13, 1) =	1·7840108640043642929 2−001
A(4, 3) =	5·9232273765931368309 5−001	A(13, 2) =	0.0000000000000000000+000
A(5, 1) =	1·9732054862870214090 0−001	A(13, 3) =	0·0000000000000000000+000
A(5, 2) =	0.0000000000000000000+000	A(13, 4) =	1·1015443953863850704 0−001
A(5, 3) =	2·9508333409267185371 1−001	A(13, 5) =	5·2518612937044877288 4−001
A(5, 4) =	−9·8480312595702383327 7−002	A(13, 6) =	−4·8914859182043621280 3−001
A(6, 1) =	1·3131341734461520076 −001	A(13, 7) =	9·3244361263513573303 8−001
A(6, 2) =	0.0000000000000000000+000	A(13, 8) =	−7·7447505343983952540 9−001
A(6, 3) =	0·0000000000000000000+000	A(13, 9) =	−1·0549021781393582427 0+000
A(6, 4) =	1·1015443953863850704 0−001	A(13,10) =	1·3104671203415715450 9−001
A(6, 5) =	5·2518612937044877288 4−001	A(13,11) =	5·8704977759948739226 7−001
A(7, 1) =	1·3420034184632240619 3−001	A(13,12) =	6·2089805207487879188 1−001
A(7, 2) =	0·0000000000000000000+000	A(14, 1) =	1·3022080660049779349 6−001
A(7, 3) =	0·0000000000000000000+000	A(14, 2) =	0·0000000000000000000+000
A(7, 4) =	6·9608870328807690807 9−001	A(14, 3) =	0·0000000000000000000+000
A(7, 5) =	2·5049772157033937535 2−001	A(14, 4) =	6·9608870328807690807 9−001
A(7, 6) =	−7·9102311649232044549 8−001	A(14, 5) =	2·5049772157033937535 2−001
A(8, 1) =	7·2218274189662145444 8−002	A(14, 6) =	−7·5894898712960734266 2−001
A(8, 2) =	0·0000000000000000000+000	A(14, 7) =	−1·7151720846348838357 7−001
A(8, 3) =	0·0000000000000000000+000	A(14, 8) =	−3·7021767367890670468 8−001
A(8, 4) =	0·0000000000000000000+000	A(14, 9) =	1·2498100857474734780 2−001
A(8, 5) =	−5·8336322936455036912 6−002	A(14,10) =	3·3531092483726703965 −003
A(8, 6) =	3·0475576685744943792 5−003	A(14,11) =	−6·6325461367615358190 7−003
A(8, 7) =	9·1548180297784610028 6−002	A(14,12) =	4·2911657312161790471 4−001
A(9, 1) =	3·1255008135165617062 0−002	A(14,13) =	−3·7177856782469789310 8−002
A(9, 2) =	0·0000000000000000000+000	A(15, 1) =	2·4929726760968197801 3−001
A(9, 3) =	0·0000000000000000000+000	A(15, 2) =	2·7721183253193018473 8−001
A(9, 4) =	0·0000000000000000000+000	A(15, 3) =	0·0000000000000000000+000
A(9, 5) =	0·0000000000000000000+000	A(15, 4) =	0·0000000000000000000+000
A(9, 6) =	1·0912382154241994687 3−004	A(15, 5) =	0·0000000000000000000+000
A(9, 7) =	1·5672575863099501516 4−001	A(15, 6) =	−1·4594059593608521818 5−001
A(9, 8) =	1·6929435117197439967 0−001	A(15, 7) =	−7·9901589351102947535 8−001
A(10, 1) =	1·1906604414675032144 5−002	A(15, 8) =	0·0000000000000000000+000
A(10, 2) =	0·0000000000000000000+000	A(15, 9) =	0·0000000000000000000+000
A(10, 3) =	0·0000000000000000000+000	A(15,10) =	0·0000000000000000000+000
A(10, 4) =	0·0000000000000000000+000	A(15,11) =	0·0000000000000000000+000
A(10, 5) =	0·0000000000000000000+000	A(15,12) =	0·0000000000000000000+000
A(10, 6) =	2·8343708202460654811 2−001	A(15,13) =	1·4594059593608521818 5−001
A(10, 7) =	−4·1631216757056131505 6−001	A(15,14) =	7·9901589351102947535 8−001
A(10, 8) =	2·6464633394974300483 7−001	A(16, 1) =	5·0000000000000000000−001
A(10, 9) =	7·3884980914626907638 8−001	A(16, 2) =	0·0000000000000000000+000

TABLE 2.17 (cont.).

$A(16,\ 3)$ =	−8·07097076095341093251−001	$A(17,11)$ =	6·95057049459982281780−001
$A(16,\ 4)$ =	0·00000000000000000000+000	$A(17,12)$ =	2·71487376457383239111−001
$A(16,\ 5)$ =	0·00000000000000000000+000	$A(17,13)$ =	5·85423734866589756811−001
$A(16,\ 6)$ =	0·00000000000000000000+000	$A(17,14)$ =	9·58819072213235370429−001
$A(16,\ 7)$ =	0·00000000000000000000+000	$A(17,15)$ =	8·97470163394855184206−001
$A(16,\ 8)$ =	0·00000000000000000000+000	$A(17,16)$ =	5·00000000000000000000−001
$A(16,\ 9)$ =	0·00000000000000000000+000	$B(\ 1)$ =	3·33333333333333333333−002
$A(16,10)$ =	0·00000000000000000000+000	$B(\ 2)$ =	−3·33333333333333333333−002
$A(16,11)$ =	0·00000000000000000000+000	$B(\ 3)$ =	−1·20000000000000000000−001
$A(16,12)$ =	0·00000000000000000000+000	$B(\ 4)$ =	0·00000000000000000000+000
$A(16,13)$ =	0·00000000000000000000+000	$B(\ 5)$ =	0·00000000000000000000+000
$A(16,14)$ =	0·00000000000000000000+000	$B(\ 6)$ =	−1·30000000000000000000−001
$A(16,15)$ =	8·07097076095341093251−001	$B(\ 7)$ =	−1·80000000000000000000−001
$A(17,\ 1)$ =	5·73207954320575412321−002	$B(\ 8)$ =	0·00000000000000000000+000
$A(17,\ 2)$ =	−5·00000000000000000000−001	$B(\ 9)$ =	2·77429188517743176508−001
$A(17,\ 3)$ =	−8·97470163394855120846−001	$B(10)$ =	1·89237478148923490158−001
$A(17,\ 4)$ =	0·00000000000000000000+000	$B(11)$ =	2·77429188517743176508−001
$A(17,\ 5)$ =	0·00000000000000000000+000	$B(12)$ =	1·89237478148923490158−001
$A(17,\ 6)$ =	−1·03991004922695343354+000	$B(13)$ =	1·30000000000000000000−001
$A(17,\ 7)$ =	−4·07357014288385809022−001	$B(14)$ =	1·80000000000000000000−001
$A(17,\ 8)$ =	−1·82830236640741849663−001	$B(15)$ =	1·20000000000000000000−001
$A(17,\ 9)$ =	−3·33659270649225021137−001	$B(16)$ =	3·33333333333333333333−002
$A(17,10)$ =	3·95648542376057924001−001	$B(17)$ =	3·33333333333333333333−002

2.2.3.4.2 Coefficients of implicit methods of the Rosenbrock type .

We take here over the formulae and the notations from /KA81/. Given the initial value problem

$$y'(x) = f(y(x)),\ y(x_o) = y_c \text{ (autonomeous system)} \tag{2.8}$$

then the approximate solution y_1 of the exact solution $y(x_o+h)$ reads

$$y_1 = y_o+h \sum_{i=1}^{s} m_i k_i \ , \quad \text{with} \tag{2.9}$$

$$Ek_i = f(y_o+h \sum_{j=1}^{i-1} a_{ij}k_j)+ \sum_{j=1}^{i-1} c_{ij}k_j \ , \ (i=1,2,3,\ldots,s), \tag{2.10}$$

and

$$E = I-\gamma hf'(y_o),\ f' = \partial f/\partial y \ , \text{ I is the unit matrix.} \tag{2.11}$$

The coefficients a_{ij}, c_{ij}, m_i and γ are constant.

Given the initial value problem

$$y'(x) = f(x;y(x)),\ y(x_o)=y_o \text{ (non-autonomeous system)} \tag{2.12}$$

then by putting $x'=1$, the differential equation becomes autonomeous and the foregoing formalism applies. However, if one leaves (2.12) non-autonomeous then the following formalism is valid

$$y_1 = y_o+h \sum_{i=1}^{s} m_i k_i \ , \tag{2.13}$$

$$Ek_i = f(x_o+A_ih,y_o+ \sum_{j=1}^{i-1} a_{ij}k_j)+B_ihf_x(x_o,y_o)+ \sum_{j=1}^{i-1} c_{ij}k_j \ , \tag{2.14}$$

$(i=1,2,\ldots,s)$,

$$E = I- \gamma hf_y(x_o,y_o) \ , \tag{2.15}$$

where

$$B_1 = \gamma, \quad B_i = \gamma + \sum_{j=1}^{i-1} c_{ij} B_j \ , \ (i=2,s), \qquad (2.16)$$

$$A_i = \sum_{j=1}^{i-1} a_{ij} B_j / \gamma \ (i=1,s) \ .$$

We will now give the tables of these RUNGE-KUTTA processes of the orders 4, 5 and 6.

TABLE 2.18: RK - coefficients of order 4 by GOTTWALD-WANNER /G081/.

$\gamma = .395$	
$a_{21} = .438$	$c_{21} = -1.94347441894707$
$a_{31} = .938948678483428$	$c_{31} = .416957530989189$
$a_{32} = .0730795420615381$	$c_{32} = 1.32396782072923$
$b_1 = .729044879960308$	$c_{41} = 1.51951325778448$
$b_2 = .0541069773272405$	$c_{42} = 1.35370815030093$
$b_3 = .281599362440017$	$c_{43} = -.854151495257539$
$b_4 = e_4 = .25$	$e_1 = -.0190858871999474$
$e_2 = .255608791716455$	$e_3 = -.0863816280897592$

with the error estimator

$$EST = \mathrm{Max}\left(U_{round}, \ \mathrm{Max}_{i=1,n}\left(\left|\sum_{j=1}^{4} e_j k_j(i)\right| \Big/ \mathrm{Max}\left(1, |y_i|, |y_{new}(i)|\right)\right)\right)$$

TABLE 2.19: RK-coefficients of order 5 by KAPS-WANNER /KA81/.

$\gamma = 0.141127125787$	$c_{53} = 0.41864844423231233D+00$
$a_{21} = 0.28225425157410630D+00$	$a_{54} = 0.66019768386713535D+00$
$c_{21} = -0.81524951688460885D+00$	$c_{54} = -0.33062805583808154D+01$
$a_{31} = 0.57116380169300584D+00$	$m_1 = 0.77900694405566295D+00$
$c_{31} = 0.81127189717323099D+01$	$m_2 = 0.37121621947171690D+01$
$a_{32} = 0.12386230035678339D+01$	$m_3 = -0.73417673328703555D+00$
$c_{32} = 0.64300627424554704D+00$	$m_4 = 0.24040545624571883D+01$
$a_{41} = 0.72285966684392441D+00$	$m_5 = 0.59299247626627483D+00$
$c_{41} = 0.18891319022399990D-01$	$A_1 = 0.00000000000000000D+00$
$a_{42} = 0.97672836707474073D+00$	$B_1 = 0.14112712578705315D+00$
$c_{42} = -0.37493862667874616D+01$	$A_2 = 0.28225425157410630D+00$
$a_{43} = -0.32856006264202144D-01$	$B_2 = 0.26073304669844646D-01$
$c_{43} = 0.95094153717403742D-01$	$A_3 = 0.80000000000000000D+00$
$a_{51} = 0.67849717523250110D+00$	$B_3 = 0.13028171350787572D+01$
$c_{51} = 0.53507619725099805D+01$	$A_4 = 0.60000000000000000D+00$
$a_{52} = 0.20208927497707465D+01$	$B_4 = 0.16992460579297673D+00$
$c_{52} = -0.25460873945213962D+01$	$A_5 = 0.85887287421294685D+00$
$a_{53} = -0.10701369811124179D+00$	$B_5 = 0.81348381758071946D+00$

with the truncation error

$$TE = \max_{i=1,n}(|y_{2,i}-\hat{y}_{2,i}|/(2^p-1)\max(1,|y_{2,i}|)),$$

where

$$y_1 = y_o+h\sum_{i=1}^{s} m_i k_i \ , \ y_2 = y_1+h\sum_{i=1}^{s} m_i k_i \ , \ \hat{y}_2 = y_o+2h\sum_{i=1}^{s} m_i k_i \ .$$

TABLE 2.2o: RK-coefficients of order 6 by KAPS-WANNER /KA81/.

γ=o.1731558684271912o

a_{21}	= 0.3463117368543824 1D+000	a_{63}	= −0.5459977078920004 3D+000
c_{21}	= −0.6345339789117937 2D+000	c_{63}	= 0.2446870642642821 9D+001
a_{31}	= 0.5148442282373302 4D+000	a_{64}	= −0.4243149493909408 8D+000
c_{31}	= 0.3643791136548091 3D+001	c_{64}	= 0.2855506312744614 4D+001
a_{32}	= 0.6160785374581451 5D+000	a_{65}	= 0.2750366499404226 4D+001
c_{32}	= 0.1294571359253505 3D+001	c_{65}	= −0.1260054187698477 3D+002
a_{41}	= 0.4208359488053255 5D+000	m_1	= −0.6724685882742295 4D+001
c_{41}	= 0.6916195305095658 5D−001	m_2	= −0.1299820394495902 5D+002
a_{42}	= 0.1239825610127653 6D+000	m_3	= −0.4170968312095445 2D+001
c_{42}	= −0.1290588421884623 4D+000	m_4	= −0.2078645533614133 9D+001
a_{43}	= −0.4366355794760995 4D−002	m_5	= 0.1563844716512527 4D+002
c_{43}	= −0.4053216789976309 3D−001	m_6	= 0.1049303167362137 9D+001
a_{51}	= 0.4450880895262415 2D+000	A_1	= 0.00000000000000000
c_{51}	= 0.1758330974830923 6D+001	B_1	= 0.1731558684271912 0D+000
a_{52}	= 0.1479095643877631 3D+000	A_2	= 0.3463117368543824 1D+000
c_{52}	= 0.1709215875522080 3D+001	B_2	= 0.6328258626215853 3D−001
a_{53}	= −0.1086318923647259 0D−001	A_3	= 0.7400000000000000 0D+000
c_{53}	= −0.5401586802868655 8D−001	B_3	= 0.8860235107579577 8D+000
a_{54}	= 0.1674965571464971 5D+000	A_4	= 0.4438050994204750 6D+000
c_{54}	= −0.1420457348981958 5D+001	B_4	= 0.1410520354549854 5D+000
a_{61}	= −0.7828696586114085 7D+000	A_5	= 0.5800000000000000 0D+000
c_{61}	= 0.9236676868909227 3D+001	B_5	= 0.3375670670615331 0D+000
a_{62}	= −0.1676361164385650 3D+001	A_6	= 0.8268441315728088 1D+000
c_{62}	= 0.1168153345012763 4D+002	B_6	= 0.8290102513088967 3D+000

with the same error estimator as for the process of

TABLE 2.19

2.2.3.5 Concluding remarks.

The derivation of a performant RUNGE-KUTTA method is centered on two main themes:

- the choice of optimal coefficients, i.e., they should permit economical computation, which is to say, there should be a minimum number of function evaluations (stages). In addition, the estimated error should be close to the true error and its computation should require little effort.
- the stability of the methods. THE FACT THAT ALL ERROR ESTIMATIONS OF AVAILABLE RUNGE-KUTTA PROCESSES ARE ONLY LOCALLY VALID RENDERS CONSIDERATIONS ON STABILITY PARTICULARLY IMPORTANT.

If the integration takes place outside or close to the boundary of the stability region the solution risks destabilisation. The knowledge of the stability region of a method is thus crucial. Moreover, knowing this region one can estimate,using a stiffness diagnostic of the system to be solved, prior to the integration, the total computational effort required for the solution.

Stability considerations assume thus a prominent position in numerical integration. Therefore, we will now explore this important field with a view to our needs.

3.0 ON THE STABILITY OF THE METHODS.

A basic problem of the numerical solution of ordinary differential equations is the stability of the method applied. Approximately speaking, the question is, which step-size is permitted for the integration of a given system without risking destabilisation of the solution functions. This problem is nowadays well studied and the so-called stability regions are for the numerical methods well known /LA73/, and /HA76/.

FOR PRACTICAL CALCULATIONS THE FOLLOWING RULE IS HELPFUL,

$$h\lambda = \bar{h} \in R \ , \tag{3.1}$$

where R is the stabilty region of the method considered, h is the step-size, λ is the maximal value of all λ_i , and λ_i are the eigenvalues of the matrix

$$\partial f_i/\partial y_j = \partial y_i'/\partial y_j \ , \quad (i,j=1,2,\ldots,n) \tag{3.2}$$

(see (2.11)).

Note that λ can be complex.

Eq.(3.1) says simply that the step-size h must be chosen such that $\bar{h}$ does not lie outside the stability region.

We give here for convenience stability regions which are relevant in the context of our work.

The stability intervals for explicit RUNGE-KUTTA methods are /LA73,p139/ given in TABLE 3.1.

TABLE 3.1: Stability intervals Re($\bar{h}$) of explicit RUNGE-KUTTA methods.

RK-order	1	2	3	4	7
Interval Re($\bar{h}$) of absolute stability	-2.o	-2.o	-2.51	-2.78	-5.^{x}

x heuristic value

The graphical representation of the corresponding stability regions in the $\bar{h}$-plane is given in FIG. 3.1 /LA73, p.227/.

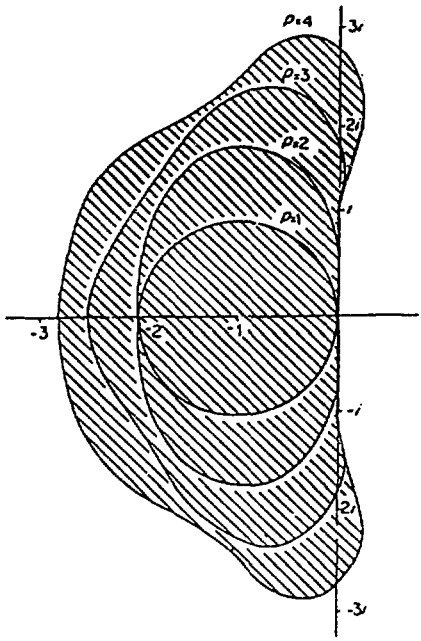


FIG.3.1: Stability regions of explicit RUNGE-KUTTA methods of orders p: 1,2,3,4.

The stability values Re($\bar{h}$) of the ADAM-BASHFORTH-MOULTON methods /LA73,p.1o7/, which are applied in the CODE GEAR, are listed in TABLE 3.2.

TABLE 3.2: Stability intervals Re($\bar{h}$) of the ADAM-BASHFORTH-MOULTON methods.

Order	4	5	6	7	8
Stability intervals	-1.3	-1.o	-o.7	-o.5	-o.4

From TABLE 3.1 and TABLE 3.2 follows that the stability interval Re($\bar{h}$) IS SHRINKING WITH INCREASING ORDER OF THE

ADAM-BASHFORTH-MOULTON methods, whereas it is extending with increasing order of explicit RUNGE-KUTTA methods.

The implicit RUNGE-KUTTA process of the Rosenbrock type of order 4 published by GOTTWALD-WANNER /GO81/ is A-STABLE. The corresponding stability region is indicated in FIG.3.2.

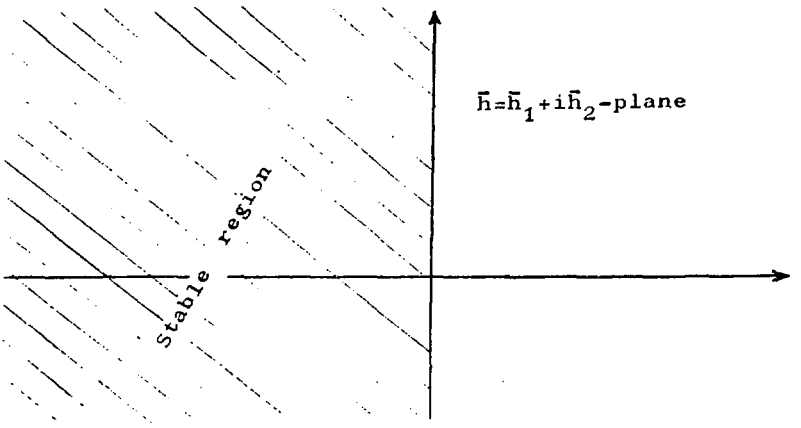


FIG.3.2: Stability region of A-STABLE methods.

The implicit RUNGE-KUTTA process of the Rosenbrock type of order 6 published by KAPS-WANNER /KA81/ under the designation ROW6B is A(α=85.74°)-STABLE. The A(α)-STABILITY region is sketched in FIG.3.3. Note that several RUNGE-KUTTA processes of the Rosenbrock type of order 5 and 6 are published together with their stability regions in /KA81/.

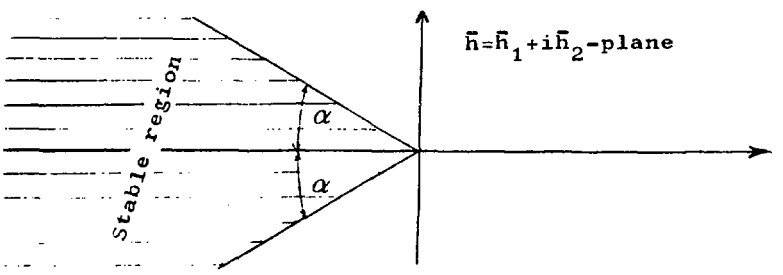


FIG.3.3: A(α)-STABILITY region.

The so-called GEAR's method, which is used in the CODE GEAR, is STIFFLY-STABLE. The corresponding stability regions are sketched in FIG.3.4. For further details see /LA73,p.234/.

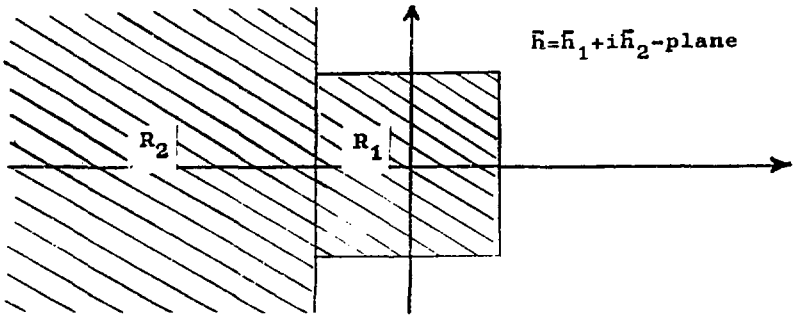


FIG.3.4: Stability regions R_1 and R_2 of STIFFLY-STABLE methods.

From the above considerations we can conclude the following with a view to practical computations:

- The step-size decreases when the order of the ADAM-BASHFORTH-MOULTON methods increases.
- Conversely, the step-size can increase when the order of the explicit RUNGE-KUTTA methods increases.
- If $\lambda = \max(\lambda_i)$ is very large and the path of integration is not very short then neither the ADAM-BASHFORTH-MOULTON methods nor the explicit RUNGE-KUTTA methods can successfully solve the system, because the number of steps required for the integration exceeds any tolerable computational time as well as any acceptable error propagation. In such a case METHODS WITH LARGE ABSOLUTE STABILITY REGION MUST BE USED.
- To solve such problems, the Code GEAR uses the GEAR's method, which is stiffly stable (see FIG.3.4). We will in that case apply the implicit RUNGE-KUTTA methods of the Rosenbrock type of order 4 and 6. They are A-stable (see FIG. 3.2)

and A(α)-stable (see FIG.3.3), respectively.

- Generally speaking, any performant ODE-solver should thus apply two classes of methods, such with large stability regions for the integration of stiff problems, i.e., implicit methods, and such with smaller stability regions for the integration of non-stiff problems. Recall that the latter requires much less computational effort per step than the former and is moreover numerically safer.

STABILITY CONSIDERATIONS GIVE THUS A VALUABLE INSIGHT WITH REGARD TO APPLICABILITY OF THE VARIOUS NUMERICAL METHODS OF SOLUTION OF SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS.

4.0 THE RUNGE-KUTTA SOLVER OSIRIS OF SYSTEMS OF ODEs.

4.1 CRITERIA OF THE CHOICE OF RUNGE-KUTTA METHODS.

To make a choice among the variety of available RUNGE-KUTTA processes we applied the following criteria:

- The RUNGE-KUTTA solver should permit integration of systems of ordinary differential equations of the types characterized in TABLE 4.1.
- The RUNGE-KUTTA solver should enable an automatic optimization of the computational time of an integration.

TABLE 4.1: Characteristics of systems of ODEs which we wish to integrate.

W	10^2-10^4	10^4-10^6	10^6-10^7	10^8-10^{20}
RUNGE-KUTTA method applicable	explicit	explicit (implicit)	implicit (explicit)	implicit

W = Path of integration x $|Re(\lambda_{max})|$.

$|Re(\lambda_{max})/Re(\lambda_{min})|$: 10^6-10^{18} .

Number of ODEs : 10-100 .

λ_i are eigenvalues of the Jacobian matrix (3.2).

According to the first criterion, we have to use both explicit as well as implicit RUNGE-KUTTA methods.
The 2-nd criterion calls for RUNGE-KUTTA methods of low order (for low precision computation) and of high order (for high precision computation).

Results of performance comparison calculations using the various methods which are published in /SH76/, /EN76/, /FE69/, /FE7o/, /VE78/, /GO81/, /KA81/ and /HA78/ together with our own experience have led us to incorporate in our RK-solver code the methods which we list in TABLE 4.2.

TABLE 4.2: RUNGE-KUTTA methods applied in our ODE-solver code.

Author(s)	Ref.	Order	Local error estimator
Explicit methods			
FEHLBERG	/FE7o/	4(5)	given
	/FE69/	7(8)	given
VERNER	/VE78/	8(9)	given
HAIRER	/HA78/	1o	by step-size division
Implicit methods (of the Rosenbrock type):			
GOTTWALD-WANNER	/GO81/	4	given (by step-size division)
KAPS-WANNER	/KA81/	6	given

Comments on TABLE 4.2:

- We found that the FEHLBERG RK-processes are more efficient than those by VERNER. We have included the process 8(9) by VERNER in our RK-solver, because FEHLBERG's processes of orders > 4 fail when applied to solve problems of the kind $y' = f(x)$ (see in 2.2.3.2.2).
- We have included HAIRER's process also, because this is the highest order of explicit RK-processes which is available at present.
- Among the implicit RUNGE-KUTTA processes, those of the Rosenbrock type are the most efficient and the safest, because they offer the advantage of linearizing a problem (see in 2.2.3.2.3).

Moreover, the application of RUNGE-KUTTA processes of low and high order is appropriate for integration with low and high precision.

4.2 DETERMINATION OF THE STIFFNESS OF A SYSTEM.

In addition to the integration methods listed in TABLE 4.2, a diagnosis of the difficulty(stiffness) of the problem is helpful and instructive. To make such a diagnosis, the eigenvalues of the matrix $\partial y_i'/\partial y_j$ must be determined (see 3.2). They give information about the stiffness and offer in principle a criterion to select dynamically the most appropriate integration method.

4.3 THE ERROR CONTROL.

We estimate the uncertainty caused by a single integration step (local error) and verify the total accumulated error (global error).

4.3.1 The local error.

An estimation of the local error is available for each RUNGE-KUTTA process mentioned in TABLE 4.2, except for the process by HAIRER. For this process we estimate the local error by step-size division.

4.3.2 The total accumulated error.

Essentially, we verify this kind of error with the aid of an additional integration imposing a different tolerance. Another possibility to verify this error is a supplementary integration using a code which is not based on RUNGE-KUTTA methods, but, for example, on the multistep methods.

In other words, we can compare results obtained independently with the help of both codes, the RK-solver as well as the Code GEAR.
In some cases, conservation laws can be used to check the accuracy of the computed functions. This is of course the ideal way to determine the total accumulated error.

4.4 THE IMPLEMENTATION OF THE RUNGE-KUTTA SOLVER OSIRIS.

The general structure of the program OSIRIS is schematized in the following FIGURE 4.1.

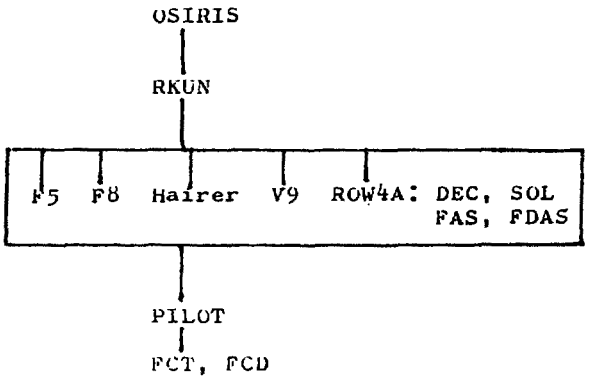


FIGURE 4.1: Scheme of the program OSIRIS.

<u>RKUN</u>: In this subroutine the appropriate algorithm for the integration is chosen. To this end the performance of the various algorithms is memorized: the integration step-sizes h_i (of i-th method) and the number of operations n_i required to perform one iteration (n_i is constant for a given method) are stored and the efficiency $= h_i/n_i$ is determined for each method i.

When the integration starts, the efficiency of each explicit RUNGE-KUTTA method is evaluated with the constrain of maximal 5o iterations per step. Then the integration is continued with the most efficient method until it needs more than 5o iterations per integration step. Then the method previously memorized as the 2-nd most efficient method is applied, etc.. If all explicit RUNGE-KUTTA methods fail to integrate one step with maximal 5o iterations then if the number of iterations > N > 5o, and in any case if the estimated total number of integratio n steps is too high, then implicit methods are called for. The program is returning to explicit methods whenever they are judged to be enough efficient. The criterium for such a switch is based on the eigenvalues of the Jacobian matrix.

<u>F5, F8, Hairer, V9, ROW4A</u>: Designate the subroutines of the algorithm RUNGE-KUTTA-FEHLBERG 4(5), RUNGE-KUTTA-FEHLBERG 7(8), RUNGE-KUTTA-HAIRER (1o), RUNGE-KUTTA-VERNER 8(9) and RUNGE-KUTTA-4 implicit of the Rosenbrock type, respectively. The latter subroutine is that developed by WANNER. (DEC and SOL are subroutines of ROW4A solving a linear algebraic system; FAS and FDAS render a non-autonomeous system automatically autonomeous).

<u>PILOT:</u> This subroutine varies instantaneously the integration step-size in dependence on:

- the ration R=local error/required precision,
- the acceptance of the preceding step-sizes(R<1).

Furthermore, PILOT is estimating continuously the total number of integration steps needed. This number affects the choice of the algorithm to be used.

Users of OSIRIS must furnish the subroutines FCT and FCD.

<u>FCT:</u> FCT(T,Y,DY) defines the system of ODE to be solved, where

$DY(I) = dY_i(t)/dt = f_i(t, Y_1(t), \ldots, Y_I(t))$.

<u>FCD:</u> FCD(T,Y,dF) or DF(I,J) denotes the Jacobian of the system

$\partial f_i / \partial Y_j$.

Users of OSIRIS must furthermore supply the following input data:

- the initial values,
- the precision to which the solution functions should be computed(common relative error for all unknowns),
- the maximal number N of iterations permitted per integration step,
- the upper bound of the integration.

The initialisation of the step-size is automatically done. For this purpose the eigenvalues of the Jacobian matrix are used.

The global error is estimated by means of two integration runs with different tolerance.

5.0 APPLICATION :.

We consider the following system of differential equations

$$dY_i(t)/dt = \sum_j^J \left(\sum_k^K b_{ijk} g_k(t) + c_{ij} \right) y_j(t) \ , \qquad (5.1)$$

where

b_{ijk} and c_{ij} are constants, $i=1,2,3,\ldots,I$.

$j=1,2,3,\ldots,J$,

$k=1,2,3,\ldots,K$.

$g_k(t) = p_k(t).C.e^{\alpha t}$, where α and C are constants.

In the following we give a representative example of problems which we have solved with the aid of OSIRIS.

Input data:

$\alpha = o,3.1o^6$,

$C=o,43.1o^9$ for $t \leq 1o^{-6}$; $C=o$ for $t > 1o^{-6}$. ,

Values of $p_k(t)$, $k=1,2,3,4$:

K=1	K=2	K=3	K=4
0.1000E 00	0.2000E 00	0.3000E 00	0.4000E 00
0.4000E 00	0.3000E 00	0.2000E 00	0.1000E 00

They have to be interpolated (in log(t)) between $t=1o^{-2o}$ (values in first line) and $t=1o^{-6}$ (values in 2-nd line).

Initial values $Y_i(t=t_o)$, $i=1,2,3,\ldots,13$:

1 Y(1)

9 1.E 20 Y(1):o else.

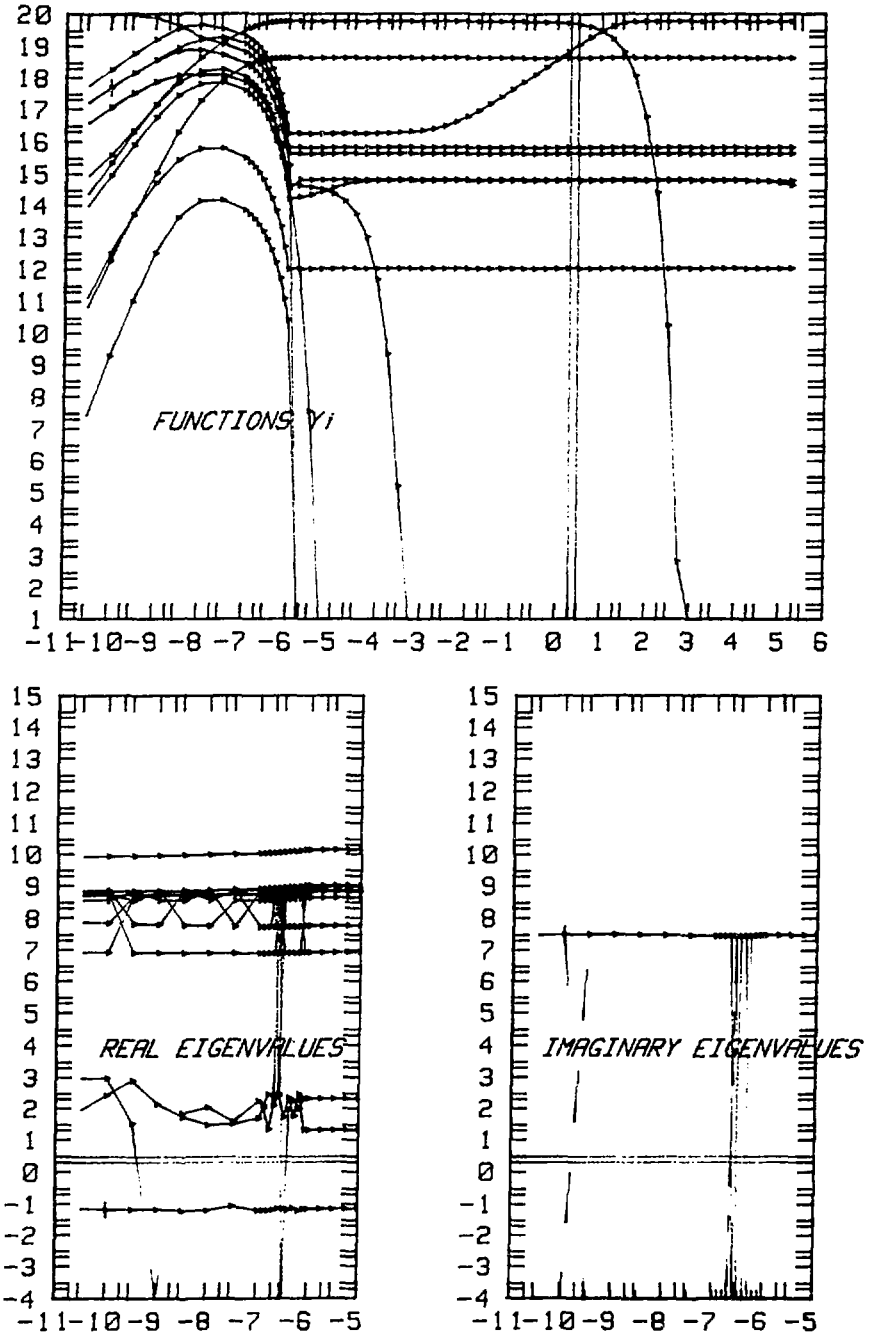
13 1.E20

Values of c_{ij}, i,j=1,2,3,...,13:

I	J	C(I,J)
4	13	1.4811E 06
7	13	2.3105E 07
8	13	4.9511E 05
10	13	4.3322E 07
5	1	1.4 E 08
4	2	7. E 04
8	6	7. E 03
8	7	7. E 06
12	11	7. E-02

Values of b_{ijk}, i,j=1,2,3,...,13;k=1,2,3,4:

I	J	B(I,J,1)	B(I,J,2)	B(I,J,3)	B(I,J,4)
1	5	0. E 00	0. E 00	0.5960E-01	0.1940E 00
1	2	6.2230E 01	0.57 E 00	0.9 E-01	0.2450E-02
2	1	0. E 00	0. E 00	0. E 00	0.479 E 00
2	3	0. E 00	0. E 00	0. E 00	0.071 E 00
2	4	0. E 00	0. E 00	0.133 E 00	0.35 E 00
2	6	4.3000E 00	0.1610E 00	0.0425E 00	0.24 E-02
4	1	0. E 00	0. E 00	0. E 00	0.3970E 00
4	3	0. E 00	0. E 00	0. E 00	0.259 E 00
4	2	0.1870E 00	0.038 E 00	0.33 E 00	0.117 E 00
4	6	1.4140E 00	0.018 E 00	0.0041E 00	0.85 E-03
4	7	0.1290E 00	0.017 E 00	0.0040E 00	1.5 E-04
4	8	0.7780E 00	0.101 E 00	0.024 E 00	0.0011E 00
6	2	0. E 00	0. E 00	0. E 00	0.283 E 00
6	4	0. E 00	0. E 00	0. E 00	0.751 E 00
6	7	0. E 00	0. E 00	0.215 E 00	0.048 E 00
6	8	0. E 00	0. E 00	0.158 E 00	0.0266E 00
6	9	0. E 00	0. E 00	0.21 E 00	0.0023E 00
7	2	0. E 00	0. E 00	0. E 00	0.517 E 00
7	4	0. E 00	0. E 00	0. E 00	0.663 E 00
7	6	0.5060E 00	0.197 E 00	0.5 E 00	0.187 E 00
7	8	0. E 00	0. E 00	0.0453E 00	0.102 E 00
7	9	1.0820E 00	0.177 E 00	0.0183E 00	0.0008E 00
7	10	0.3830E 00	0.063 E 00	0.0065E 00	0.0012E 00
8	2	0. E 00	0. E 00	0. E 00	0.105 E 00
8	4	0. E 00	0. E 00	0. E 00	0.553 E 00
8	6	1.7700E 00	0.245 E 00	0.238 E 00	0.151 E 00
8	7	2.6000E-08	1.4 E-07	1.2 E-03	1.48 E-02
8	9	0.2800E 00	0.003 E 00	0.001 E 00	0.0001E 00
8	10	1.2200E 00	0.163 E 00	0.019 E 00	0.0019E 00
9	6	0. E 00	0. E 00	0. E 00	0.564 E 00
9	7	0. E 00	0. E 00	0. E 00	0.14 E 00
9	8	0. E 00	0. E 00	0. E 00	0.132 E 00
9	10	0. E 00	0. E 00	0.0907E 00	0.482 E 00
9	11	1.2800E 00	0.1315E 00	0.0047E 00	1.38 E-03
9	12	0.0010E 00	1.6 E-04	2.3 E-04	7.7 E-04
10	6	0. E 00	0. E 00	0. E 00	0.945 E 00
10	7	0. E 00	0. E 00	0. E 00	0.093 E 00
10	8	0. E 00	0. E 00	0. E 00	0.41 E 00
10	9	1.7390E 00	0.2 E 00	0.171 E 00	0.178 E 00
10	11	0.0650E 00	0.0043E 00	0.0011E 00	0.0005E 00
10	12	0.2060E 00	0.013 E 00	0.0033E 00	0.0016E 00
11	9	0. E 00	0. E 00	0. E 00	0.51 E 00
11	10	0. E 00	0. E 00	0. E 00	0.931 E 00
11	12	0. E 00	0. E 00	0.005 E 00	0.076 E 00



Explanation of the figures (next page).

- the numbers on the scale are exponents, for example, -11 means 10^{-11},
- the eigenvalues of the Jacobian matrix remain constant in the interval $t=10^{-6}$ to 10^{6}.

The calculations for this particular example have been carried out on a MITRA-125. Computing time: 216 minutes. This corresponds to about 1 minute on a CDC-7600.

ACKNOWLEDGEMENTS

We are much indebted to Professor Gerhard WANNER for his program ROW4A (implicit RUNGE-KUTTA method of the Rosenbrock type of order 4). We are very grateful to Olivier BERSILLON for the program EIGENP (calculation of eigenvalues and eigenvectors of real matrices) and we thank Roger PERRIER and Denise COTTEN for computing assistance.

<u>ANNEX 1</u>: An Algorithm to solve Systems of non-linear, coupled Ordinary Differential Equations.

We consider the system (1.2) which we repeat here for the sake of convenience

$$dY_i(t)/dt = \sum_{s=1}^{S} g_{is}(t) \prod_{j=1}^{I} Y_j^{r_{isj}}(t) \ , \qquad (A.1)$$

where $i=1,2,3,\ldots,I$. r_{isj} are integers.

The formal solution of (A.1) can be written

$$Y_i(t) = \sum_{n=o}^{\infty} \frac{(t-t_o)^n}{n!} \sum_{H,K} (a_{i,H,K} G^H)_n \prod_{j=1}^{I} y_j^{k_j} \ , \qquad (A.2)$$

where

$$K=k_1k_2k_3\ldots k_I \ , \quad H = \begin{pmatrix} h_{q11} & h_{q12}\ldots h_{q1S} \\ h_{q21} & h_{q22}\ldots h_{q2S} \\ \ldots & \ldots \\ h_{qI1} & h_{qI2}\ldots h_{qIS} \end{pmatrix} \qquad (A.3)$$

with $q=o,1,2,3,\ldots,n-1$ for a term of order n.

If the g_{is} are constants, then q=o for terms of any order.

$$(G^H)_n = \prod_{q=o}^{n-1} \prod_{i=1}^{I} \prod_{s=1}^{S} g_{is}^{h_{qis}} \ , \text{ here q symbolizes the} \qquad (A.4)$$

number of derivations of g_{is}, for example, $g_{is}^{h_{3is}=7} = (\frac{d^3}{dt^3} g_{is})^7$

and $y_j = Y_j(t=t_o)$. (A.5)

Evidently, for the computation of the solution functions Y_i one has first to know the coefficients

$a_{i,H,K}$. By means of mathematical induction one can show that the following recurrence relation is valid

$$(a_{i,H,K})_{n+1} = \sum_{i=1}^{I} \sum_{s=1}^{S} ((k_i+1-r_{isi})a_{i,k_1+D_1-r_{is1},k_2+D_2-r_{is2},}$$

$$,k_3+D_3-r_{is3},\ldots,k_I+D_I-r_{isI};H_{ois})+$$

$$+\sum_{q=1}^{n} \sum_{i=1}^{I} \sum_{s=1}^{S} (h_{q-1,is}+1)a_{i,H_{qis},K})_n \qquad (A.6)$$

with the initial values

$$(a_{i,H,K})_o = (a_{i,0,k_i=1,=0 \text{ else}}) = 1, \text{ for example,}$$
$$(a_{2,0,010\ldots0}) = 1 \; . \qquad (A.7)$$

$D_i=\int\delta(i-p)dp=1$ if $i=p$ and equal to zero else.

$$H_{ois}=H-1_{ois},$$
$$H_{q+1,is}=H+1_{qis}-1_{q+1,is} \quad , \text{ where, for example,} \qquad (A.8)$$

H_{o23} means the element h_{o23} becomes $h_{o23}-1$, and H remains unchanged else.

The recurrence relation (A.6) has the following essential features:

- The coefficients of series terms of order n+1 can be calculated from coefficients of series terms of order n.

  Thereby, the labels H and K of $(a_{i,H,K})_{n+1}$ are automatically determined too.

- It permits calculation of the solution functions

to any order of approximation and thus to any accuracy desired.

- The coefficients $a_{i,H,K}$ have only to be computed once for a given system (A.1). They are neither dependent on the initial values nor on the g_{is}. They can , therefore, be stored on a permanent file. As the labels H,K are known automatically with the coefficients, the quantity G^H in (A.2) is also known and can equally be stored on a permanent file, if the g_{is} are constants. In this latter case the whole integration process is simply reduced to the formation of the initial value dependent part $y_j^{k_j}$ in (A.2) and the summation of the power series. This renders the present method economical.
- If the g_{is} are constants, then only terms with H_{ois}, i.e. only the first two lines of (A.6) remain active. In this case (A.6) is substantially simpler.
- Finally, the recurrence relation (A.6) generates curiously in certain cases, i.e. for certain systems (A.1), number sequences which are known from other mathematical disciplines, for example, from combinatorial theory /SC76/, /DU79/, /SC83/.

To illustrate this algorithm we apply it to solve EULER's equations of motion in a force-free field. They read

$$dY_1(t)/dt = b_1 Y_2 Y_3 ,$$
$$dY_2(t)/dt = b_2 Y_1 Y_3 ,$$
$$dY_3(t)/dt = b_3 Y_1 Y_2 , \text{ where } b_1, b_2, b_3 \text{ are constants.} \qquad (A.1.1)$$

Comparing (A.1.1) with (A.1) one obtains

$b_1=g_{11}$, $b_2=g_{21}$, $b_3=g_{31}$, and

$r_{112}=r_{113}=1$, $r_{211}=r_{213}=1$, $r_{311}=r_{312}=1$, and $r_{isj}=0$ else.

Eq.(2) becomes thus

$$Y_i(t) = \sum_{n=o}^{\infty} \frac{(t-t_o)^n}{n!} \sum_{H,K} (a_{i,H,K}\, G^H)_n\, y_1^{k_1} y_2^{k_2} y_3^{k_3} , \qquad (A.2.1)$$

where

$$H=h_{11}h_{21}h_{31},\ K=k_1k_2k_3 \text{ , and} \qquad (A.3.1)$$

$$(G^H)_n = b_1^{h_{11}} b_2^{h_{21}} b_3^{h_{31}}. \qquad (A.4.1)$$

(A.6) becomes

$$(a_{i,H,K})_{n+1} = ((k_1+1).a_{i,H_{o11},k_1+1,k_2-1,k_3-1}+ (k_2+1).a_{i,H_{o21},k_1-1,k_2+1,k_3-1}+(k_3+1).a_{i,H_{o31},k_1-1,k_2-1,k} \qquad (A.6.1)$$

with the initial values

$(a_{1,H=0,100})=(a_{2,H=0,010})=(a_{3,H=0,001})=1$, and with

$H_{o11}=h_{11}-1,h_{21}h_{31}$, $H_{o21}=h_{11}h_{21}-1,h_{31}$, $H_{o31}=h_{11}h_{21}h_{31}-1$. (A.6.1)

We can make the following concluding remarks:

- By means of the here presented method the solution functions of a system of type (A.1) can be computed directly, which is to say, without being obliged to pass by the intermediate step of determining schemata, up to any approximation desired.
- The new method is economical. This is due to the fact that the computation is split-up into three parts:
  - The determination of the coefficients $a_{H,K}$, which are independent of the initial values, i.e. they are constant for a given system (A.1).

- The calculation of the quantity G^H which is also variable independent, if the g_{is} are constants.
- The formation of the initial value dependent part consisting of the y_j.

A substantial part of the computation has thus only to be done once for a given system (A.1), more precisely, the coefficients $a_{H,K}$ and the quantity G^H can be stored on a permanent file, if the g_{is} are variable independent.

Consequently, the integration is essentially reduced to the summation of the terms of the power series. This results, of course, in a considerable gain of execution time.

- The application of the newly developed algorithm does not involve the solution of algebraic equations at each integration step as this is, for example, the case when applying implicit methods.
- According to our experience the here presented method seems to be particularly efficient for solving non-linear systems of type (A.1) with g_{is} constant.
- The method which has been developed in this ANNEX 1 should be regarded as being complementary to RUNGE-KUTTA and multistep methods.

Finally, we summarize for convenience in TABLE A.1 some characteristics of the numerical methods discussed in this work.

TABLE A.1: Characteristics of numerical methods.

Method	Order of approximation for which coefficients are available	Comment
RUNGE-KUTTA	explicit up to 1o implicit up to 6.	Schemata prior to computation have to be established for each order. Algebraic equations have to be solved, when applying implicit methods.
Multistep	ADAM-BASHFORTH-MOULTON up to 12, GEAR's method up to 5.	
This work	unlimited (recurrence relation for coefficients).	No schemata prior to its usage needed. No algebraic equations to be solved.

BIBLIOGRAPHY

/AL77/ ALEXANDER R., Diagonally Implicit Runge-Kutta Methods for Stiff ODEs, SIAM J.Numer.Anal. 14(1977),pp.1006-1021.

/BI51/ BIEBERBACH L., On the Remainder of the Runge-Kutta Formula in the Theory of Ordinary Differential Equations, ZAMP 2(1951),pp.233-248.

/BU63/ BUTCHER J.C., Coefficients for the Study of Runge-Kutta Integration Processes, J.Australian Math.Soc. 3 (1963),pp.185-201.

/BU64a/ BUTCHER J.C., On Runge-Kutta Processes of High Order, J.Australian Math.Soc. 4(1964),pp.179-194.

/BU64/ BUTCHER J.C., Implicit Runge-Kutta Processes, Math. of Computation 18(1964),pp.50-64.

/BU75/ BUTCHER J.C., An Order Bound for Runge-Kutta Methods, SIAM J.Numer.Anal. 12(1975),pp.304-315.

/CA83/ CASH J.R., Block Runge-Kutta Methods for the Numerical Integration of Initial Value Problems in Ordinary Differential Equations, Part I. The Nonstiff Case, Math. of Computation 40(1983),pp.175-191.

/CA83/ CASH J.R., Block Runge-Kutta Methods for the Numerical Integration of Initial Value Problems in Ordinary Differential Equations, Part II. The Stiff Case, Math. of Computation 40(1983),pp.193-206.

/CO72/ COOPER G.J., VERNER J.H., Some explicit Runge-Kutta methods of High Order, SIAM J.Numer.Anal. 9(1972), pp.389-405.

/CO63/ COOPER G.J., SAYFY A., Additive Runge-Kutta Methods for Stiff Ordinary Differential Equations, Math. of Computation 40(1983),pp.207-210.

/CR75/ CROUZEIX M., Sur l'approximation des equations differentielles opérationnelles linéaires par des méthodes de Runge-Kutta, Thèse presentée à l'Université PARIS VI, Paris 1975.

/CU75/ CURTIS A.R., High-order Explicit Runge-Kutta Formulae, their Uses, and Limitations, J.Inst.Math.Appls. 16(1975), pp.35-55.

/DA63/ DAHLQUIST G., A Special Stability Problem for Linear multistep Methods, BIT 3(1963),pp.27-43.

/DU79/ DUMONT D., A Combinatorial Interpretation for the Schett Recurrance on the Jacobian Elliptic Functions, Math. of Computation 33(1979),pp.1293-1297.

/EH69/ EHLE B.L., On Padé Approximations to the Exponential Function and A-stable Methods for the Solution of Initial Value Problems, Thesis, University of Waterloo, Waterloo, Ontario, Canada, 1969.

/EN67/ ENGLAND R., Automatic Methods for Solving Systems of Ordinary Differential Equations, Thesis (2oo pages + annexes), University of Liverpool, Liverpool 1967.

/EN69/ ENGLAND R., Error Estimates for Runge-Kutta type Solutions to Systems of Ordinary Differential Equations, Comput. J. 12(1969),pp.166-17o.

/EN75/ ENRIGHT W.H., HULL T.E., LINDBERG B., Comparing Numerical Methods for Stiff Systems of ODEs, BIT 15(1975),pp.1o-48.

/EN76/ ENRIGHT W.H., Test Results on Initial Value Methods for Non-stiff Ordinary Differential Equations, SIAM J.Numer.Anal. 13(1976),pp.944-961.

/FE68/ FEHLBERG E., Classical Fifth-, Sixth-, Seventh-, and Eight-Order Runge-Kutta Formulas with Step-size Control, NASA-TR-R-287(1968),pp.1-82.

/FE69/ FEHLBERG E., Low Order Classical Runge-Kutta Formulas with Step-size Control and their Application to some Heat Transfer Problems, NASA-TR-R-315(1969),pp.1-43.

/FE69/ FEHLBERG E., Klassische Runge-Kutta Formeln fünfter und siebenter Ordnung mit Schrittweiten-Kontrolle, Computing 4(1969),pp.93-1o6.

/FE7o/ FEHLBERG E., Klassische Runge-Kutta Formeln vierter und niedrigerer Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme, Computing 6(197o),pp.61-71.

/GE71/ GEAR C.W., Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs., N.J., 1971.

/GO81/ GOTTWALD B.A., WANNER G.,A reliable Rosenbrock Integrator for Stiff Differential Equations, Computing 26(1981),pp.355-36o.

/GO82/ GOTTWALD B.A., WANNER G., Comparison of Numerical Methods for Stiff Differential Equations in Biology and Chemistry, Simulation 38 (1982),pp.61-66.

/GR6o/ GRÖBNER W., Die Lie-Reihen und ihre Anwendungen, Verlag der Wissenschaften, Berlin, 196o.

/GR68/ GRAD J., BREBNER M.A., Algorithm 343, Eigenvalues and Eigenvectors of a real General Matrix, Communication of the ACM 11(1968),pp.82o-826.

/HA69/ HAINES C.F., Implicit Integration Processes with Error Estimate for the Numerical Solution of Differential Equations, Comput.Journal 12(1969),pp.183-187.

/HA78/ HAIRER E., A Runge-Kutta Method of Order 1o, J.Inst.Maths.Applics. 21(1978),pp.47-59.

/HA76/ HALL G., WATT J.M., Modern Numerical Methods for Ordinary Differential Equations, Oxford Univ.Press, 1976.

/HI74/ HINDMARSH A.C., GEAR: Ordinary Differential Equation System Solver, UCID-30001 Rev.3, 1974, pp.1-70.

/HU72/ HULL T.E., ENRIGHT W.H., FELLEN B.M., SEDGWICK A.E., Comparing Numerical Methods for Ordinary Differential Equations, SIAM J.Numer.Anal. 9(1972),pp.603-637.

/KA81/ KAPS P., WANNER G., A Study of Rosenbrock type Methods of High Order, Numer.Math. 38(1981),pp.279-298.

/KU01/ KUTTA W., Beitrag zur näherungsweisen Integration totaler Differentialgleichungen, Zeitschrift Math. Phys. 46(1901),pp.435-452.

/LA73/ LAMBERT J.D., Computational Methods in Ordinary Differential Equations, John Wiley, 1973.

/ME57/ MERSON R.H.,An Operational Method for the Study of Integration Processes and Automatic Computing Machines at Weapon Research Establishment, Salisbury , South Australia, 1957.

/NO74/ NORSETT S.P., Semi-explicit Runge-Kutta Methods, Mathematics and Computation No.6/74, 1974.

/RA82/ RACZKA P. jr., On the Class of Simple Solutions of the SU(2) Yang-Mills Equations, Il Nuovo Cim. 72A (1982),pp.289-302.

/RO63/ ROSENBROCK H.H., Some General Implicit Processes for the Numerical Solution of Differential Equations, The Computer Journal 5(1963),pp.329-330.

/RU95/ RUNGE C., Über die numerische Auflösung von Differentialgleichungen, Math.Ann. 46(1895), pp.167-178.

/SA66/ SARAFYAN D., Error Estimation for Runge-Kutta Methods through Pseudo-Iterative Formulas, Technical Report No.14, Louisiana State University in New Orleans, 1966.

/SC76/ SCHETT A., Properties of the Taylor Series Expansion Coefficients of the Jacobian Elliptic Functions, Math. of Computation 3o(1976),pp.143-147.

/SC83/ SCHETT A., On the Solution of the SU(2) YANG-MILLS Equations, to be published.

/SH66/ SHANKS E.B., Solutions of Differential Equations by Evaluations of Functions, Math.Comp. 2o(1966),pp.21-38.

/SH76/ SHAMPINE L.F., WATTS H.A., DAVENPORT S.M., Solving Nonstiff Ordinary Differential Equations, the State of the Art, SIAM Rev. 18(1976),pp.376-411.

/VE78/ VERNER J.H., Explicit Runge-Kutta Methods with Estimates of the Local Truncation Error, SIAM J.Numer. Anal. 15 (1978),pp.772-79o.