AECL 6220
G2

AECL-6220

ATOMIC ENERGY OF CANADA LIMITED

L'ÉNERGIE ATOMIQUE DU CANADA LIMITÉE

ASSESSMENT OF AVAILABLE INTEGRATION ALGORITHMS FOR INITIAL VALUE ORDINARY DIFFERENTIAL EQUATIONS: SELECTION FOR THE CRNL SUBROUTINE LIBRARY AND SIMULATION PACKAGES

Evaluation des algorithmes d'intégration disponibles pour les équations différentielles ordinaires à valeur initiale: sélection pour la bibliothèque des sous-programmes et les ensembles de simulation de Chalk River

M.B. CARVER, J.L. LIU and D.G. STEWART

Chalk River Nuclear Laboratories   Laboratoires nucléaires de Chalk River

Chalk River, Ontario

November 1979 novembre

ATOMIC ENERGY OF CANADA LIMITED

ASSESSMENT OF AVAILABLE INTEGRATION ALGORITHMS FOR INITIAL VALUE ORDINARY DIFFERENTIAL EQUATIONS: SELECTION FOR THE CRNL SUBROUTINE LIBRARY AND SIMULATION PACKAGES

by

M.B. Carver, J.L. Liu*and D.G. Stewart

*Summer Student from University of Toronto

Chalk River Nuclear Laboratories
Chalk River, Ontario KOJ 1JO

1979 November

AECL-6220

Evaluation des algorithmes d'intégration disponibles pour les équations différentielles ordinaires à valeur initiale: sélection pour la bibliothèque des sous-programmes et les ensembles de simulation de Chalk River

par

M.B. Carver, J.L. Liu* et D.G. Stewart

Résumé

Il existe un très grand nombre d'algorithmes conçus pour le problème de la valeur initiale des équations différentielles ordinaires, c'est-à-dire qu'ils permettent d'effectuer l'intégration suivante:

$$\underline{y}(t) = \int_{t_0}^{t} \underline{\underline{f}}(y)dt + \underline{y}(t_0)$$

où $\underline{y}(t)$ et $\underline{y}(t_0)$ sont des vecteurs de colonne et $\underline{\underline{f}}$ une matrice carrée. L'intégration se fait normalement au moyen d'une somme finie à des intervalles de temps choisis dynamiquement pour répondre à une tolérance d'erreur imposée.

Ce rapport décrit la logistique de base du processus d'intégration, il identifie les zones communes de difficulté et il établit un profil d'essai complet pour les algorithmes d'intégration. Un certain nombre d'algorithmes sont décrits et quelques sous-programmes publiés et choisis sont évalués au moyen du profil d'essai.

La conclusion est qu'une bonne bibliothèque destinée à un usage général n'a besoin que de deux sous-programmes de ce genre. Les deux sous-programmes choisis sont des variantes des algorithmes bien connus Gear et Runge-Kutta-Fehlberg. On trouvera dans le rapport une documentation complète et des listes imprimées.

* Etudiant de l'université de Toronto en stage d'été à Chalk River

L'Energie Atomique du Canada, Limitée
Laboratoires nucléaires de Chalk River
Chalk River, Ontario K0J 1J0

Novembre 1979

AECL-6220

ATOMIC ENERGY OF CANADA LIMITED

ASSESSMENT OF AVAILABLE INTEGRATION ALGORITHMS FOR INITIAL VALUE ORDINARY DIFFERENTIAL EQUATIONS: SELECTION FOR THE CRNL SUBROUTINE LIBRARY AND SIMULATION PACKAGES

by

M.B. Carver, J.L. Liu[*] and D.G. Stewart

ABSTRACT

There exists an extremely large number of algorithms designed for the ordinary differential equation initial value problem, that is to perform the integration

$$\underline{y}(t) = \int_{t_0}^{t} \underline{\underline{f}}(y)dt + \underline{y}(t_0)$$

where $\underline{y}(t)$ and $\underline{y}(t_0)$ are column vectors, and $\underline{\underline{f}}$ is a square matrix. The integration is normally done by a finite sum at time intervals which are chosen dynamically to satisfy an imposed error tolerance.

This report describes the basic logistics of the integration process, identifies common areas of difficulty, and establishes a comprehensive test profile for integration algorithms. A number of algorithms are described, and selected published subroutines are evaluated using the test profile.

It concludes that an effective library for general use need have only two such routines. The two selected are versions of the well-known Gear and Runge-Kutta-Fehlberg algorithms. Full documentation and listings are included.

[*] Summer Student from University of Toronto

Chalk River Nuclear Laboratories
Chalk River, Ontario K0J 1J0
1979 November

AECL-6220

CONTENTS

		Page
1.	INTRODUCTION	1
2.	THE INTEGRATION PROBLEM	3
2.1	Fundamental Methods to Integrate a Step	4
2.2	Error Estimation	7
2.3	Tolerance Specification	7
2.4	Step Size Control	7
2.5	Global Error	8
2.6	Round Off Error	8
2.7	Stability, Accuracy and Stiffness	8
3.	SELECTION OF SUITABLE CODES FOR LIBRARY INSTALLATION	10
3.1	Preliminary Assessment	10
3.2	Previous Comparative Studies	10
4.	REQUIREMENTS OF A ROBUST ALGORITHM	12
4.1	The Need for Relative Error Control	13
4.2	Tolerable Error Imposition	13
4.3	Initial Step Size Selection	14
4.4	Jacobian Evaluation and Handling	14
4.5	Sparse Matrix Approach	15
4.6	Discontinuity Detection	16
4.7	Stiffness Detection	16
4.8	Accuracy Level	17
4.9	Equation Set Size	17
5.	TEST RESULTS	17
5.1	Initial Tests	17
5.2	Further Tests	18
5.3	General Assessment of Performance	19
5.4	Large Equation Sets	20
5.5	The Robertson Kinetics Problem	21
5.6	The Belousov Reaction Limit Cycle - A Case for Relative Error Control	22
5.7	Another Case for Relative Error Control	22
6.	CONCLUSION	22
7.	REFERENCES	23
APPENDIX 1:	Result Summary for All Tests	41
APPENDIX 2:	Documentation for GEARZ	82

ASSESSMENT OF AVAILABLE INTEGRATION ALGORITHMS FOR INITIAL VALUE ORDINARY DIFFERENTIAL EQUATIONS: SELECTION FOR THE CRNL SUBROUTINE LIBRARY AND SIMULATION PACKAGES

by

M.B. Carver, J.L. Liu and D.G. Stewart

1. INTRODUCTION

An extremely large number of algorithms have been proposed for the numerical solution of the ordinary differential equation initial value problem. The design or selection of such algorithms for use in a library of mathematical subroutines for general scientific and engineering use is difficult, as it is impractical to maintain a large number of subroutines which attempt to do the same job with varying degrees of success.

Criteria for selection must include accuracy, efficiency and ease of use, thus one can consider realistically only those algorithms which have options to determine an optimal integration step size, and possibly order, by means of a built-in estimate of the associated truncation error, and are presented as quality software, complete with detailed internal and external documentation which emphasizes the weaknesses as well as the strengths of the method. These restrictions considerably reduce the possibilities, but a number of candidates remain. Typical amongst these are algorithms published by Gear[1], Hindmarsh[2], Byrne[3], Shampine[4] and Krogh[5], all of which satisfy the above criteria.

Because these algorithms are reliable, quality software, numerical analysts have incorporated them in a multitude of applications packages, in which the user is shielded from the complexities involved in management of the integration. This type of application has in fact been their greatest success. To the uninitiated user of a subroutine library, however, their correct implementation can be seen as a prohibitive task.

In order to focus further on quality for general use, the criterion of robustness must be added, a robust algorithm being defined as one which produces the result to the desired accuracy or a clear indication of failure, requires a minimum of effort and does not demand clairvoyance from the user. This criterion is essential because the majority of projected users are not numerical differential equation experts, and therefore, not qualified to make decisions on fundamental issues such as the initial step size, error base, sparsity, and error recovery. Unfortunately, as such decisions are frequently left to the user's discretion in the guise of generality, the application of this robustness criterion eliminates most of the above-mentioned candidate algorithms.

This report attempts to show that such algorithms can be made considerably more robust at the expense of a negligible loss of generality by further automation and simplification of the decision process during start up, integration, error processing, and discontinuity handling. Several specific areas where traditionally required user interaction can be either eliminated or reduced to optional status to improve effectiveness, are identified and discussed.

A number of results of tests on both academic and applications problems are given to illustrate that the modifications proposed above are not only more robust, but frequently more efficient particularly in the most common accuracy range encountered in general use.

The motivation of the testing project was threefold, firstly to reassess and fortify the integration section of the CRNL mathematical subroutine library AELIB[7], secondly to provide a smoother relationship between AELIB and the simulation program FORSIM[8] in which many algorithms had already been tested and/or developed, and thirdly to select an efficient algorithm for the MAKSIM[9] chemical kinetic simulation package which was concurrently under development.

To get an impression of the vast profusion of algorithms available for integration of Ordinary Differential Equations (ODE's), one need merely question a computerized literature data base to get a couple of thousand references. Why then yet another report? Basically because profusion begets confusion, particularly as, on the surface, no two routines appear to have anything in common.

We start, therefore, with a very strong recommendation for uniformity. Without delving at all into the mechanics of integration, we state a priori that if n routines are intended to perform the same job, then whatever their internal differences, those n routines should require the same input information and return output information in the same form; quite possibly taking radically different amounts of time and storage to compute similar results. This statement appears trite, but an examination of any well-known library such as IMSL[10], Harwell[11] and previous versions of AELIB shows that even the routines in one section of one library, say the ODE section, do not have remotely similar calling sequences, so there is certainly no chance for inter-library conformity. It is usually, therefore, very awkward for a user to consider building in alternative methods in any program.

The Sandia library [12] has addressed this problem, authors of quadrature routines [13] are also beginning to, and purveyors of simulation packages have recognized it for some time. Packages such as ACSL [14], DDS [15], FORSIM [16] and LEANS [17] have provided integration algorithm selection at the flip of a data card. However, the approach has been to provide a large number of routines permitting the mechanics of making a choice to be simple, but giving little guidance to establishing reasons behind the choice. In this report, and in the new version of FORSIM [8], this choice is reduced to two routines only, each with clearly defined options. The chemical simulation package, MAKSIM[9], has only one of these integrators as it deals exclusively with stiff equation systems.

2. THE INTEGRATION PROBLEM

A first order ODE

$$\frac{dy}{dt} = f(t,y) \qquad (1)$$

may be represented graphically as a series of slopes. The initial value integration problem is to start at (y_0,t_0), and use the available information y_0, t_0, $f(t_0,y_0)$ to progress to a new point $y(t_0+h_1)$ ensuring that the error in the new point does not exceed a specified tolerable maximum. Obviously for the first new point, only a linear approximation may be used:

$$y_1 = y(t_0+h_1) = y(t_0) + f(t_0,y_0)h \qquad (2)$$

For the next step we now have more information $f(t_0,y_0)$, $f(t+h_1,y_1)$. Although the second value is not exact, we can now use a second order approximation to get to $y(t_0+h_1+h_2)$. Obviously to keep down the probable error in (2), h_1 must be small. Because our next approximation is probably somewhat better, h_2 may perhaps be a little larger. Thus we have the structure of a variable step size variable order integration method, providing we can relate a realistic estimate of the related error to the step size h.

Because we accept a certain error level at each step, our solution will tend to wander away from the true solution, but the deviation will be in random fashion and hopefully, will not accumulate. However, if we used a fixed step h regardless of the associated error, we accept no information feedback from the equations, and unless h is inefficiently small, can be sure that the solution will deviate, as in curve c.

There are many ways of performing a single step of integration from t_i to t_i+h_i, and each have various possibilities of subsequently choosing h_i based on an associated error estimate.

Here, we merely identify various types of algorithm but pay more attention to implementation, as the strengths of any algorithm are only useful when implemented in a robust code which recognizes the practical problems of integrating ODE's.

2.1 Fundamental Methods to Integrate a Step

Taylor Series

The Taylor expansion of y is

$$y(t+h) = y(t) + h\, y'(t) + \frac{h^2}{2} y''(t) +$$

or using the relationship (1)

$$= y(t) + h\, f(g,t) + \frac{h^2}{2} f'(y,t) +$$

$$= y(t) + \sum_{i=1}^{q} \frac{h^i}{i!} f^{(i-1)}(y,t) \qquad (3)$$

Note for the order q=1 we have equation (2) the fundamental linear or Euler method, and that the error associated with choosing a finite order r is the sum of the remaining terms and probably comparable in magnitude to the first omitted term.

We can, therefore, get a good approximation by differentiating (1) successively for each term and assembling (3). As this method often requires involved analytical operations in the function, it is not easily automated. Successful implementations have been reported, but are limited in application to continuously differentiable functions, and are not suitable for the stiff equations discussed below.

Runge-Kutta

The Runge-Kutta technique is to select a formula equivalent to (3) by using additional alternate points within the interval h.

Thus we write

$$y(t+h) = y(t) + h \sum_{i=1}^{q} w_i k_i \qquad (4)$$

using successively available amounts of information to define k.

$$k_1 = f(t,y)$$

$$k_2 = f(t + a_{20}h,\ y + a_{21}k_1)$$

$$k_3 = f(t + a_{30}h,\ y + a_{31}k_1 + a_{32}k_2)$$

etc.

and then equate (3) and (4) to determine the w and a coefficients. The resulting system is under specified so one is left with a choice of parameter which defines the Runge-Kutta subset for a given order q.

For example, for one additional internal point giving a second order formula, we have

$$y(t+h) = y(t) + h(w_1 f(t,y) + w_2 f(t + ah,\ y + bf(t,y))) \qquad (5)$$

which can be expanded to first order as

$$y(t) + h(w_1 f(t,y) + w_2 (f(t,y) + ahf_t(t,y) + bf(t,y)f_y(t,y)) \qquad (6)$$

expanding the first two terms of (3) by using

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y}\frac{dy}{dt} = f_t + f_y f$$

we have

$$y(t+h) \quad y(t) + \frac{h}{2}(f_t(t,y) + f_y(t,y)f(t,y)) \qquad (7)$$

and comparing (6) and (7) yields $w_1+w_2 = 1$ and $w_2 a = w_2 b = 1/2$, which leave us free to choose one parameter.

Choosing $w_2 = 1/2$ gives

$$y(t+h) = y(t) + \frac{h}{2}(f(t,y) + f(t+h,y+hf(t,y)) \qquad (8)$$

which has been called the improved Euler method. One can also choose $w_2 =1$ giving

$$y(t+h) = y(t) + hf(t + \frac{h}{2},\ y + \frac{h}{2}\ f(t,y)) \qquad (9)$$

also known as the mid-point rule.

A pth order formula may be developed in a similar manner. P function evaluations are required if $p \leq 4$, but more than p are required for higher orders, hence the popularity of fourth order Runge-Kutta.

Implicit Formulae

A more accurate second order formulae than (8) or (9) would be

$$y(t+h) = y(t) + \frac{h}{2}[f(t,y) + f(t+h,y(t+h))] \qquad (10)$$

which is called implicit, as the derivative value at the new point is only known when the new point itself is known. To solve this directly, it would be necessary to attempt a nonlinear equation solution, a whole field in itself.

Predictor Correctors

One can approximate (10) by using the simple Euler formula

$$y(t+h) = y(t) + hf(t,y(t))$$

as a first step, and then using this prediction to compute $f(t+h),y(t+h)_1)$ and obtain a corrected value

$$y(t+h)_{i=1} = y(t) + \frac{h}{2}[f(t,y(t)) + f(t+h,y(t+h)_i)] \qquad (11)$$

This is a simple predictor corrector method and if terminated at i=1, is equivalent to (8). It may be continued until

$$|y(t+h)_{i+1} - y(t+h)_i|$$

converges to tolerance.

Multi-step Methods

The above are all single step methods as they comprise only the current step. As mentioned previously, after the initial step, we have the past history available for use. Thus a general formula can be written

$$y_{n+1} = \sum_{i=1}^{n_1+1} \alpha_i y_{n+1-i} + \sum_{j=0}^{n_2+1} \beta_i f_{n+1-j} \qquad (12)$$

that is, take into account the previous n points and derivatives. If β_0 is non zero, the formula is implicit, thus using an explicit predictor $\beta=0$, and an implicit corrector, one can develop sets to account for any order. Equivalences between versions of (12) and (3) may also be established.

The most effective and hence most popular versions of (12) have been the Adams methods, $n_1=1$, $n_2=q$, and the Gear backwards difference methods for stiff equations, $n_2=1$, $n_1=q$.

Extrapolation Methods

The number of terms used in (3), (4) or (12) determines the order of the method which in turn reduces the associated error, which is normally $O(h^q)$. Thus a low order method is satisfactory for very small h but can require a large number of steps which in turn can lead to an accumulation of round off error. It can be advantageous to use an h sequence which permits extrapolation to h=0 without approaching very small values of h. The Richardson method uses the Euler formulae to do this, and the method has been considerably further developed in the higher order rational extrapolation of Bulirsh-Stoer[23].

2.2 Error Estimation

For a fixed order Runge-Kutta method,two types of error estimation are possible.

The Romberg method compares results from taking one step h to two steps h/2, the Fehlberg and Merson methods instead carry along different order formulae in the form of mth and nth order approximations, or mth order approximation and associated nth order error estimate. The first method can be wasteful of function evaluations, 3q evaluations for a 9th order method but does look at a hard or exact error. The second method can be made to use only q+1 function evaluations but is looking at an error estimate,which is not so precise.

For a predictor corrector a fixed order formulae normally has a built in truncation error estimate providing the predictor corrector is itself first iterated to convergence. If the algorithm is variable order, one also needs to know the expected error associated with orders neighbouring the current one.

2.3 Tolerance Specification

The error estimation itself is then used to compare to a given tolerance TOL. The manner in which this tolerance is specified greatly affects the resulting behaviour, and we will discuss this in some detail below. For now let us say a step is acceptable if the error estimate E_{st} associated with y_j satisfies

$$E_{st_j} < \mathrm{TOL*YBASE}_j \qquad (13)$$

then the current step size is acceptable; otherwise, a new step size or higher order must be used.

2.4 Step Size Control

Having established (12) as a criterion we must now adjust h in some manner. Crude algorithms merely halve h if (12) is not satisfied, and double h once in a while when it is. More realistic control is to use (12) to establish an error ratio

$$R = \frac{\mathrm{TOL*YBASE}}{\mathrm{EST}}, \quad \mathrm{EST} \neq 0$$

and to use this to compute the next step size to be attempted

$$h_2 = h_1 * \phi(R) \qquad (14)$$

where ϕ is some function of R related to the method.

2.5 Global Error

The error estimate associated with any of the above methods is merely an approximation of the local error of the current step and contains no information about possible error in the starting point due to accumulated local errors in each step. Normally a user prefers to know his probable accumulated error at certain points in the solution. This is not a simple proposition. In fact it is impractical to attempt to impose a global error criterion, one must as usual impose a local error criterion, but attempt to integrate local errors to accumulate a global error estimation for each equation.

In practice, however, providing the mathematical problem is not unstable, controlling local error does in fact approximately control global error and local error control adjusts step size to maintain stability. Shampine[4] discusses this in some detail, and also has a routine which does provide a global error estimate. However, he indicates this is subject to considerable uncertainty.

2.6 Round Off Error

One of the prime problems in attempting to estimate global error is that round off errors become significant when small tolerances are imposed. For a CDC 6600/170 the 60-bit word gives a last bit uncertainty of the order of 10^{-14} on a unity base. Every calculation performed in floating point arithmetic is subject to this uncertainty. This means that for normal specified tolerances of about 10^{-5}, round off should not accumulate serious errors, but for long running problems or more stringent tolerances, the number of steps required may be large enough to accumulate significant errors due to round off.

2.7 Stability, Accuracy and Stiffness

An ODE of first order possesses only one solution. In attempting to achieve accuracy, multistep formulas represent the equation by difference equations of order $k>1$, which themselves have k solutions. One of these, the *principal solution*, will approximate the true solution of the ODE. The remaining k-1 solutions, termed *extraneous, spurious or parasitic* solutions have no relation to the original problem, but will always appear to some degree in a computed solution. As $h \to 0$, if an extraneous solution grows with the number of steps, the formula or method is said to be *unstable*. In some sense the requirement of accuracy, economy and stability are conflicting and for predictor-corrector formulas of order k this is resolved in practice by choosing a formula in which the degree of accuracy accepted is less than the maximum available, which is $O(h^{2k+1})$. The precise nature of the compromise depends on the type of ODE being solved. An obvious requirement is that an ODE or system whose solutions are decaying must be solved by a method whose spurious solutions all decay. For systems, the significant parameters are *eigenvalues* of the *Jacobian* matrix, defined as $J = \partial\vec{F}/\partial\vec{y}$ where the system of coupled first order ODE's is written in the vector form,

$$\bar{y}'(x) = F(x,\bar{y}). \tag{15}$$

These eigenvalues will not be defined in the ordinary way if the system is nonlinear, but are meaningful if the system is considered to be linearized at any point in the $\bar{y}$-space. Then it can be shown that the stability and accuracy with which the system will be solved depends on the coefficients of the integration formula and on the (complex) quantity $h\lambda$, where λ is the eigenvalue of largest modulus.

A special problem arises in the case of what is called a *stiff* system, in which the eigenvalues are negative and widely separated in value. An example is the following:

$$\begin{aligned} u'(t) &= 998u + 1998v \\ v'(t) &= -999u - 1999v \\ u(0) &= 1, \quad v(0) = 0 \end{aligned} \tag{16}$$

whose solution is

$$\begin{aligned} u &= 2e^{-t} - e^{-1000t} \\ v &= -e^{-t} + e^{-1000t} \end{aligned} \tag{17}$$

The difficulty is that the step size is governed by the time constant, $\lambda = -1000$, whereas the significant part of the solution decays according to e^{-t}. Thus a choice of $h\lambda = 1$, for example, would require 1000 time steps for one e-folding interval. Nonlinear systems may also be stiff, if eigenvalues are considered to exist in the sense described earlier.

Another difficulty can occur when the corrector equation is to be iterated to improve the estimate of y_{n+1}. The equation can be written in the form,

$$y_{n+1} = h\beta_0 F(x_{n+1}, y_{n+1}) - g_n \tag{18}$$

where g_n is a known term, and y and F may be considered either as scalars or vectors. Ordinary functional iteration of this equation will fail to converge unless

$$h\beta_0|\lambda| < 1 \tag{19}$$

If $|\lambda|$ is large, then either h must be chosen very small or a good estimation of the Jacobian must be used in order to obtain convergence of the iterations; this is the same problem encountered in stiff equations.

3. SELECTION OF SUITABLE CODES FOR LIBRARY INSTALLATION

3.1 Preliminary Assessment

During the several years of development and use of the FORSIM pack-program which started in 1971 [16], several integration routines have been used and subsequently replaced. These included CORK, the original Runge-Kutta-Gill algorithm from AELIB [7], FOWL the Fowler-Warten algorithm for stiff equations [18], and DIFSUB the original Gear algorithm [1]. The latter was replaced in FORSIM and AELIB with the CRNL routine GEARZ, which combines the Hindmarsh routines GEAR and GEARB [2] with improved Jacobian generation and built-in sparse matrix option [19]. CORK managed to survive in AELIB despite considerable modification in the FORSIM version, but it was obviously a prime candidate for replacement, and a routine RKFINT, using the Runge-Kutta-Fehlberg technique, and written locally [7], appeared to perform considerably better than CORK.

The above decisions had been made in a pragmatic sense as a result of experience. The discarded routines had proved either less robust, less efficient, or both, than their successors, during a number of applications problems, the combination of GEAR and GEARB is merely of practical advantage, and the additional inclusion of the sparse matrix option provided a new ability to optimize both storage and Jacobian evaluation in a manner which turned out much more practical than the subsequently announed GEARS [20].

Two integration routines only, RKFINT and GEARZ, remained in FORSIM and AELIB, the first a self-contained routine, requiring a minimum of storage is suitable for small non-stiff equations, etc. The latter is a poly-algorithm containing stiff and non-stiff options and is more suited to demanding integration problems.

However, to coordinate the planned revision of the library, with the issue of FORSIM VI and MAKSIM, it was decided to assess the performance of GEARZ and RKFINT in a more systematic manner, and if necessary, replace or revise either.

3.2 Previous Comparative Studies

Enright, Hull et al. (1972-75) [21]

In these studies the authors compared a number of current algorithms on small sets of stiff and non-stiff ODE's. Tested routines included those from the 1967 IBM Scientific Subroutine Package, the 1971 IMSL library, several Runge-Kutta methods, several variable order Adams methods, Gear's original DIFSUB package containing both Adams and BDF stiff formulations, several of the subsequent GEAR mutations due to Hindmarsh [2] and EPISODE [3].

Because of the chronological spread, the papers show a slight variance of conclusion, but basically the gist is for non-stiff equations, where functional evaluations are expensive, one should use variable order Adams; otherwise, for stringent tolerances, the extrapolation methods and most other problems, Runge-Kutta-Fehlberg. Other Runge-Kutta methods were not competitive. For stiff equations the above were not recommended, while the Gear BDF-based routines, the Enright second derivative formulae and an extrapolated trapezoidal rule were satisfactory. No clear recommendation was made concerning the choice between GEAR and EPISODE, both of which are included in the current tests.

Krogh (1974) [22]

This study assembles a comprehensive collection of test problems by more clearly identifying particular properties of ODE sets which could cause problems. It is not a comparative study, being primarily concerned with verifying DVDQ, but the criteria established are useful, and frequently referred to in later studies.

Shampine et al. (1976) [4]

The authors confine their attention to non-stiff equations, studying the variable order Adams codes DVDQ [22], DE/STEP [4] and the DIFSUB and GEAR Adams options, the Runge-Kutta-Fehlberg RK45 [4], and the Bulirsch-Stoer code of Fox [23] referred to as EXTRAP. The latter was competitive only at high imposed tolerances, DVDQ and DE/STEP generally outperformed DIFSUB and GEAR. An interesting comparison is given for choosing between DVDQ, DE, and RK45 on the basis of function evaluation cost.

Thompson (1977) [24]

This extensive collection of test programs is restricted to stiff equation sets, but tests are very well documented and discussed in some detail.

DE/STEP, referred to in this study as DODE, is recommended among the Adams methods, and along with RK45 is found competitive with the GEAR and EPISODE packages for mildly stiff equations. For stiff equations, GEAR and EPISODE are clearly superior to other routines tested, and GEAR is found preferable to EPISODE unless the equations are both stiff and highly nonlinear.

Hindmarsh and Byrne (1977) [25]

Discussions of the application of GEAR and EPISODE to a number of test problems are given in this report. The behaviour peculiar to each test is discussed in unusual detail, and such information is undoubtedly invaluable in optimizing efficient use of these and other codes.

A number of additional papers discussing testing are available, but few comprise anthologies of test programs, and few are concerned primarily with robustness.

4. REQUIREMENTS OF A ROBUST ALGORITHM

We are primarily interested in reliability, and require an algorithm to return results having an associated confidence comparable in magnitude with the imposed tolerable local error. Difficulties encountered by the algorithm should be transcended automatically but reported in monitorable fashion, so that the user can reassume control if he wishes; numerical overflows or inexplicable time limits are unacceptable.

Once the reliability criteria are established, the secondary requirement, robustness, is also of great importance. This comprises not only the ability of the algorithm to negotiate and recover from areas of difficulty without the guidance of a clairvoyant user, but also the ease of use and flexibility of the user interface.

These considerations are fundamental to the design of the integration routines now installed in the CRNL FORTRAN library, the FORSIM partial differential equation simulation package, and the MAKSIM mass action chemical kinetics equation simulation package. These routines, RKFINT and GEARZ, evolved to their present form during the testing program described here. The test results shown justify the selection of these routines.

The GEARZ algorithm remains substantially the same as described in reference [19], and apart from minor housekeeping changes, only two salient changes were made. The first was to consolidate all working storage to follow the above expressed philosophy that routines purporting to do the same job should have identical calling sequences. GEARZ and RKFINT now adhere to this.

The parameter sequence is reduced to

```
(EQNF,Y,N,T,DT,TOL,H,M,IF,WS)
```

where the relevant parameters are respectively the derivative equation evaluation subroutine EQNF, dependent variable vector Y, with N entries, independent variable T, increment in T over which to integrate, error tolerance, integration step size H, option control M, failure return flag IF, and working storage array WS. The inclusion of H is for edification only as its behaviour is an excellent window on the system; the remaining are the minimal number of parameters. The very few others required in practice may be accessed by optional common blocks. The second and more fundamental salient common feature is the treatment of error control discussed below.

4.1 The Need for Relative Error Control

The need for stringent relative error control becomes particularly evident in solving highly nonlinear equations such as those arising from mass action chemical kinetics. Here equation terms are typically second order and frequently third order, so a slight deviation even at small absolute values diffuses through the system and rapidly influences all variables. This is further discussed in Section 5.

In integrating such equations, however, one frequently has further useful information available for testing, such as the presence of charge and stoichiometric balance requirements which can be monitored along with the solution to give a concurrent indication of reliability. When the equations are reduced to simple algebraic form as they normally are in reported test results, these additional criteria are irretrievably lost unless specifically noted.

4.2 Tolerable Error Imposition

An algorithm will deem acceptable the values Y_j computed by integration over the current step, if the computed estimate Est_j of the local truncation error in Y_j satisfies

$$Est_j < TOL*YBASE_j$$

where TOL is the tolerable relative error. $YBASE_j$ is relative to Y_j, but obviously to avoid problems with variables which transcend zero, $YBASE_j$ cannot always be equal to Y_j. Both Gear and Hindmarsh recommend the use of $YBASE_j = YMAX_j$, where $YMAX_j$ is the largest value attained by Y_j during the calculation, reasoning that if absolute errors of the order of TOL*YMAX have been accepted at some point of the calculation, it is unreasonable to impose the smaller absolute errors necessary to maintain true relative error when Y subsequently decreases. Initially, YMAX is set to $Y_j(0)$ or if this is zero, to unity. This gives semi-relative error control in initially non-zero variables and absolute error control to those initially zero. The latter course can give gross relative errors when variables never attain magnitudes approaching unity, and the YMAX approach loses accuracy in oscillating systems. This type of problem is countered by allowing the user to access the $YMAX_j$ in an array, insert starting values, and reduce YMAX when necessary when strict relative error is required. This requires a certain amount of clairvoyance by the user, and the user of a general library or a method of lines package, is not necessarily fully conversant with the intricacies of error control.

A far simpler, and certainly clearer method of error control is to establish the significance levels of variables, that is the level at which a variable may be deemed effectively zero. As it is unlikely that we will numerically integrate to precisely zero, any variable which decays below the significance level may be regarded as zero and relieved of relative error control.

Thus we could establish the relative error control base as

$$\mathrm{YBASE}_j = \mathrm{MAX}(|Y_j|,\mathrm{YSIG}_j). \tag{20}$$

This again requires a certain amount of response from the user, as default values established for YSIG_j, for example, near round off level with respect to unity, will not be acceptable for all applications, and the user must then assign appropriate values. However, the user can rarely justify any rationale for assigning individual values, and in fact this practice is potentially dangerous. The ability to assign individual YSIG_j can, therefore, be dropped as a luxury of dubious value.

The Sandia library integration codes recognize this and use a criterion

$$\mathrm{Est}_j < \mathrm{TOL}*|Y_j| + \mathrm{Eabs} \tag{21}$$

where Eabs is the tolerable absolute error [4]. The two formulations (20) and (21) are almost equivalent; (21) can also be used to impose strict absolute error control if necessary, but introduces a slight anomaly if both TOL and Eabs are used in the region $\mathrm{TOL}*|Y_j| > \mathrm{Est}_j > \mathrm{Eabs},\ |Y_j| > \mathrm{Eabs}$.

4.3 Initial Step Size Selection

The selection of initial step size is again a chore often left to the user, although the reason for this is obscure. In any good integration algorithm, the step size adjustment should be capable of large changes, so a poor guess is quickly corrected. In general, the appropriate step size will be a function of the error tolerance and the Jacobian eigenvalues, but as this information is immediately applied to assess an initial step, a preliminary computation requires unjustifiable overhead and a simple built-in estimate related to the tolerance and the largest initial derivative is almost invariably adequate.

4.4 Jacobian Evaluation and Handling

Stiff integrators require Jacobian evaluations at initial conditions and at intervals during the integration. Experience shows that a large fraction of the computation time can be spent evaluating Jacobian elements. Standard procedure is to compute these numerically by in turn perturbing individual Y_j and assessing $\partial F_i/\partial Y_j$ for each F_i. This requires n function evaluation calls per Jacobian.

It is computationally more efficient to provide a routine which evaluates analytical expressions for Jacobian elements, and for small equation systems it is quite feasible for the user to code the required expressions. For most applications, however, the function evaluation routine arrives at the differential equation definition by means of a number of subsidiary subroutine calls, and the associated Jacobian element definitions are either not transparent to the user or are equally complicated to extract. In a specialized application, however, when the form of the equations is fixed, such as in chemical kinetic systems, an analytical Jacobian routine can be used to advantage, and may even be implemented in sparse form [9].

However, for library use or use in a general simulation package where the form of the equations is arbitrary, numerical means of evaluating the Jacobian are necessary, and the straight perturbation method can usually be improved on. If the Jacobian of an order n is a tridiagonal matrix resulting from a simple PDE solution, for example, it can obviously be evaluated using only three function evaluation calls instead of n, and similar savings are available for any banded system, but are not taken advantage of, for example, in the release versions of EPISODEB or GEARB.

4.5 Sparse Matrix Approach

In particular when the Jacobian is handled as a sparse matrix, the resulting increase in efficiency is not primarily due to reduction in storage and linear equation solution time; the largest economization is in Jacobian evaluation optimization. As an illustration, the Curtis and Reid sparse matrix Jacobian routine [6] used in GEARZ requires only seven function calls to pick up the Jacobian for a 242 equation set in test example 41 quoted below.

In large systems of equations, even in a subroutine in which Jacobian elements are coded analytically, it is difficult to approach this kind of efficiency unless the routine is coded extremely carefully. However, there is something to be gained by coding the analytic Jacobian as a sparse matrix. This is an exercise which certainly will not appeal to a casual user, but is useful in cases for which the equation structure is known, for example, the solution of chemical kinetics equations in the MAKSIM program uses this technique and the gains are clearly illustrated in test case 42 discussed below.

Although the introduction of sparse matrix Jacobian handling considerably reduces overhead, it also introduces another degree of freedom into the integration logic, that is the structure of the sparse matrix. Again in specialized codes such as chemical kinetics, the structure is known in advance and may be established as the equations are assembled. However, the Jacobian structure of an arbitrary equation set must be established numerically, and it is probable that this structure changes and sparsity decreases if certain terms depart from zero as the solution evolves. Thus, storage must be arranged to permit density increase, and a switch must be provided to guard against excessive loss of sparsity. At this point either the structure must be frozen and future departures from zero ignored, or the routine must change to a full matrix method. The former is considerably easier to implement in dynamic storage and merely proceeds slightly less efficiently, requiring a lower step size to converge.

Finally, the problem of when to reassess structure must be addressed. Obviously this is not necessary when integration is progressing well, but must be considered along with reducing order and step size when the routine encounters difficulties, and must be done as the last resort in the hierarchy of corrective measures. In addition to this, if user is aware that the equation system loses sparsity during solution, he may direct that structure be reassessed at regular intervals.

4.6 Discontinuity Detection

A more general definition of an ordinary differential equation set is

$$\overline{Y}' = \overline{F}_i(X,\overline{Y}), \; i = \phi(X,\overline{Y})$$

where ϕ is an integer switching function which changes values at certain critical $\phi_c(X,\overline{Y})$ combinations. Such switches dictate that the integration problem i terminates at some critical point, and a new integration i+1 starts from initial conditions at that point, with at least one element of F_{i+1} differing from F_i. Most error controlled algorithms experience considerable difficulty in precisely locating ϕ_c and restarting, as they attempt to transcend the switch, not being able to do so within imposed error tolerance.

Frequently, particularly when large Jacobians are repeatedly evaluated, routines will run to time limit at a discontinuity or simply fail with a message that the posed problem is insoluble.

A robust integration algorithm has two alternative means of handling discontinuities. The first is to permit an integration step to exceed error bounds at the point at which the step reduction pattern suggests the presence of a discontinuity. The point will then be flagged as suspect. This wastes some computing time but negotiates the discontinuity without requiring user action. The second returns control to the user at the end of each successful step so that possibility of an imminent discontinuity occurring within the next predicted step can be assessed, and the user can then assume step size control if necessary. This procedure is effective, providing the user correctly follows appropriate procedures, such as those discussed in reference [26].

4.7 Stiffness Detection

It is not advisable to attempt to use a non-stiff algorithm for stiff equation sets. Most such algorithms detect stiffness merely by running to time limit because their step size is severely limited. It is more satisfactory to test for suspected stiffness and either abandon the integration, or, if possible, switch to a stiff integration option. The Sandia routine DE assesses stiffness in this manner.

4.8 Accuracy Level

As we are concerned with general use, the accuracy level suitable for most scientific engineering calculations is of primary interest. For this type of calculation 3-6 reliable digits usually suffice. It is unrealistic to carry less than three digits and expect meaningful results and tolerances demanding more than ten digits are liable to suffer from accumulated round-off effects. The CRNL routines thus limit $10^{-10} \leq TOL \leq 10^{-3}$, and the current tests were run at $TOL = 10^{-5}$.

4.9 Equation Set Size

Again in general use there are no foreseeable limits on size of the equation set, moderately large sets of equations are common, and variables may range within $10^{\pm n}$ where n can also be large. Single equations are also encountered and it is not unknown for a user to request sparse Jacobian analysis of a single equation. Protocol for handling such eventualities must be automatic.

5. TEST RESULTS

5.1 Initial Tests

With the exception of CORK, the algorithms selected for testing are all reported in the literature, and are listed in Table 1 along with the appropriate references.

The algorithms were implemented as received or reported, and as we are concerned chiefly with robustness, no attempt to promote uniformity was made. Furthermore, it must be pointed out that the WES routines obtained from W.E. Schiesser have been published as algorithms[27] and are used and controlled in the Lehigh packages[8] but not generally released as separate documented code. Also the SODE package obtained from F.T. Krogh is still under development; we tested a preliminary release and did not take full advantage of the numerous options for intricate user interaction which are a unique feature of SODE[28].

The initial test profile consisted of the eight equation sets in Table 2. These tests have a known analytical solution, and should not be particularly demanding. All algorithms were called with the same control parameter values and each called the same function to evaluate derivatives. The following evaluation criteria were extracted from each algorithm using a standard relative tolerance of 10^{-5}, and an initial step of 10^{-5} where needed.

(a) ER - Actual global error returned (average relative error at the end point).
(b) NF - The number of function evaluations required.
(c) NS - The number of time steps completed.
(d) CP - Execution CPU time required.

Obviously CP should be minimum, but as the equation sets are all quite small, the number of function evaluations and number of time steps are more important. As we have pointed out, the algorithms do not monitor global error, but the proximity of ER to the imposed local tolerable error gives a good indication of the stability of the algorithm[4].

The initial test profile showed that CORK, and the Bulirsch-Stoer routine DESUB[23], lacked robustness. DESUB in particular required about ten times as many function evaluations as any of the others, and returned a larger error. CORK returned inaccurate values for the simple discontinuity of problem 5, and was also inaccurate for the simple equations 1 and 2. The Runge-Kutta-Merson routines WES-1, WES-3 and WES-12[27] were generally more satisfactory, but 1 and 12 became numerically unstable in Test Case 4, and this group consistently returned error an order of magnitude larger than most routines, but required a comparable number of function evaluations [29].

A further test profile of 32 problems was then assembled from cases quoted in the literature and current applications at CRNL. These are summarized in Table 3. Chemical problems feature quite prominently in the set as, firstly the MAKSIM package was concurrently under development, and secondly chemical problems are normally stiff, highly nonlinear and magnitudes of the dependent variables frequently range through several orders of magnitude.

First cases 1 to 40 were run with all the integration algorithms. Cases 41 and 42 are specialized applications and were run only on GEARZ to illustrate the potential of the sparse matrix option.

While establishing the final test profile, exploratory runs confirmed that the above routines were again less reliable. In fact for test cases 1 to 20, CORK satisfactorily completed only 9, WES-1, -3 and -12 completed 11, 15 and 14, respectively, and DESUB completed 16, but took an inordinate amount of time.

A short list of robust routines was then completed, these are designated by an asterisk in Table 1.

5.2 Further Tests

In the GEAR, GEARZ and EPISODE packages, a number of options are available for handling the Jacobian for the predictor-corrector iteration. As we are promoting GEARZ, all the GEARZ options are reported for the tests. They are:

Adams Non-stiff	BDF Stiff	
11	21	Functional iteration, Jacobian not analyzed.
12	22	Newton iteration, Jacobian analyzed as full matrix.
13	23	Diagonal Newton, Jacobian analyzed as diagonal.
14	24	Newton iteration, Jacobian analyzed as banded.
15	25	Newton iteration, Jacobian analyzed as sparse.

All the above assess the Jacobian numerically, 15 and 25 optimize this. The two additional options available in the MAKSIM package, which assemble the Jacobian as a full analytically defined matrix or a sparse analytic matrix, are not included except for problem 42, as their assembly, particularly in the sparse case, is considered beyond the desires and capabilities of a library user.

The relative performance of the various options will be similar for GEARZ and EPISODE so we chose to test only options equivalent to 22 and 23 in GEAR and to 22 in EPISODE, and stress that only equivalent options should be compared.

5.3 General Assessment of Performance

As we are primarily interested in accuracy and robustness, we first assess the total number of successes and failures across the entire test profile using the standard relative tolerance of 10^{-5}, and all options set to default. Immediately apparent is the fact that the non-stiff algorithms fail on most stiff equation problems, although the Sandia routines and SODE, instead of merely running to time limit, detect integration problems and rightly refuse to waste further computer time.

In spite of the fact that the stiff algorithms do not in turn fail on non-stiff equations, it is illuminating to look at overall results for separate groups of stiff and non-stiff or mildly stiff equation sets. Equations 20 through 25 and 32 through 40 are significantly stiff.

Table 4 shows that most of the algorithms negotiate the non-stiff problems fairly acceptably, and that the routines EPISODE and GEAR, which do not use strict relative error control, fare significantly worse than the routines which do. This could be corrected, however, by informed user interaction. The singularity in problem 9 and the discontinuous behaviour of problem 12 are the only ones which cause any of the routines to fail completely.

The stiff equation test set shows a different picture. In Table 5 the GEARZ stiff, full Jacobian option is the only one with a clean sheet (the water radiolysis equations are too populated for the sparse option which switches to the diagonal option. EPISODE, GEAR option 22, and GEARZ options 14, 15, 24, 25 all perform adequately, and it is interesting to note that the functional iteration and diagonal approximation methods perform no better with the BDF option than with the Adams. The diagonal options in GEARZ and in GEAR occasionally return deviant errors in comparison with the other options. This observation agrees with Thompson[24].

Tabular results are given in Appendix 2. Here comments are restricted to cases which best illustrate features of particular interest, as in many of the test cases the algorithms all behave acceptably. We first illustrate typical behaviour by examining results listed in Appendix 2 from the two methods of lines examples tests 13 and 16. The first, the diffusion equation, is moderately stiff and has a decaying solution, so one would expect EPISODE and GEAR to show poor accuracy unless Y_{max} was adjusted. The GEARZ diagonal options also show poor accuracy in this problem, and although the GEARZ sparse matrix options require fewer function evaluations, the associated additional overhead makes computing time slightly more than the full option. The non-stiff Adams routines which do use Jacobian analysis perform better than the other non-stiff options in this moderately stiff case.

In contrast is the results summary for the wave equation, case 16. Naturally, this oscillates, so the non-stiff algorithms perform well. SODE in particular requires few steps, but has a large overhead, and GEAR, GEARZ options without Jacobian evaluation are more efficient than those which evaluate Jacobians. RK45 takes the least amount of time. RKFINT somewhat typically uses a comparably lower step size, returning a more precise result than required. EPISODE is less precise than GEAR in this case.

These two test cases have also been repeated with finer division, and are discussed further below.

5.4 Large Equation Sets

In large stiff equation sets, a large portion of the computation time is absorbed in Jacobian evaluation and manipulation. The algorithms, as received, required modification to accommodate large equation sets, so tests 1-40 contain only sets of up to moderate size. Some idea of the effect of size in Jacobian handling efficiency can be gleaned from test cases 13-15, the diffusion equations solved by the method of lines using 15, 25 and 35 points, as this set is moderately stiff.

Results are summarized in Table 6 in the form of the number of function evaluations, and the relative error averaged over the points at problem completion. Computation time is closely proportional to the number of function evaluations for all the GEAR and

GEARZ options, EPISODE takes a little longer. Note that even at the 35-equation size, the sparse matrix routines already save over 50% of the computing time. As mentioned above, the errors returned by EPISODE and GEAR are excessively large mainly because the results decay with time and no compensating measures were taken in the user routine.

A more demanding problem is the solution of the ANS neutron kinetics benchmark problem ID6-A2 for which a full description and independent solution are given in reference [30]. This generates 968 ordinary differential equations which were solved by partitioning, the 242 equations for fast neutrons being integrated by the GEARZ algorithm and the 726 equations describing slower delayed neutron groups by Euler integration. This problem was not run on other integrators as only the GEARZ sparse matrix option would complete the calculation, as shown in Table 7. In this case, the structure of the Jacobian shown in Figure 1, is such that GEARZ requires only seven functional calls to complete all 58544 Jacobian elements [19].

Test case 42 is included to illustrate the possibilities offered by coding Jacobian elements analytically in sparse matrix form, and again results are shown only for GEARZ. This problem concerns simulation of photochemical action in smog. The chemical model contains 81 reactions involving 50 species. The 50 resulting ordinary differential equations contain second and third order nonlinear terms and the resulting Jacobian is approximately 15% populated when assembled. A full description and independent solution is given in [31]. Results in Table 8 are included as computation times rather than function evaluations, as the overhead for the sparse matrix options is higher per function evaluation, and unlike problem 41, it is possible to complete this calculation using the full matrix methods.

5.5 The Robertson Kinetics Problem

The chemical kinetics problem of Robertson is one of the most commonly used examples and appears twice in this test profile, problem 25 is the usual call, and problem 34 is scaled to diminish the spread of coefficient magnitudes. Hindmarsh and Byrne discuss in detail the application of EPISODE to this problem [25] and state that scaling should not be necessary. It is interesting to note from Table 5, that EPISODE option 22 (numeric Jacobian) completes the scaled problem without difficulty, but becomes unstable in the unscaled problem. This instability is discussed in [25], and attributed to the fact that the eigenvalues of the Jacobian are 0, 0 and $\sim 10^{+4}$, thus small errors can generate a spurious positive eigenvalue; stable results were obtained using an analytic Jacobian. It is anomalous that GEAR and GEARZ options 22 are not subject to this instability. This is the only case in the test problem in which unstable performance by EPISODE is not merely due to our refusal to adjust the Y_{max} error base array.

5.6 The Belousov Reaction Limit Cycle - A Case for Relative Error Control

This example, test problem 33, is also frequently cited. The solution is difficult, not because it is a limit cycle, but because it exhibits extremely sharp, short lived peaks. These are most severe in Y_1, which varies between 1 and 10^7 in the initial few seconds of each cycle, and clearly requires relative error control. It is quite difficult to provide comparisons of accuracy for this problem unless the test time span terminates before the second spike as in the tests of Enright and Hull [21c]. Even then, all routines discussed in [21c] failed to complete this problem under the imposition of very small absolute error tolerances. The semi-relative error (Y_{max}) approach produces instability in Y_1 and Y_3 after the first peak, and Byrne and Hindmarsh [25] adjust Y_{max} dynamically to simulate relative error control. Curtis [32] shows also that the computation fails to adhere to the limit cycle if the tolerated relative error exceeds 10^{-3}. In the current tests, the computation adheres well to the cycle even past the fourth peak when a true relative error of 10^{-5} is used.

5.7 Another Case for Relative Error Control

The chemical pyrolysis example, test problem 21, was found to give a similar response in the range of absolute tolerances (10^{-2}, 10^{-8}) [21c]. In fact, all these tolerances exceed the maximum values of Y_2, Y_3 and Y_4, so even the 10^{-8} absolute tolerance is lax. Semi-relative error control is suitable for this problem providing starting Y_{max} values for Y_2 to Y_4 are set to a more realistic value than unity.

6. CONCLUSION

It is apparent that all the routines selected for the short list of Table 1 behave in a reasonably robust manner throughout the test profile, and are quite suitable for installation in a general library or simulation package. There is considerable evidence, however, that the recommendations of Section 2 improve robustness by automating decisions usually required from the user, and frequently also increase efficiency. In particular, the complete automation of the sparse matrix option greatly increases the potential of the GEAR-based routines.

Storage requirements of the various routines are given in Tables 9 and 10. Because the Adams options in the stiff routines behave quite acceptably, for the general use context, on non-stiff equations, the only motivation for providing a non-stiff solver would appear to be storage economy. For this reason the single subroutine RKFINT remains in use at CRNL. Although the tests show that RK45 is more efficient in many cases, this has been found attributable to a more stringent error criterion in RKFINT, which will be suitably relieved.

A further area of necessary investigation is the occasional erratic behaviour of the diagonal Jacobian option in GEAR and GEARZ also corroborated by Thompson. It is important that this be remedied as the algorithms in FORSIM and MAKSIM normally switch to this option from the sparse or full options if sufficient storage is not available. While this is being resolved, it is probably safer to use the banded option instead for this purpose. The banded option should also be modified to optimize Jacobian evaluation, and then in fact, should resolve to the diagonal case in the absence of bands.

7. REFERENCES

[1] C.W. Gear, The Automatic Integration of Ordinary Differential Equations, Comm. Acm, V14, 3, p.176, March 1971.

[2] A.C. Hindmarsh, The LLL Family of Ordinary Differential Equation Solvers, UCRL-78129, April 1976.

[3] G. Byrne and A.C. Hindmarsh, A Polyalgorithm for the Numerical Solution of Ordinary Differential Equations, ACM TOMS, V1, p.71, 1975.

[4] L.F. Shampine, H.W. Watts and S.M. Davenport, Solving Non-stiff Ordinary Differential Equations - The State of the Art, Siam Review, V18, 3, 1976.

[5] F.T. Krogh, Variable Order Integrators for the Numerical Solution of Ordinary Differential Equations, Jet Propulsion Lab., Section 314 Subroutine Write-up, May 1969.

[6] A.R. Curtis and J.K. Reid, FORTRAN Routines for the Solution of Sparse Sets of Linear Equations, AERE-R-6844, 1971.

[7] L.E. Evans, Editor, The AELIB Users' Manual, Atomic Energy of Canada Limited report AECL-6076, 1978.

[8] M.B. Carver, D.G. Stewart, J.M. Blair and W.N. Selander, The FORSIM VI Package for Automated Solution of Arbitarily Defined PDE/ODE Systems, Atomic Energy of Canada Limited report AECL-5821, February 1978. Comput. Phys. Commun. 17 (1979) 239-282.

[9] M.B. Carver and A.W. Boyd, A Program Package Using Stiff Sparse Techniques for the Automatic Solution of Mass Action Chemical Kinetics, Int. J. Chem. Kinet., October 1979.

[10] IMSL, International Mathematical and Statistical Libraries publication G0004, Houston, Texas, 1978.

[11] M.J. Hopper, Editor, Harwell Subroutine Library, UKEA Harwell report AERE-R7477, 1973.

[12] R.E. Huddleston and T.H. Jefferson, User's Guide to the Sandia Mathematical Program Library, Sandia report SAND76-8209, 1976.

[13] F.T. Krogh, Editor, Los Alamos Quadrature Workshop Panel Discussion, ACM Signum Newsletter, V11, 1, 1976.

[14] E.L. Mitchell, Advanced Continuous Simulation Language, ACSL, "Numerical Methods for Differential Equation Systems", Elsevier, North-Holland, New York, p.135, 1978.

[15] M.G. Zellner, DSS: Distributed System Simulator, Phd. Dissertation, Lehigh University, June 1970.

[16] Carver, M.B. and Likeness, S.L., FORSIM: A FORTRAN Oriented Simulation Program for the Continuous Solution of Systems or Ordinary Differential equations, Atomic Energy of Canada Limited report AECL-3902, May 1971.

[17] W.E. Schiesser, A Digital Simulation System for Mixed Ordinary/Partial Differential Equation Modes, Proceedings, IFAC Symposium on Digital Simulation of Continuous Systems, 2, p.S2-1 to S2-9, September 1971, Gyor, Hungary.

[18] M.E. Fowler and R.M. Warten, A Numerical Integration Technique for Ordinary Differential Equations with Widely Separated Eigenvalues, IBM Journal, September 1967.

[19] M.B. Carver and A.P. Baudouin, Solution of Reactor Kinetics Problems Using Sparse Matrix Technqiues in an ODE Integrator for Stiff Equations, Advances in Computer Methods for Partial Differential Equations, R. Vichnevetsky, Ed., AICA Press, June 1975, Atomic Energy of Canada report AECL-5177, 1975.

[20] J.W. Spellman and A.C. Hindmarsh, GEARS: Solution of Ordinary Differential Equations Having Sparse Jacobian, Lawrence Livermore Laboratory report UCID-30116, August 1975.

[21] T.E. Hull, W.H. Enright, et al., (a) Comparing Numerical Methods for Ordinary Differential Equations, Siam J. Num. Anal. V9, 4, p.603, 1972. (b) Comparing Numerical Methods for Stiff Systems of ODE's, Bit, V15, 1, 1975. (c) Comparing Numerical Methods for Stiff Ordinary Differential Equations Arising in Chemistry, in Numerical Methods for Differential Systems, L. Lapidus and W.E. Schiesser, Editors, Academic, 1975.

[22] F.T. Krogh, On Testing a Subroutine for Numerical integration of Ordinary Differential Equations, J.A.C.M., V4, p.545, 1973.

[23] P.A. Fox, Integration of a First-Order System of Ordinary Differential Equations, in Mathematical Software, J.R. Rice, Editor, Academic, 1971.

[24] S. Thompson, A Comparison of Software for the Numerical Solution of Ordinary Differential Equations, Babcock & Wilson report NPGD-TM-368, June 1977.

[25] G.D. Byrne, A.C. Hindmarsh, et al., Comparative Test Results for Two ODE Solvers EPISODE and GEAR, ANL-77-19.

[26] M.B. Carver and S.R. MacEwen, Simulation of a System Described by Implicitly Defined Ordinary Differential Equations Containing Numerous Discontinuities, Appl. Math. Modl. V2, 4, December 1978.

[27] W.E. Schiesser, A Comparative Study of Merson-type Runge-Kutta Algorithms, Proceedings 1973 Summer Computer Simulation Conference, SCI Press, 1973.

[28] F.T. Krogh, Variable Order Integrators for the Numerical Solution of Ordinary Differential Equations, Jet Propulsion Lab. Technical Brief NPO11643, November 1970.

[29] M.B. Carver, In Search of a Robust Integration Algorithm for General Library Use, Proceedings, ACM SIGNUM Conference on Numerical Ordinary Differential Equations, University of Illinois, April 1979, R.D. Skeel, Editor, p.36.

[30] Argonne Code Center Benchmark Problem Book, ANL-7416-1, Argonne National Laboratory, 1972.

[31] L.A. Farrow and D. Edelson, The Steady State Approximation, Fact or Fiction, Int. J. Chem. Kinet. VI, p.787, 1974.

[32] A.R. Curtis, Solution of Large Stiff Initial Value Problems, The State of the Art, Proceedings IMA Conference on Numerical Software Needs and Availability, D.A.H. Jacobs, Editor, Academic Press, 1978.

[33] E.J.L. Rossinger and R.S. Dixon, Mathematical Modelling of Water Radiolysis, A Discussion of Methods, Atomic Energy of Canada Limited report AECL-5958, November 1977.

[34] A.I. Johnson and J.R. Barney, Numerical Solution of Large Systems of Stiff Ordinary Differential Equations in a Modular Simulation Framework, in Numerical Methods for Differential Systems, L. Lapidus and W.E. Schiesser, Editors, Academic, 1975.

TABLE 1
Summary of Integration Algorithms Tested

Name		Author	Comment
DESUB		Fox[23]	Bulirsh-Stoer rational extrapolation.
DIFSUB		Gear[1]	Original Gear stiff equation algorithm with matrix inversion.
EPISODE	*	Byrne and Hindmarsh[3]	Fully dynamic step size extension of GEAR.
GEAR	*	Hindmarsh[2]	Extension of DIFSUB with linear equation solution.
ODE	*	Shampine[4]	Sandia Adams predictor corrector.
RK45	*	Shampine[4]	Sandia Runge-Kutta-Fehlberg.
RKFINT	*	Liu[7]	CRNL Runge-Kutta-Fehlberg.
SODE	*	Krogh[22]	Experimental all inclusive Adams algorithm.
GEARZ	*	Carver (Hindmarsh, Curtis, Reid)[19]	Extension of GEAR with automatic sparse matrix option and revised error control base.
WES-1 WES-3 WES-12		Schiesser[27]	Runge-Kutta-Merson.

TABLE 2
Initial Test Profile

Case	N	Equations	Initial Value	Solution	Final Time	Comments
1.	1	$\dot{y}=y$	1.	e^{t}	10.	Positive eigenvalue
2.	2	$\dot{y}=-y$	1.	e^{-t}	10.	Negative eigenvalue
3.	2	$\dot{y}_1 = -y_2$	1.	$\cos.t - \sin.t$	10.	Complex eigenvalues
		$\dot{y}_2 = y_1$	1.	$\cos.t + \sin.t$		
4.	2	$\dot{y}_1 = 998y_1 + 1998y_2$	1.	$2e^{-t} - e^{-1000t}$	2.	Moderately stiff, coupled
		$\dot{y}_2 = -999y_1 - 1999y_2$	1.	$-e^{-t} + e^{-1000t}$		
5.	1	$\dot{y} = y^2 \quad \emptyset > 0$	0.1	$y(19.) = 2.$	19.	Mild discontinuity
		$y = 0 \quad \emptyset \leq 0$				
		$\emptyset = \sin(4\pi t/19.)$				
6.	1	$\dot{y} = (1-y)t + (1-t)e^{-t}$	0	$1+e^{-t^2/2} - e^{-t}$	15.	Prone to becoming unstable
7.	2	$\dot{y}_1 = -0.1y_1$	1.	$e^{-.1t}$	1.	Moderately stiff, not coupled
		$\dot{y}_2 = -100y_2$	1.	e^{-100t}		
8.	3	$\dot{y}_1 = 4.5(y_2-y_3)-5.5y_1$	2.	$e^{-t} + e^{-10t}$	2.	Mildly stiff, coupled
		$\dot{y}_2 = 49.5(y_1-y_3)-50.5y_2$	2.	$e^{-t} + e^{-100t}$		
		$\dot{y}_3 = 45.(y_1-y_2)-55.y_3$	2.	$e^{-10t} + e^{-100t}$		

TABLE 3
Further Test Profile

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
9.	1	$\dot{y}=5(y-t^2)$	0.08	$Ae^{5t}+t^2+.4t+.08$	5	A=0 is the true solution but the hidden exponential can cause instability.	[22]
10.	2	$\dot{y}_1=-y_1(t-3)^2$	e^9	$e^{-(t-3)^2/3}$	6	Variable stiffness stiff at t=0 or 6, equal eigenvalues at t=3.	[24]
		$\dot{y}_2=y_2((t-3)^2-9)$	e^{-9}	$e^{+((t-3)^2/3-9t)}$			
11.	4	$\dot{y}_1=y_2$	0	sin t	10	Mildy stiff set with considerable range in variable magnitudes.	[24]
		$\dot{y}_2=-y_1$	1	cos t			
		$\dot{y}_3=-ty_3/(y_3+1)$	1	$(t+1)e^{-t}$			
		$\dot{y}_4=y_4+(1-t)$	1	$t+e^t$			
		$\dot{y}_5=1/(1+t)$	1	ln(t+1)			
12.	10	$\dot{y}_i=-i^5y_i+f_i$	1	$e^{-i^5t}+g_it$	1.0	Heat equation with discontinuous source term.	[24]
		$f_i=5.\ t\geq.707, 4\leq i\leq 5$		$g_i=5(1-e^{-i^5(t-.707)})$			
		$f_i=10.\ t\geq.866, 8\leq i\leq 9$		$g_i=10(1-e^{-i^5(t-.866)})$			
		$f_i=0$ otherwise		$g_i=0$			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
13.	15	$\dot{y}_i=(n-1)^2(y_{i+1}-2y_i+y_{i-1})$	$\rightarrow$	$\sum_{j=1}^{3} \frac{1}{\alpha_j} e^{-\omega^2_{\alpha_j} t} \sin(\theta\alpha_j(i-1))$		Heat equation by Method of lines with central difference formula.	[24]
		$\dot{y}_1=\dot{y}_n=0$		$\alpha_1=1,\ \alpha_2=1+\frac{n-1}{2},\ \alpha_3=n-2$			
		$i=1,15$		$\omega_{\alpha_j} = 2(n-1)\sin(\pi\alpha_j/(2n-2))$			
				α_j are integer, $\theta=\pi/(n-1)$.			
14.	25	As 13, $i=1,25$					
		As 13, $i=1,35$					
16.	14	$\dot{y}_{2i}=(n-1)^2(y_{2i-3}-2y_{2i-1}+y_{2i+1})$	$\rightarrow$	$-\sum_{j=1}^{3} \frac{\omega_{\alpha_j}}{\alpha_j} \sin(\omega_{\alpha_j} t)*\sin(\theta\alpha_j(i-1))$		Wave equation solved as 13.	[24]
		$\dot{y}_{2i-1}=y_{2i},\ i=1,7$		$\sum_{j=1}^{3} \frac{1}{\alpha_j} \cos(\omega_{\alpha_j} t)*\sin(\theta\alpha_j(i-1))$			
				ω,α,θ as 13.			
17.	26	As 16, $i=1,13$					
18.	38	As 16, $i=1,19$					

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
19.	3	$\dot{y} = U[BUY+D]$		$y=UZ$	3	Rapid oscillation midly stiff	[24]
		$U = \frac{1}{3}\begin{bmatrix} -1 & 2 & 2 \\ 2 & -1 & 2 \\ 2 & 2 & -1 \end{bmatrix}$ $D = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$	-1/2 2/3	$Z=e^{-100t}\cos 1000t$ $e^{-100t}\sin 1000t$			
		$B = \begin{bmatrix} -100 & 1000 & 0 \\ -1000 & -100 & 0 \\ 0 & 0 & .1 \end{bmatrix}$	2/3	$1-e^{.1t}$			
20.	3	$\dot{y}_1=-.013y_2-1000y_1y_2$ $-2500y_1y_3$	0	-1.893×10^{-6} $.5976$ 1.402	50	Stiff, nonlinear real eigen-values.	[21c]
		$\dot{y}_2=-.013y_1-1000y_1y_3$	1				
		$\dot{y}_3=-2500y_1y_3$	1				
21.	4	$\dot{y}_1=-r_1-r_2,\ r_1=7.89x10^{-10}$	$1.76x10^{-3}$	$1.6808x10^{-2}$	1000	Chemical pyrolysis	[21c]
		$\dot{y}_2=r_1-r_3,\ r_2=1.10x10^7y_1y_3$	0	$1.3822x10^{-9}$		stiff, nonlinear	
		$\dot{y}_3=r_1-r_2,\ r_3=1.12x10^9y_2y_3$ $-r_3+r_4,\ r_4=1.13x10^3y_4$	0	$8.2516x10^{-11}$		complex eigenvalues	
		$\dot{y}_4=r_2-r_4$	0	$1.2300x10^{-9}$			
22.	4	$\dot{y}_1=-y_1+10y_2$	1	$e^{-t}(\cos10t+\sin10t)$	1		[24]
		$\dot{y}_2=-10y_1-y_2$	1	$e^{-t}(\cos10t-\sin10t)$			
		$\dot{y}_3=-100(y_3-y_4)$	1	$e^{-100t}(\cos100t+\sin100t)$			
		$\dot{y}_4=-100(y_3+y_4)$	1	$e^{-100t}(\cos100t-\sin100t)$			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
23.	5	$\dot{y}_1=r_1+r_2+r_3+r_4+r_5+r_6+r_{13}$	0	$.67\text{x}10^{-14}$	10.	Radiolysis of air, stiff, real eigenvalues nonlinear.	[9]
		$\dot{y}_2=r_8+r_9+r_{10}-r_1-r_3-r_4-2r_7+r_{14}$	0	$.94\text{x}10^{-13}$			
		$\dot{y}_3=-r_1-r_2-2r_5-r_9+r_3+r_{10}+r_{11}+r_{15}$	0	.10779			
		$\dot{y}_4=r_4+r_5+r_6+r_7+r_9-r_{10}-r_{11}$	0	.02853			
		$\dot{y}_5=r_2+r_{11}-r_8$	0	$.45\text{x}10^{-14}$			
		$r_1=-6\text{x}10^4zy_1$, $r_2=-3.4\text{x}10^8y_1y_5$, $r_3=-1.32\text{x}10^{10}y_1y_3$, $r_4=-4.62\text{x}10^9y_1y_4$					
		$r_5=-3.54\text{x}10^9y_1y_4$, $r_6=-1.08\text{x}10^9y_1y_4$, $r_7=-1.38\text{x}10^9y_1y_4$, $r_8=-6.00\text{x}10^6zy_2$					
		$r_9=-4.80\text{x}10^9y_2y_4$, $r_{10}=-1.50\text{x}10^9y_2y_3$, $r_{11}=-1.00\text{x}10^{-7}y_3y_5$, $r_{12}=0$					
		$r_{13}=2.8R$, $r_{14}=4.8R$, $r_{15}=r_{13}$					
		$R=D/V$, $V=6.02\text{x}10^{25}$, $D=2.5\text{x}10^{17}$, $z=.008$					
24.	11	Eleven equations describing radiolysis of water, written as above; they use 39 r_i reactions, so the equations should be taken from reference [33].					
25.	3	$\dot{y}_1=-r_1+r_2$, $r_1=.04y_1$	1	0	10^{10}	Robertson reaction.	[21c]
		$\dot{y}_2=r_1-r_2-r_3$, $r_2=10^4y_1y_2$	0	0			
		$\dot{y}_3=r_3$, $r_3=3\text{x}10^7y_2$	0	1			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
26.	1	$\dot{y}=\frac{2}{3}(t-1)^{-1/3} t\neq 1$	-1	$(t-1)^{2/3}$	2	Singularity at t=1	[22]
		$\dot{y}=0$ t=1					
27.	4	$\dot{y}_1=y_3$	1.2	1.2	z_2	Three body problem	[22]
		$\dot{y}_2=y_4$	0	0			
		$\dot{y}_3=2y_2+y_1+v(y_1+u)/r_1^3$ $-v(y_1-v)/r_2^3$	0	0			
		$\dot{y}_4=-2y_1+y_2-v(y_2/r_1^3)$ $-uy_2/r_2^3$	z_1	z_1			
		$z_1=-1.049357509830319$, $z_2=6.192169331319639$					
		$r_1=((y_1+u)^2+y_2^2)^{1/2}$, $r_2=((y_1-v)^2+y_2^2)^{1/2}$, u=1/82.45, v=1-u					
28.	1	$\dot{y}=h-b(y-h)$	10^{-27}	y=h			[24]
		h=(d+af)/b					
		$f=e^{-cw/\sin wt}$, $\sin wt \geq 0$ else f=0. $b=10^8$, $d=10^{-19}$, $a=10^{-18}$, c=4, $w=\pi/43200$					
29.	3	$\dot{y}_1=-y_1$	1	4.5397×10^{-5}	10	Simple nonlinear reaction	[34]
		$\dot{y}_2=y_1-y_2^2$	0	0.11079			
		$\dot{y}_3=y_2^2$	0	0.89916			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	(m)	Comments	Source
30.	6	$\dot{y}_1=(-(a+b(m_1+cy_1))y_1$	-.03424992	.031111		.73576500	Gas absorber	[34]
		$+b(m_2+cy_2)y_2/z_1+ev_1/z_1$	-.06192031	.052333		.74875687		
			-.08368619	.062525	5.0	.75929635		
		$\dot{y}_i=(ay_{i-1}-(a+b(m_i+cy_i))y_i$	-.10042889	.060285		.76774008		
		$+b(m_{i+1}+cy_{i+1})y_{i+1})/z_i$	-.046113	.046113		.77443837		
			-.12243691	.023789		.77921100		
		$\dot{y}_6=ay_5-(a+b(m_6+cy_6)y_6)/z_6$				.78383672		
		$+b(m_7+cv_2)v_2/z_6$						
31.	2	$\dot{y}_1=-(1+e^a)y_1-.5(e^a-1)$	-.1111889	.064952	10.0		Limit cycle chemical reactor	[24]
		$\dot{y}_2=e^a y_1-8.9y_2^2$ -	+.0323358	-.035880				
		$+.5(e^a-1)-y_2$						
		$a=25y_2/(y_2+2)$						

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
32.	6	$\dot{y}_1=r_1-r_2+r_5-r_{10}-r_3$	100.	49658	1000.	Cesium flare reaction with third order nonlinear terms and large range of variable magnitudes.	[9]
		$\dot{y}_2=r_1+r_3-r_4+r_{10}$	520.	25914			
		$\dot{y}_3=-r_2-r_4+r_5$	620.	75572			
		$\dot{y}_4=r_2+r_4-r_5-r_6-r_7-r_8-r_9$	10^{12}	1532			
		$\dot{y}_5=r_7+r_8+r_9+r_6$	0	10^{12}			
		$\dot{y}_6=r_1-r_3+r_4-r_6-r_7-r_8-r_9-r_{10}$	3.6×10^{14}	3.59×10^{14}			
		$r_1=.4y_2,\ r_2=10^{-12}y_1y_3,\ r_3=1.4\times10^{-16}y_1y_6,\ r_4=5\times10^{-8}y_2y_3,\ r_5=.00324y_4$					
		$r_6=10^{-31}y_6y_4^2,\ r_7=10^{-31}y_4y_5y_6,\ r_8=1.4\times10^{-16}y_4y_6,\ r_9=10^{-31}y_4y_6^2,\ r_{10}=1.24\times10^{-30}y_1y_6^2$					
33.	3	$\dot{y}_1=77.27(y_2-y_1y_2-\alpha y_1^2+y_1)$	4	$y(t+\tau)=y(t+n\tau)$	950.	Belousov limit cycle reaction variable stiffness $\tau=302.9$ $\alpha=8.375\times10^{-6}$	[21c]
		$\dot{y}_2=(-y_2-y_1y_2+y_3)/77.27$	1.1				
		$\dot{y}_3=.161(y_1-y_3)$	4				
34.	3	$\dot{y}_1=-.04y_1+.01y_2y_3$	1	0	10^{10}	Scaled Robertson equations.	[21c]
		$\dot{y}_2 = 100(4y_1-y_2y_3-30y_2^2)$	0	0			
		$\dot{y}_3=30y_2^2$	0	100			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
35.	4	$\dot{y}_1=y_3-100y_1y_2$	1	.63976	20.	Chemical kinetic reaction, attaining steady state.	[21c]
		$\dot{y}_2=y_3+2y_4-100y_1y_2-2\alpha y_2^2$	1	$.56308\times10^{-2}$			
		$\dot{y}_3=-y_3+100y_1y_2$	0	.36024			
		$\dot{y}_4=-y_4+\alpha y_2^2 \quad \alpha=10^4$	0	.31706			
36.	2	$\dot{y}_1=.01-(1+(y_1+1000)(y_1+1)*$ $(.01+y_1+y_2)$	0	-.99161	100.	Reactor kinetics.	[21c]
		$\dot{y}_2=.01-(1+y_3^2)(.01+y_1+y_2)$	0	.98329			
37.	4	$\dot{y}_1=1.3(y_3-y_1)+10400ky_2$	761	1211.2	1000.	Catalytic fluidized bed, attaining steady state.	[21c]
		$\dot{y}_2=1880(y_4-y_2(1+k))$	0	1.11×10^{-11}			
		$\dot{y}_3=1752-269y_3+267y_1$	600	1208.7			
		$\dot{y}_4=.1+320y_2-321y_4$	01	3.1153×10^{-3}			
		$k= {}^{b}, \quad b=20.7-1500/y_1$					
38.	2	$\dot{y}_1=-y_1-y_1y_2+294y_2$	1	.39127	240.	Decomposition of Ozone, attaining steady state.	[21c]
		$\dot{y}_2=y_1(1-y_2)/98-3y_2$	0	1.220×10^{-3}			

TABLE 3 (Continued)

Case	N	Equations	Initial Values	Solution	Final Time	Comments	Source
39.	3	$\dot{y}_1=.2(y_1-y_2)$	0	22.243	400	Reactor kinetics	[21c]
		$\dot{y}_2=10y_1-(60-.125y_3)y_2+y_3/8$	0	27.112			
		$\dot{y}_3=1$	0	400			
40.	4	$\dot{y}_1=10^{11}(-3y_1y_2+.0012y_4-9y_1y_3)$	$3.365\text{x}10^{-7}$	$1.714\text{x}10^{-7}$	100.	Enzyme kinetics, attaining steady state.	[21c]
		$\dot{y}_2=-3\text{x}10^{11}y_1y_2+2\text{x}10^{7}y_4$	$8.261\text{x}10^{-3}$	$3.714\text{x}10^{-3}$			
		$\dot{y}_3=10^{11}(-9y_1y_3+.001y_4)$	$1.642\text{x}10^{-3}$	$6.189\text{x}10^{-2}$			
		$\dot{y}_4=10^{11}(3y_1y_2-.0012y_9+9y_1y_3)$	$9.380\text{x}10^{-6}$	$9.545\text{x}10^{-6}$			
41.	242	Two Group Neutron Kinetics Equations ANS Benchmark described fully in the references quoted.					[19,30]
42.	50	Photochemical Smog Model comprising 80 reactions described fully in the references quoted.					[31]

TABLE 4

Overall Results for 25 Non-stiff Equation Sets (1-19, 26-31)

Routine	Option	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	26	27	28	29	30	31	Failures
EPISODE	(22)*		e										t	e	e	e			e			e		e			8
GEAR	(23)*		e								e			e	e					e				e			6
	(22)*		e								e			e						e							6
GEARZ	11									e			t														2
	12																										0
	13																			e							1
	14																										0
	15																										0
	21									e			t														2
	22																										0
	23																			e		e					2
	24																										0
	25																										0
ODE										e			f														2
RK45										f			f														2
RKFINT										f			t														2
SODE		e								e	e				e												4

e denotes unacceptable error.
t denotes time limit.
f denotes failure reported.
*(22) denotes equivalent to GEARZ option 22, full matrix chord method.

TABLE 5
Overall Results for 15 Stiff Equation Sets 20-25, 32-40

Routine	Option	20	21	22	23	24	25	32	33	34	35	36	37	38	39	40	Failures Stiff Set	Failures Both Sets
EPISODE	(22)*						f										1	9
GEAR	(23)*		e		f	t	f	f				e	e	e		t	9	15
					f	t											2	6
GEARZ	11	t	t	t	t	t	t	f		t		t	t			t	11	13
	12					t				t			t				3	3
	13		e		t	t	t	f		e			e	e		t	9	10
	14					t							t			t	2	3
	15					f							t				2	2
	21	t	t	t	t	t	t			t		t	t			t	10	12
	22																0	0
	23		e		t	t	t	f		e			e			t	8	10
	24					t										t	2	2
	25					f											1	1
ODE		f	f	f	f	f	f	f		f	f	f	f		f	f	13	15
RK45		f	f	f	f	f	f	f		f	f	f	f		f	f	13	15
RKFINT		t	t	t	t	t	t			t		t	t			t	10	12
SODE		f	f	t	f	f	f			f		t	f			f	10	14

e denotes unacceptable error.
t denotes time limit.
f denotes failure reported.
*(22) denotes equivalent to GEARZ option (22) full matrix chord method.

TABLE 6
Performance on Stiff Problem of Increasing Size
Tests 13-15 Diffusion by Method of Lines

Number of Function Evaluations, Average Error

Number of Equations			15	25	35
GEARZ	Functional	(21)	3021,10^{-4}	8848,10^{-5}	13040,10^{-5}
	Diagonal	(23)	1904,10^{-4}	5385,10^{-4}	10761,10^{-4}
	Banded	(24)	446,10^{-5}	679,10^{-5}	718,10^{-5}
	Full	(22)	446,10^{-5}	679,10^{-5}	718,10^{-5}
	Sparse	(25)	320,10^{-5}	398,10^{-5}	340,10^{-5}
EPISODE	Full	(22)	537,10^{-2}	796,10^{-2}	1094,10^{-1}
GEAR	Diagonal	(23)	1416,10^{-2}	3765,10^{-1}	6868,10^{-1}
	Full	(22)	375,10^{-3}	572,10^{-4}	705,10^{-4}

TABLE 7
Performance of GEARZ Algorithm with Neutron Kinetics Equations
Number of Function Calls NF, and Number of Time Steps NS
Required to Complete the Given Problem Times to .01% Accuracy

Method / Time		Functional Iteration	Diagonal Matrix	Banded Matrix	Full Matrix	Sparse Matrix
10^{-5}	NF	3064	749	3453	2491	131
	NS	863	143	38	55	55
10^{-3}	NF	∿30000	12884	8916	4731	247
	NS	∿3000	1725	112	106	106
10^{-1}	NF	-	-	57434	5737	309
	NS	-	-	406	142	142
1.	NF	-	-	-	16129	1034
	NS	-	-	-	209	209
4.	NF	-	-	-	-	1947*
	NS	-	-	-	-	263

*Computing time 49.5 seconds on CDC CYBER 175 Computer.

TABLE 8
Performance of Sparse Matrix Integrator on Photochemical Smog Model

Jacobian Option	Computing Times to 200 s Problem Time (seconds CDC CYBER 175)			
Matrix Type	Numeric	Analytic	Numeric	Analytic
Evaluation	Full	Full	Sparse	Sparse
Relative Error Requested				
10^{-4}	45	42	33	19
10^{-5}	55	50	41	23
10^{-6}	67	54	51	30

TABLE 9
Integration Package Storage

Package	Number of Subroutines	Approximately Total Words Program Storage	Working Storage
RKFINT	1	650	8N
RK45	4	1150	6N
ODE	5	1700	21N
GEAR	5	2650	$18N+N^2max$
EPISODE	8	2700	$18N+N^2max$
GEARZ	11	5300	$11N+N^2max$
SODE	19	5500	14N

TABLE 10
Working Storage Requirements for GEARZ

Option	Storage
Functional Iteration	10N
Diagonal Jacobian	11N
Banded Jacobian	(12+2MU+ML)N
Full Jacobian	$11N+N^2$
Sparse Jacobian	(27N+4NZ)N

MU width of upper band
ML width of lower band
NZ maximum non-zero entries per row.

APPENDIX 1
RESULT SUMMARY FOR ALL TESTS

The following tables summarize results for each test problem of tables 2 and 3 under the following headings:

ROUTINE	Routine referred to in Table 1.
OPTION	Jacobian option as in Section 5.2.
NFE	Number of function evaluations required to complete problem.
STEPS	Number of time steps required to complete problem.
TIME	Execution time required to complete problem on CYBER 175.
H	Integration time step at end of problem.
ERROR	Relative error between computed and desired values at end of problem.
FLAG	0 denotes no error reported. +1 denotes error reported (routine detects inability to complete problem). -1 numeric overflow usually due to instability. -2 routine fails to complete within reasonable time.

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 1
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	298	181	.08	.288E+00	.426E-04	0
GEARZ	13	108	54	.06	.250E+00	.115E-03	0
GEARZ	23	129	79	.06	.168E+00	.165E-03	0
EPISODE	22	148	92	.09	.126E+00	.436E-03	0
RKFSAN	0	197	124	.06	.362E+00	.291E-04	0
ODESAN	0	89	39	.06	.364E+00	.251E-04	0
GEAR	3	106	63	.06	.250E+00	.113E-03	0
GEAR	6	127	88	.07	.168E+00	.164E-03	0
DIFSUB	2	215	76	.08	.209E+00	.489E-04	0
SODE	0	52	27	.05	.221E+00	.131E-02	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 2
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	34Ø	2Ø3	.Ø9	.24ØE+ØØ	.66ØE-Ø4	Ø
GEARZ	13	114	58	.Ø6	.229E+ØØ	.128E-Ø3	Ø
GEARZ	23	142	85	.Ø6	.155E+ØØ	.1Ø1E-Ø3	Ø
EPISODE	22	139	81	.Ø8	.147E+ØØ	.113E-Ø2	Ø
RKFSAN	Ø	251	16Ø	.Ø7	.331E+ØØ	.291E-Ø4	Ø
ODESAN	Ø	1Ø4	45	.Ø6	.32ØE+ØØ	.685E-Ø4	Ø
GEAR	3	73	49	.Ø5	.9ØØE+ØØ	.63ØE-Ø1	Ø
GEAR	6	1Ø4	66	.Ø6	.492E+ØØ	.446E-Ø1	Ø
DIFSUB	2	151	55	.Ø7	.916E+ØØ	.281E-Ø1	Ø
SODE	Ø	86	46	.Ø6	.321E+ØØ	.253E-Ø4	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 3
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	616	255	.15	.219E+ØØ	-.424E-Ø4	Ø
GEARZ	11	138	78	.Ø7	.172E+ØØ	-.113E-Ø3	Ø
GEARZ	12	137	69	.Ø8	.214E+ØØ	-.657E-Ø4	Ø
GEARZ	13	167	82	.Ø8	.222E+ØØ	-.168E-Ø3	Ø
GEARZ	14	137	69	.Ø7	.214E+ØØ	-.657E-Ø4	Ø
GEARZ	15	139	69	.Ø8	.214E+ØØ	-.657E-Ø4	Ø
GEARZ	21	212	1Ø5	.Ø9	.157E+ØØ	-.126E-Ø3	Ø
GEARZ	22	246	1Ø7	.1Ø	.138E+ØØ	-.872E-Ø4	Ø
GEARZ	23	261	1Ø9	.1Ø	.138E+ØØ	-.8Ø6E-Ø4	Ø
GEARZ	24	246	1Ø7	.1Ø	.138E+ØØ	-.872E-Ø4	Ø
GEARZ	25	252	1Ø7	.13	.138E+ØØ	-.872E-Ø4	Ø
EPISODE	22	289	82	.13	.527E-Ø1	-.252E-Ø3	Ø
RKFSAN	Ø	294	171	.Ø8	.278E+ØØ	-.177E-Ø4	Ø
ODESAN	Ø	1Ø4	45	.Ø7	.271E+ØØ	-.255E-Ø4	Ø
GEAR	3	121	66	.Ø6	.235E+ØØ	-.379E-Ø3	Ø
GEAR	5	142	87	.Ø8	.174E+ØØ	-.348E-Ø3	Ø
DIFSUB	2	283	86	.Ø9	.592E-Ø1	-.325E-Ø4	Ø
SODE	Ø	96	51	.Ø7	.254E+ØØ	-.3Ø4E-Ø5	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 4
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	5098	2933	.98	.296E-02	-.402E-08	0
GEARZ	11	4609	1485	1.18	.101E-02	-.480E-07	0
GEARZ	12	285	94	.12	.136E+00	.589E-06	0
GEARZ	13	711	157	.21	.428E-01	-.106E-06	0
GEARZ	14	285	94	.12	.136E+00	.589E-06	0
GEARZ	15	293	94	.15	.136E+00	.590E-06	0
GEARZ	21	4578	1373	1.14	.151E-02	.431E-07	0
GEARZ	22	252	100	.11	.139E+00	-.404E-10	0
GEARZ	23	840	166	.24	.788E-02	-.201E-07	0
GEARZ	24	252	100	.11	.139E+00	-.404E-10	0
GEARZ	25	258	100	.14	.139E+00	-.404E-10	0
EPISODE	22	214	89	.11	.195E+00	.355E-09	0
RKFSAN	0	3435	2114	.66	.717E-02	-.166E-08	0
ODESAN	0	2801	1173	.92	.300E-02	-.892E-05	0
GEAR	3	420	163	.14	.986E-01	.266E-04	0
GEAR	5	181	107	.09	.151E+00	.459E-13	0
DIFSUB	2	20	1	.01	.100E-11	-.250E+00	1
SODE	0	4801	2348	1.55	.651E-03	.893E-07	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 5
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	2854	148	.53	.1ØØE+Ø1	.152E-Ø3	Ø
GEARZ	13	465	69	.15	.979E-Ø5	.17ØE-Ø2	Ø
GEARZ	23	515	82	.17	.975E-Ø5	.282E-Ø2	Ø
EPISODE	22	481	74	.2Ø	.837E-Ø5	.387E-Ø2	Ø
RKFSAN	Ø	48Ø	78	.13	.95ØE+Ø1	.844E-Ø2	Ø
ODESAN	Ø	384	29	.15	.816E-Ø5	.363E-Ø3	Ø
GEAR	3	337	69	.11	.1ØØE+Ø1	.18ØE-Ø2	Ø
GEAR	6	362	95	.12	.1ØØE+Ø1	.243E-Ø2	Ø
DIFSUB	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-1
SODE	Ø	247	24	.19	.99ØE+Ø1	.921E-Ø3	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 6
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	46Ø	26Ø	.11	.183E+ØØ	.1ØØE-Ø5	Ø
GEARZ	13	156	69	.Ø7	.547E+ØØ	.224E-Ø5	Ø
GEARZ	23	161	7Ø	.Ø8	.1ØØE+Ø1	.321E-Ø6	Ø
EPISODE	22	119	52	.Ø7	.122E+Ø1	.193E-Ø5	Ø
RKFSAN	Ø	384	225	.1Ø	.338E+ØØ	.197E-Ø6	Ø
ODESAN	Ø	328	146	.12	.163E+ØØ	.119E-Ø4	Ø
GEAR	3	136	75	.Ø6	.1ØØE+Ø1	.989E-Ø7	Ø
GEAR	6	116	76	.Ø5	.757E+ØØ	.274E-Ø4	Ø
DIFSUB	2	516	1Ø3	.15	.168E+ØØ	.265E-Ø5	Ø
SODE	Ø	156	79	.Ø8	.249E+ØØ	.584E-Ø8	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 7
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	241Ø	157Ø	.97	.3Ø5E-Ø1	.589E-14	Ø
GEARZ	11	52Ø	268	.4Ø	.825E-Ø2	.842E-Ø5	Ø
GEARZ	12	24Ø	169	.29	.173E+ØØ	.825E-Ø8	Ø
GEARZ	13	228	169	.26	.173E+ØØ	.832E-Ø8	Ø
GEARZ	14	24Ø	169	.28	.173E+ØØ	.825E-Ø8	Ø
GEARZ	15	232	169	.31	.173E+ØØ	.825E-Ø8	Ø
GEARZ	21	426	275	.39	.332E-Ø1	.879E-Ø5	Ø
GEARZ	22	298	226	.35	.184E+ØØ	.189E-Ø5	Ø
GEARZ	23	287	226	.3Ø	.184E+ØØ	.189E-Ø5	Ø
GEARZ	24	298	226	.37	.184E+ØØ	.189E-Ø5	Ø
GEARZ	25	291	226	.4Ø	.184E+ØØ	.189E-Ø5	Ø
EPISODE	22	711	571	1.Ø3	.179E-Ø2	.589E-14	Ø
RKFSAN	Ø	1245	772	.49	.696E-Ø1	.982E-14	Ø
ODESAN	Ø	538	256	.52	.397E-Ø1	.41ØE-11	Ø
GEAR	3	11Ø	75	.17	.956E-Ø1	.317E-Ø5	Ø
GEAR	5	141	87	.19	.1Ø6E+ØØ	.559E-Ø8	Ø
DIFSUB	2	476	82	.34	.514E-Ø1	.634E-Ø5	Ø
SODE	Ø	28Ø	134	.35	.676E+ØØ	.634E-12	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 8
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	12Ø4	784	.27	.137E-Ø1	.399E-Ø5	Ø
GEARZ	11	577	29Ø	.19	.941E-Ø2	.136E-Ø3	Ø
GEARZ	12	3Ø3	165	.14	.161E-Ø1	.476E-Ø4	Ø
GEARZ	13	584	244	.19	.77ØE-Ø2	.554E-Ø3	Ø
GEARZ	14	562	2Ø9	.22	.437E-Ø2	.45ØE-Ø4	Ø
GEARZ	15	313	172	.17	.157E-Ø1	.444E-Ø4	Ø
GEARZ	21	6Ø8	318	.21	.293E-Ø2	.333E-Ø4	Ø
GEARZ	22	338	2Ø2	.16	.143E-Ø1	.824E-Ø4	Ø
GEARZ	23	696	286	.23	.568E-Ø2	.117E-Ø3	Ø
GEARZ	24	513	216	.2Ø	.128E-Ø1	.1Ø8E-Ø4	Ø
GEARZ	25	344	2Ø2	.19	.143E-Ø1	.825E-Ø4	Ø
EPISODE	22	314	151	.16	.195E-Ø1	.43ØE-Ø3	Ø
RKFSAN	Ø	525	3Ø2	.13	.328E-Ø1	.211E-Ø4	Ø
ODESAN	Ø	485	218	.2Ø	.1Ø8E-Ø1	.133E-Ø4	Ø
GEAR	3	259	116	.1Ø	.3Ø4E-Ø1	.198E-Ø3	Ø
GEAR	5	184	111	.1Ø	.149E+ØØ	.313E-Ø4	Ø
DIFSUB	2	729	126	.2Ø	.152E+ØØ	.199E-Ø5	Ø
SODE	Ø	4Ø4	193	.21	.255E-Ø1	.4Ø5E-Ø7	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 9
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	416	262	.Ø9	.396E-Ø1	.4Ø9E+Ø2	1
GEARZ	11	1Ø9	4Ø	.Ø6	.437E-Ø1	.368E+ØØ	Ø
GEARZ	12	34	14	.Ø4	.1ØØE+Ø1	.45ØE-Ø7	Ø
GEARZ	13	33	17	.Ø4	.1ØØE+Ø1	.315E-Ø8	Ø
GEARZ	14	34	14	.Ø4	.1ØØE+Ø1	.45ØE-Ø7	Ø
GEARZ	15	32	14	.Ø4	.1ØØE+Ø1	.45ØE-Ø7	Ø
GEARZ	21	135	48	.Ø6	.244E-Ø1	.367E+ØØ	Ø
GEARZ	22	34	14	.Ø4	.1ØØE+Ø1	.49ØE-Ø9	Ø
GEARZ	23	35	18	.Ø4	.1ØØE+Ø1	.349E-1Ø	Ø
GEARZ	24	34	14	.Ø4	.1ØØE+Ø1	.49ØE-Ø9	Ø
GEARZ	25	32	14	.Ø4	.1ØØE+Ø1	.49ØE-Ø9	Ø
EPISODE	22	44	11	.Ø5	.5ØØE+Ø1	.636E-12	Ø
RKFSAN	Ø	248	14Ø	.Ø6	.537E-Ø1	.995E+Ø1	1
ODESAN	Ø	143	6Ø	.Ø7	.127E+ØØ	.263E-Ø2	Ø
GEAR	3	32	18	.Ø4	.1ØØE+Ø1	.315E-Ø8	Ø
GEAR	5	33	19	.Ø4	.1ØØE+Ø1	.49ØE-Ø9	Ø
DIFSUB	2	132	35	.Ø6	.148E+ØØ	.956E-Ø2	Ø
SODE	Ø	43	9	.Ø5	.1Ø4E+ØØ	.242E-Ø2	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 1Ø
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	1318	83Ø	.24	.164E-Ø1	.711E-Ø5	Ø
GEARZ	11	342	225	.13	.256E-Ø1	.831E-Ø4	Ø
GEARZ	12	288	182	.13	.261E-Ø1	.395E-Ø4	Ø
GEARZ	13	285	186	.11	.382E-Ø1	.8Ø5E-Ø4	Ø
GEARZ	14	288	182	.13	.261E-Ø1	.395E-Ø4	Ø
GEARZ	15	281	182	.13	.261E-Ø1	.395E-Ø4	Ø
GEARZ	21	331	216	.13	.231E-Ø1	.41ØE-Ø4	Ø
GEARZ	22	34Ø	218	.14	.263E-Ø1	.448E-Ø4	Ø
GEARZ	23	323	218	.14	.263E-Ø1	.45ØE-Ø4	Ø
GEARZ	24	34Ø	218	.15	.263E-Ø1	.448E-Ø4	Ø
GEARZ	25	327	218	.16	.263E-Ø1	.448E-Ø4	Ø
EPISODE	22	462	271	.23	.343E-Ø1	.6Ø7E-Ø4	Ø
RKFSAN	Ø	85Ø	554	.16	.367E-Ø1	.239E-Ø4	Ø
ODESAN	Ø	34Ø	161	.15	.495E-Ø1	.29ØE-Ø5	Ø
GEAR	3	122	75	.Ø7	.263E+ØØ	.791E-Ø1	Ø
GEAR	5	159	97	.Ø8	.491E+ØØ	.34ØE-Ø1	Ø
DIFSUB	2	11	1	.Ø1	.1ØØE-11	.454E-11	1
SODE	Ø	243	123	.12	.112E+ØØ	.16ØE-Ø1	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 11
CONTAINING 5 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	6Ø4	257	.33	.2Ø9E+ØØ	-.356E-Ø5	Ø
GEARZ	11	161	86	.24	.126E+ØØ	-.111E-Ø4	Ø
GEARZ	12	237	84	.32	.179E+ØØ	-.727E-Ø5	Ø
GEARZ	13	2Ø2	93	.26	.951E-Ø1	-.11ØE-Ø4	Ø
GEARZ	14	237	84	.32	.179E+ØØ	-.727E-Ø5	Ø
GEARZ	15	2Ø2	84	.37	.179E+ØØ	-.727E-Ø5	Ø
GEARZ	21	2Ø5	1Ø5	.28	.154E+ØØ	-.132E-Ø4	Ø
GEARZ	22	354	113	.42	.997E-Ø1	-.963E-Ø5	Ø
GEARZ	23	25Ø	1Ø8	.31	.153E+ØØ	-.931E-Ø5	Ø
GEARZ	24	354	113	.42	.997E-Ø1	-.963E-Ø5	Ø
GEARZ	25	294	113	.49	.997E-Ø1	-.963E-Ø5	Ø
EPISODE	22	197	69	.3Ø	.2Ø7E+ØØ	-.548E-Ø4	Ø
RKFSAN	Ø	321	178	.24	.299E+ØØ	-.116E-Ø5	Ø
ODESAN	Ø	217	1Ø3	.3Ø	.237E+ØØ	.196E-Ø6	Ø
GEAR	3	149	8Ø	.22	.233E+ØØ	-.727E-Ø5	Ø
GEAR	5	175	1Ø2	.28	.16ØE+ØØ	-.19ØE-Ø4	Ø
DIFSUB	2	314	1Ø1	.33	.221E+ØØ	.851E-Ø6	Ø
SODE	Ø	91	49	.27	.246E+ØØ	-.1Ø2E-Ø5	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 12
CONTAINING 10 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	12	6952	2836	9.89	.239E-03	.133E-06	0
GEARZ	13	5856	3155	6.63	.165E-01	.775E-08	0
GEARZ	14	6952	2836	8.99	.239E-03	.133E-06	0
GEARZ	15	7496	4290	11.76	.772E-02	.185E-06	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	22	2816	640	3.99	.516E-01	.989E-06	0
GEARZ	23	1410	640	2.17	.516E-01	.989E-06	0
GEARZ	24	2816	640	3.57	.516E-01	.989E-06	0
GEARZ	25	1608	640	3.85	.515E-01	.566E-06	0
EPISODE	0	0	0	0.00	0.	0.	-2
RKFSAN	0	6002	1781	3.90	.368E-04	.211E-14	1
ODESAN	0	1008	489	1.42	.724E-05	.134E-13	1
GEAR	3	964	510	1.17	.496E-01	.791E-06	0
GEAR	5	1139	230	1.55	.528E-01	.398E-05	0
DIFSUB	0	0	0	0.00	0.	0.	-1

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 13
CONTAINING 15 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	2122	1261	.65	.373E-Ø2	.71ØE-Ø7	Ø
GEARZ	11	1717	733	.81	.131E-Ø2	.917E-Ø4	Ø
GEARZ	12	717	217	.55	.658E-Ø2	.165E-Ø4	Ø
GEARZ	13	2219	755	1.Ø1	.745E-Ø3	.243E-Ø4	Ø
GEARZ	14	717	217	.45	.658E-Ø2	.165E-Ø4	Ø
GEARZ	15	478	2Ø8	.55	.361E-Ø2	.812E-Ø5	Ø
GEARZ	21	19Ø4	942	.9Ø	.85ØE-Ø3	.3Ø1E-Ø3	Ø
GEARZ	22	446	153	.38	.112E-Ø1	.563E-Ø4	Ø
GEARZ	23	3Ø21	861	1.23	.11ØE-Ø2	.1Ø2E-Ø3	Ø
GEARZ	24	446	153	.34	.112E-Ø1	.563E-Ø4	Ø
GEARZ	25	32Ø	153	.39	.112E-Ø1	.563E-Ø4	Ø
EPISODE	22	537	88	.41	.628E-Ø1	.443E-Ø1	Ø
RKFSAN	Ø	1399	786	.46	.887E-Ø2	.141E-Ø7	Ø
ODESAN	Ø	1226	535	.71	.215E-Ø2	.328E-Ø5	Ø
GEAR	3	1416	364	.59	.956E-Ø2	.54ØE-Ø1	Ø
GEAR	5	375	125	.32	.4ØØE-Ø1	.244E-Ø2	Ø
DIFSUB	2	2948	282	1.16	.138E-Ø1	.524E-Ø2	Ø
SODE	Ø	1Ø18	495	1.2Ø	.284E-Ø2	.262E-Ø3	Ø

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 14
CONTAINING 25 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	5500	3254	4.62	.140E-02	.756E-08	0
GEARZ	11	6233	3045	10.39	.147E-03	.516E-05	0
GEARZ	12	1348	383	4.02	.277E-02	.429E-05	0
GEARZ	13	6340	1438	8.20	.303E-03	.594E-03	0
GEARZ	14	1428	357	2.53	.799E-02	.452E-05	0
GEARZ	15	819	383	3.19	.389E-02	.178E-04	0
GEARZ	21	5385	2642	8.92	.689E-03	.247E-04	0
GEARZ	22	679	194	2.27	.447E-02	.135E-04	0
GEARZ	23	8848	2108	11.29	.417E-03	.122E-03	0
GEARZ	24	679	194	1.55	.448E-02	.135E-04	0
GEARZ	25	398	194	1.79	.447E-02	.135E-04	0
EPISODE	22	796	87	1.75	.860E-01	.331E-01	0
RKFSAN	0	3886	2396	3.29	.247E-02	.550E-07	0
ODESAN	0	3247	1376	6.21	.794E-03	.167E-05	0
GEAR	3	3765	802	4.73	.196E-02	.714E-01	0
GEAR	5	572	147	1.81	.376E-01	.764E-04	0
DIFSUB	2	20	1	.07	.100E-11	.549E-08	1
SODE	0	2778	1357	11.71	.791E-03	.125E-02	0

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 15
CONTAINING 35 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	10852	6279	4.27	.808E-03	.642E-09	0
GEARZ	11	12516	6119	8.38	.147E-03	.124E-05	0
GEARZ	12	2262	541	3.36	.211E-02	.706E-04	0
GEARZ	13	10313	2254	5.24	.485E-03	.311E-03	0
GEARZ	14	2566	483	1.76	.150E-02	.591E-05	0
GEARZ	15	1096	574	1.75	.173E-02	.148E-03	0
GEARZ	21	10761	5232	7.46	.171E-03	.632E-05	0
GEARZ	22	718	167	1.18	.133E-01	.712E-04	0
GEARZ	23	13040	2933	6.80	.268E-03	.148E-03	0
GEARZ	24	718	167	.69	.133E-01	.712E-04	0
GEARZ	25	340	167	.77	.133E-01	.712E-04	0
EPISODE	22	1094	84	1.17	.650E-01	.105E+00	0
RKFSAN	0	6012	3736	2.40	.809E-03	.250E-05	1
ODESAN	0	6257	2597	4.93	.926E-03	.815E-06	0
GEAR	3	6868	1446	3.50	.891E-03	.276E+00	0
GEAR	5	705	136	1.05	.471E-01	.162E-03	0
DIFSUB	2	20	1	.03	.100E-11	.102E-07	1
SODE	0	5160	2508	11.27	.555E-03	.989E-03	0

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 16
CONTAINING 14 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	868	468	.16	.158E-Ø1	-.32ØE-Ø7	Ø
GEARZ	11	173	87	.12	.188E-Ø1	-.173E-Ø5	Ø
GEARZ	12	349	9Ø	.21	.2Ø7E-Ø1	-.183E-Ø5	Ø
GEARZ	13	253	1Ø1	.15	.1Ø7E-Ø1	-.426E-Ø5	Ø
GEARZ	14	359	88	.17	.2Ø8E-Ø1	-.195E-Ø5	Ø
GEARZ	15	247	9Ø	.26	.2Ø7E-Ø1	-.183E-Ø5	Ø
GEARZ	21	257	132	.15	.135E-Ø1	-.182E-Ø5	Ø
GEARZ	22	437	129	.26	.143E-Ø1	-.237E-Ø5	Ø
GEARZ	23	244	123	.15	.115E-Ø1	-.121E-Ø5	Ø
GEARZ	24	499	131	.23	.155E-Ø1	-.297E-Ø5	Ø
GEARZ	25	3Ø5	129	.3Ø	.143E-Ø1	-.237E-Ø5	Ø
EPISODE	22	321	74	.22	.22ØE-Ø1	-.186E-Ø4	Ø
RKFSAN	Ø	328	182	.1Ø	.362E-Ø1	-.751E-Ø6	Ø
ODESAN	Ø	224	1Ø5	.15	.315E-Ø1	-.148E-Ø6	Ø
GEAR	3	158	9Ø	.12	.178E-Ø1	-.43ØE-Ø5	Ø
GEAR	5	297	111	.2Ø	.156E-Ø1	-.76ØE-Ø5	Ø
DIFSUB	2	391	1Ø9	.2Ø	.538E-Ø2	-.594E-Ø6	Ø
SODE	Ø	1ØØ	5Ø	.19	.213E-Ø1	-.177E-Ø6	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 17
CONTAINING 26 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	2524	1237	.54	.716E-02	-.875E-08	0
GEARZ	11	347	189	.30	.810E-02	.707E-07	0
GEARZ	12	1138	176	.93	.725E-02	.695E-06	0
GEARZ	13	619	224	.38	.701E-02	-.259E-05	0
GEARZ	14	825	165	.45	.854E-02	.908E-06	0
GEARZ	15	472	173	.82	.922E-02	.616E-06	0
GEARZ	21	448	243	.34	.605E-02	.101E-05	0
GEARZ	22	1774	265	1.35	.587E-02	.929E-06	0
GEARZ	23	596	282	.41	.642E-02	.366E-06	0
GEARZ	24	1306	253	.62	.609E-02	.101E-05	0
GEARZ	25	793	270	1.45	.557E-02	.908E-06	0
EPISODE	22	755	112	.66	.106E-01	.210E-04	0
RKFSAN	0	667	365	.21	.177E-01	.333E-06	0
ODESAN	0	335	152	.30	.929E-02	-.355E-07	0
GEAR	3	412	180	.28	.836E-02	.133E-05	0
GEAR	5	502	183	.58	.695E-02	.437E-05	0
DIFSUB	2	777	218	.54	.982E-02	.283E-06	0
SODE	0	193	96	.45	.102E-01	-.272E-08	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 18
CONTAINING 38 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	3982	19ØØ	1.Ø6	.322E-Ø2	-.717E-Ø7	Ø
GEARZ	11	571	282	.59	.392E-Ø2	-.16ØE-Ø5	Ø
GEARZ	12	1526	265	2.Ø6	.523E-Ø2	-.179E-Ø5	Ø
GEARZ	13	892	348	.72	.361E-Ø2	-.84ØE-Ø5	Ø
GEARZ	14	2389	259	1.Ø8	.394E-Ø2	-.188E-Ø5	Ø
GEARZ	15	68Ø	265	1.77	.499E-Ø2	-.17ØE-Ø5	Ø
GEARZ	21	715	379	.69	.357E-Ø2	-.354E-Ø5	Ø
GEARZ	22	348Ø	396	4.Ø7	.362E-Ø2	-.337E-Ø5	Ø
GEARZ	23	1292	5Ø8	.95	.17ØE-Ø2	-.462E-Ø5	Ø
GEARZ	24	3252	389	1.41	.216E-Ø2	-.254E-Ø5	Ø
GEARZ	25	1Ø1Ø	369	2.67	.246E-Ø2	-.337E-Ø5	Ø
EPISODE	22	1Ø55	147	1.31	.788E-Ø2	-.132E-Ø3	Ø
RKFSAN	Ø	95Ø	5Ø3	.34	.98ØE-Ø2	-.135E-Ø5	Ø
ODESAN	Ø	434	2Ø3	.49	.6Ø1E-Ø2	-.638E-Ø7	Ø
GEAR	3	546	231	.45	.521E-Ø2	-.38ØE-Ø5	Ø
GEAR	5	658	258	1.23	.532E-Ø2	-.128E-Ø4	Ø
DIFSUB	2	116Ø	322	1.Ø8	.628E-Ø2	-.5Ø3E-Ø6	Ø
SODE	Ø	287	142	.86	.737E-Ø2	-.899E-Ø7	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 19
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	11152	6442	4.8Ø	.4Ø5E-Ø4	.1Ø9E-Ø6	Ø
GEARZ	11	873	433	.53	.148E-Ø3	.293E-Ø3	Ø
GEARZ	12	572	318	.38	.2ØØE-Ø3	.197E-Ø3	Ø
GEARZ	13	951	385	.54	.217E-Ø3	.133E-Ø1	Ø
GEARZ	14	615	314	.4Ø	.225E-Ø3	.375E-Ø3	Ø
GEARZ	15	482	3Ø2	.36	.197E-Ø3	.2Ø8E-Ø3	Ø
GEARZ	21	763	435	.45	.148E-Ø3	.255E-Ø3	Ø
GEARZ	22	753	438	.5Ø	.148E-Ø3	.274E-Ø3	Ø
GEARZ	23	1293	534	.7Ø	.152E-Ø3	.138E-Ø2	Ø
GEARZ	24	76Ø	43Ø	.51	.148E-Ø3	.3ØØE-Ø3	Ø
GEARZ	25	69Ø	436	.51	.148E-Ø3	.267E-Ø3	Ø
EPISODE	22	93Ø	351	.63	.138E-Ø3	.753E-Ø3	Ø
RKFSAN	Ø	1583	747	.71	.3Ø6E-Ø3	.495E-Ø4	Ø
ODESAN	Ø	494	231	.32	.233E-Ø3	.1Ø9E-Ø4	Ø
GEAR	3	374	198	.23	.425E-Ø3	.154E-Ø1	Ø
GEAR	5	325	257	.24	.337E-Ø3	.592E-Ø2	Ø
DIFSUB	2	14	1	.Ø1	.1ØØE-11	-.167E-Ø9	1
SODE	Ø	433	2Ø6	.31	.244E-Ø3	.66ØE-Ø5	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 2Ø
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	12	3Ø9	122	.13	.966E+ØØ	.698E-Ø3	Ø
GEARZ	13	726	283	.21	.1ØØE+Ø1	.853E-Ø3	Ø
GEARZ	14	295	12Ø	.14	.624E+ØØ	.7ØØE-Ø3	Ø
GEARZ	15	318	122	.16	.966E+ØØ	.698E-Ø3	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	265	117	.12	.1ØØE+Ø1	.7ØØE-Ø3	Ø
GEARZ	23	332	139	.16	.346E+ØØ	.7ØØE-Ø3	Ø
GEARZ	24	267	119	.12	.1ØØE+Ø1	.7Ø1E-Ø3	Ø
GEARZ	25	274	117	.16	.1ØØE+Ø1	.7ØØE-Ø3	Ø
EPISODE	22	11Ø	27	.Ø7	.995E+Ø1	.716E-Ø3	Ø
RKFSAN	Ø	6ØØ7	3Ø24	1.Ø6	.1Ø5E-Ø2	Ø.	1
ODESAN	Ø	1Ø7Ø	448	.35	.442E-Ø3	Ø.	1
GEAR	3	584	235	.17	.777E+ØØ	.854E-Ø3	Ø
GEAR	5	218	127	.1Ø	.1ØØE+Ø1	.7ØØE-Ø3	Ø
DIFSUB	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
SODE	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 21
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	12	1243	380	.41	.225E+02	.466E-06	0
GEARZ	13	4378	1260	1.09	.424E+02	.254E+00	0
GEARZ	14	1553	642	.51	.100E+03	.236E-06	0
GEARZ	15	1338	416	.63	.224E+01	.466E-06	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	22	201	75	.09	.100E+03	.480E-06	0
GEARZ	23	2017	200	.49	.237E+02	.367E-01	0
GEARZ	24	201	75	.10	.100E+03	.522E-06	0
GEARZ	25	199	75	.13	.100E+03	.480E-06	0
EPISODE	22	121	29	.07	.231E+03	.709E-04	0
RKFSAN	0	6004	4	1.10	.180E-03	0.	1
ODESAN	0	1070	447	.35	.686E-04	0.	1
GEAR	3	841	197	.23	.460E+01	.310E+00	0
GEAR	5	236	93	.11	.627E+02	.143E-07	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 22
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	5326	3217	1.Ø4	.11ØE-Ø1	-.2Ø3E-Ø5	Ø
GEARZ	11	951	667	.36	.253E-Ø2	-.196E-Ø4	Ø
GEARZ	12	46Ø	299	.23	.927E-Ø2	-.23ØE-Ø5	Ø
GEARZ	13	94Ø	379	.3Ø	.198E-Ø1	-.235E-Ø4	Ø
GEARZ	14	46Ø	299	.23	.927E-Ø2	-.23ØE-Ø5	Ø
GEARZ	15	438	299	.25	.927E-Ø2	-.23ØE-Ø5	Ø
GEARZ	21	7Ø6	472	.28	.431E-Ø2	-.335E-Ø4	Ø
GEARZ	22	735	4Ø7	.32	.145E-Ø1	-.29ØE-Ø4	Ø
GEARZ	23	1Ø82	487	.37	.123E-Ø1	-.395E-Ø4	Ø
GEARZ	24	735	4Ø7	.33	.145E-Ø1	-.29ØE-Ø4	Ø
GEARZ	25	664	4Ø6	.38	.153E-Ø1	-.279E-Ø4	Ø
EPISODE	22	4828	1142	1.83	.11ØE-Ø2	-.254E-Ø7	Ø
RKFSAN	Ø	233Ø	13Ø4	.48	.267E-Ø1	-.163E-Ø5	Ø
ODESAN	Ø	911	424	.38	.4Ø2E-Ø2	-.166E-Ø6	Ø
GEAR	3	568	233	.2Ø	.114E-Ø1	-.463E-Ø4	Ø
GEAR	5	24Ø	143	.13	.181E-Ø1	-.858E-Ø4	Ø
DIFSUB	2	859	2Ø6	.27	.3Ø7E-Ø2	-.395E-Ø5	Ø
SODE	Ø	471	22Ø	.29	.283E-Ø1	-.471E-Ø7	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 23
CONTAINING 5 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	12	5626	21Ø8	2.11	.7ØØE+Ø4	.279E-Ø1	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	14	84Ø5	2457	2.94	.273E+Ø3	.175E-Ø1	Ø
GEARZ	15	7344	3589	3.64	.61ØE+Ø4	.2Ø6E-Ø1	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	1622	431	.64	.7ØØE+Ø4	.156E-Ø1	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	24	311Ø	584	1.07	.131E+Ø4	.169E-Ø1	Ø
GEARZ	25	162Ø	433	.97	.7ØØE+Ø4	.237E-Ø1	Ø
EPISODE	22	327	46	.14	.423E+Ø5	.978E-Ø2	Ø
RKFSAN	Ø	6ØØ6	3731	1.23	.752E-Ø4	Ø.	1
ODESAN	Ø	1Ø46	468	.37	.419E-Ø4	Ø.	1
GEAR	3	181Ø6	2838	4.91	.7ØØE+Ø4	.192E+15	1
GEAR	5	1Ø44	368	.41	.7ØØE+Ø4	.371E+Ø1	1

ODE INTEGRATION ALGORITHM TEST RESULTS
WATER CASE 24
CONTAINING 11 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	15	14	1	.Ø1	.1ØØE-Ø3	Ø.	1
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	974	247	.91	.643E+Ø1	.361E-Ø3	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	25	14	1	.Ø1	.1ØØE-Ø3	Ø.	1
EPISODE	22	41Ø	26	.38	.39ØE+Ø2	.21ØE-Ø6	Ø
RKFSAN	Ø	6ØØ4	4	3.48	.13ØE-1Ø	Ø.	1
ODESAN	Ø	1337	117	1.ØØ	.396E-11	Ø.	1
GEAR	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEAR	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
DIFSUB	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 25
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	12	49156	46631	44.38	.900E+04	.174E-05	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	14	84181	47618	64.42	.900E+04	.178E-04	0
GEARZ	15	81789	46278	69.29	.900E+04	.174E-05	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	22	45288	44826	43.68	.900E+04	.174E-05	0
GEARZ	23	16664	6718	11.36	.900E+04	.643E+29	1
GEARZ	24	45403	44851	44.22	.900E+04	.514E-05	0
GEARZ	25	45284	44845	47.85	.900E+04	.174E-05	0
EPISODE	22	1431	374	1.22	.847E+08	.526E+05	1
RKFSAN	0	6003	3439	2.19	.168E-02	.317E+00	1
ODESAN	0	1050	462	.85	.356E-03	.330E+00	1
GEAR	3	4666	1228	2.67	.238E+04	.154E+01	1
GEAR	5	45016	44743	37.50	.900E+04	.174E-05	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 26
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	0	0	0.00	0.	0.	-2
GEARZ	13	662	391	.20	.816E-01	.116E-03	0
GEARZ	23	848	548	.27	.638E-01	.120E-03	0
EPISODE	22	988	421	.43	.922E-01	.201E-03	0
RKFSAN	0	700	321	.17	.602E+00	.309E-04	0
ODESAN	0	751	278	.29	.130E+00	.425E-04	0
GEAR	3	307	141	.12	.125E+00	.384E-03	0
GEAR	6	361	195	.12	.714E-01	.505E-03	0
DIFSUB	2	355	135	.11	.880E-01	.384E-03	0
SODE	0	373	173	.15	.103E+00	.259E-04	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 27
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	2554	1234	.61	.179E-Ø1	-.191E-Ø5	Ø
GEARZ	11	6ØØ	357	.27	.135E-Ø1	-.164E-Ø4	Ø
GEARZ	12	789	356	.37	.173E-Ø1	-.2Ø4E-Ø4	Ø
GEARZ	13	983	4Ø8	.36	.158E-Ø1	-.13ØE-Ø4	Ø
GEARZ	14	899	366	.4Ø	.227E-Ø1	-.499E-Ø4	Ø
GEARZ	15	764	358	.46	.136E-Ø1	-.2Ø6E-Ø4	Ø
GEARZ	21	848	515	.37	.146E-Ø1	-.862E-Ø4	Ø
GEARZ	22	1ØØ9	495	.45	.892E-Ø2	-.725E-Ø5	Ø
GEARZ	23	1188	528	.46	.347E-Ø2	-.113E-Ø3	Ø
GEARZ	24	1179	52Ø	.51	.178E-Ø1	-.884E-Ø4	Ø
GEARZ	25	1Ø6Ø	5Ø3	.63	.136E-Ø1	-.127E-Ø4	Ø
EPISODE	22	959	342	.46	.491E-Ø1	-.134E-Ø3	Ø
RKFSAN	Ø	857	41Ø	.24	.831E-Ø1	-.141E-Ø4	Ø
ODESAN	Ø	746	344	.36	.333E-Ø1	.559E-Ø5	Ø
GEAR	3	717	339	.27	.488E-Ø1	-.111E-Ø2	Ø
GEAR	5	785	426	.36	.372E-Ø1	-.263E-Ø4	Ø
DIFSUB	2	12Ø8	368	.41	.14ØE-Ø1	-.622E-Ø5	Ø
SODE	Ø	767	385	.46	.244E-Ø1	-.358E-Ø7	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 28
CONTAINING 1 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
GEARZ	13	39	22	.Ø4	.1ØØE+Ø5	.582E-3Ø	Ø
GEARZ	23	39	22	.Ø4	.1ØØE+Ø5	.582E-3Ø	Ø
EPISODE	22	1131	289	.5Ø	.193E+Ø4	.229E-31	Ø
GEAR	3	37	31	.Ø5	.1ØØE+Ø5	.666E-26	Ø
GEAR	6	37	31	.Ø4	.1ØØE+Ø5	.666E-26	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 29
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	454	281	.12	.240E+00	.371E-02	0
GEARZ	11	170	93	.09	.217E+00	.377E-02	0
GEARZ	12	210	93	.11	.225E+00	.376E-02	0
GEARZ	13	182	92	.09	.226E+00	.376E-02	0
GEARZ	14	210	93	.11	.225E+00	.376E-02	0
GEARZ	15	201	93	.11	.225E+00	.376E-02	0
GEARZ	21	211	130	.10	.158E+00	.380E-02	0
GEARZ	22	263	130	.12	.157E+00	.378E-02	0
GEARZ	23	229	131	.10	.156E+00	.378E-02	0
GEARZ	24	263	130	.12	.157E+00	.378E-02	0
GEARZ	25	249	130	.15	.157E+00	.378E-02	0
EPISODE	22	155	57	.09	.854E+00	.251E-01	0
RKFSAN	0	325	187	.09	.331E+00	.372E-02	0
ODESAN	0	251	120	.12	.253E+00	.372E-02	0
GEAR	3	137	89	.08	.631E+00	.576E-02	0
GEAR	5	215	118	.10	.487E+00	.130E-01	0
DIFSUB	2	1051	125	.28	.763E+00	.562E-02	0
SODE	0	105	55	.09	.356E+00	.373E-02	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 30
CONTAINING 6 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	190	108	.06	.595E+00	-.412E-03	0
GEARZ	11	76	36	.06	.278E+00	-.411E-03	0
GEARZ	12	108	36	.07	.346E+00	-.413E-03	0
GEARZ	13	100	44	.06	.117E+00	-.413E-03	0
GEARZ	14	108	36	.07	.346E+00	-.413E-03	0
GEARZ	15	93	36	.08	.346E+00	-.413E-03	0
GEARZ	21	94	51	.06	.239E+00	-.414E-03	0
GEARZ	22	136	53	.08	.304E+00	-.414E-03	0
GEARZ	23	119	57	.07	.175E+00	-.413E-03	0
GEARZ	24	136	53	.08	.304E+00	-.414E-03	0
GEARZ	25	121	53	.09	.304E+00	-.414E-03	0
EPISODE	22	158	38	.08	.335E+00	-.414E-03	0
RKFSAN	0	119	72	.05	.705E+00	-.413E-03	0
ODESAN	0	75	32	.06	.369E+00	-.413E-03	0
GEAR	3	74	47	.06	.285E+00	-.412E-03	0
GEAR	5	129	58	.08	.367E+00	-.414E-03	0
DIFSUB	2	165	55	.08	.334E+00	-.413E-03	0
SODE	0	78	37	.09	.316E+00	-.413E-03	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 31
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	790	353	.19	.218E+00	.494E-05	0
GEARZ	11	321	156	.14	.116E+00	.369E-05	0
GEARZ	12	305	144	.14	.983E-01	.147E-05	0
GEARZ	13	325	167	.13	.816E-01	-.179E-04	0
GEARZ	14	305	144	.14	.983E-01	.147E-05	0
GEARZ	15	315	146	.17	.136E+00	.214E-05	0
GEARZ	21	365	204	.15	.923E-01	-.154E-04	0
GEARZ	22	338	197	.15	.851E-01	.342E-05	0
GEARZ	23	503	210	.18	.125E+00	-.135E-04	0
GEARZ	24	338	197	.16	.851E-01	.342E-05	0
GEARZ	25	344	197	.18	.851E-01	.342E-05	0
EPISODE	22	537	173	.24	.136E+00	.165E-04	0
RKFSAN	0	366	220	.11	.296E+00	-.858E-06	0
ODESAN	0	242	113	.13	.149E+00	.845E-07	0
GEAR	3	283	161	.12	.759E-01	-.250E-04	0
GEAR	5	283	171	.14	.788E-01	-.536E-04	0
DIFSUB	2	435	139	.14	.113E+00	-.297E-06	0
SODE	0	259	132	.14	.111E+00	-.289E-05	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 32
CONTAINING 6 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	32320	19598	6.09	.100E+01	.543E-05	0
GEARZ	11	13	1	.01	.100E-14	0.	1
GEARZ	12	1973	1291	.88	.100E+01	.548E-05	0
GEARZ	13	2112	1335	.75	.100E+01	.284E+06	1
GEARZ	14	2304	1366	1.01	.100E+01	.872E-01	0
GEARZ	15	2135	1361	1.29	.100E+01	.534E-03	0
GEARZ	21	26938	8858	7.45	.100E+01	.861E-05	0
GEARZ	22	1805	1141	.88	.100E+01	.533E-05	0
GEARZ	23	1998	1291	.75	.100E+01	.152E+05	1
GEARZ	24	1786	1138	.82	.100E+01	.694E-02	0
GEARZ	25	1719	1136	1.07	.100E+01	.415E-03	0
EPISODE	22	801	183	.36	.121E+03	.188E-02	0
RKFSAN	0	6007	3894	1.21	.855E-02	0.	1
ODESAN	0	1051	459	.39	.873E-02	0.	1
GEAR	3	2055	1352	.66	.100E+01	.164E+07	1
GEAR	5	1514	1113	.76	.100E+01	.530E-04	0
DIFSUB	2	29	1	.01	.100E-11	0.	1
SODE	0	7833	3676	4.68	.381E+01	.484E-03	0

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 33
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	12	17169	9464	5.37	.513E-Ø1	.2Ø6E+Ø1	1
GEARZ	13	7725	2325	2.Ø3	.141E+ØØ	.223E+ØØ	Ø
GEARZ	14	1Ø722	4648	3.33	.579E-Ø1	.2Ø5E+Ø1	1
GEARZ	15	7389	2726	3.Ø4	.527E-Ø1	.2Ø6E+Ø1	1
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	2966	1316	.98	.567E-Ø1	.2Ø6E+Ø1	1
GEARZ	23	58Ø3	2Ø17	1.5Ø	.314E-Ø1	.212E+Ø1	1
GEARZ	24	3188	1339	1.Ø8	.53ØE-Ø1	.2Ø6E+Ø1	1
GEARZ	25	3Ø9Ø	1278	1.29	.555E-Ø1	.2Ø6E+Ø1	1
EPISODE	22	3462	1114	1.27	.6Ø3E-Ø1	.2Ø2E+Ø1	1
RKFSAN	Ø	6ØØ5	3795	1.Ø2	.295E-Ø2	Ø.	1
ODESAN	Ø	1Ø6Ø	453	.35	.146E-Ø2	Ø.	1
GEAR	3	6Ø52	1Ø68	1.45	.299E+Ø1	.154E+Ø3	1
GEAR	5	1917	619	.6Ø	.126E+Ø2	.984E-Ø1	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

Note: Because this test set is a limit cycle with severe variations, the unusually large relative errors quoted are due to a slight error in frequency response. This problem is discussed further in Section 5.6.

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 34
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	13	7Ø74	2464	4.23	.4ØØE+Ø8	.64ØE-Ø1	Ø
GEARZ	14	7925	5116	6.24	.543E+Ø7	.185E-Ø4	Ø
GEARZ	15	76764	42275	65.7Ø	.197E+Ø8	.174E-Ø5	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	1233	618	1.12	.145E+Ø8	.174E-Ø5	Ø
GEARZ	23	3845	635	2.2Ø	.4ØØE+Ø8	.168E-Ø1	Ø
GEARZ	24	1722	63Ø	1.42	.112E+Ø8	.269E-Ø5	Ø
GEARZ	25	1244	616	1.45	.159E+Ø8	.174E-Ø5	Ø
EPISODE	22	748	296	.73	.171E+Ø9	.147E-Ø5	Ø
RKFSAN	Ø	6ØØ5	3838	2.21	.168E-Ø2	Ø.	1
ODESAN	Ø	1Ø59	458	.86	.782E-Ø3	Ø.	1
GEAR	3	314Ø	924	1.88	.4ØØE+Ø8	.799E-Ø1	Ø
GEAR	5	711	34Ø	.65	.4ØØE+Ø8	.188E-Ø5	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 35
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	14998	8155	2.77	.1Ø6E-Ø1	.645E-Ø5	Ø
GEARZ	11	12127	11754	3.76	.172E-Ø2	.5Ø3E-Ø5	Ø
GEARZ	12	671	248	.26	.118E+Ø1	.66ØE-Ø5	Ø
GEARZ	13	959	344	.29	.6Ø6E+ØØ	.154E-Ø3	Ø
GEARZ	14	7ØØ	332	.28	.454E+ØØ	.8Ø2E-Ø5	Ø
GEARZ	15	984	441	.47	.641E-Ø1	.599E-Ø5	Ø
GEARZ	21	11963	3481	3.16	.235E-Ø2	.659E-Ø5	Ø
GEARZ	22	58Ø	245	.23	.2ØØE+Ø1	.629E-Ø5	Ø
GEARZ	23	727	3Ø8	.25	.2ØØE+Ø1	.348E-Ø4	Ø
GEARZ	24	66Ø	257	.26	.662E+ØØ	.7Ø2E-Ø5	Ø
GEARZ	25	584	245	.33	.2ØØE+Ø1	.629E-Ø5	Ø
EPISODE	22	395	14Ø	.19	.5Ø5E+Ø1	.652E-Ø5	Ø
RKFSAN	Ø	6Ø1Ø	3782	1.23	.127E-Ø1	Ø.	1
ODESAN	Ø	1Ø54	465	.36	.587E-Ø2	Ø.	1
GEAR	3	557	24Ø	.19	.2ØØE+Ø1	.639E-Ø3	Ø
GEAR	5	39Ø	196	.18	.2ØØE+Ø1	.673E-Ø5	Ø
DIFSUB	2	32	1	.Ø1	.1ØØE-11	Ø.	1
SODE	Ø	3162	1323	1.54	.128E+Ø1	.554E-Ø5	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 36
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	12	277	88	.11	.639E+ØØ	-.184E-Ø5	Ø
GEARZ	13	455	11Ø	.15	.251E+ØØ	.12ØE-Ø3	Ø
GEARZ	14	277	88	.12	.639E+ØØ	-.184E-Ø5	Ø
GEARZ	15	287	89	.14	.647E+ØØ	-.23ØE-Ø5	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	331	1Ø6	.13	.462E+ØØ	.631E-Ø5	Ø
GEARZ	23	641	134	.18	.289E+ØØ	.785E-Ø4	Ø
GEARZ	24	331	1Ø6	.13	.462E+ØØ	.631E-Ø5	Ø
GEARZ	25	339	1Ø6	.16	.462E+ØØ	.631E-Ø5	Ø
EPISODE	22	232	63	.1Ø	.6Ø5E+ØØ	.699E-Ø6	Ø
RKFSAN	Ø	6ØØ6	3995	1.Ø3	.385E-Ø2	Ø.	1
ODESAN	Ø	1Ø64	449	.32	.221E-Ø2	Ø.	1
GEAR	3	4Ø2	133	.13	.3Ø1E+ØØ	.751E-Ø4	Ø
GEAR	5	3ØØ	114	.12	.315E+ØØ	-.42ØE-Ø5	Ø
DIFSUB	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
SODE	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 37
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	12	98	1	.Ø3	.1ØØE-15	Ø.	1
GEARZ	13	9454	926	2.39	.392E+Ø2	.259E+ØØ	Ø
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	Ø	Ø	Ø	Ø.ØØ	Ø.	Ø.	-2
GEARZ	22	638	21Ø	.26	.337E+Ø2	.3Ø9E-Ø4	Ø
GEARZ	23	3198	31Ø	.85	.265E+Ø2	.263E-Ø1	Ø
GEARZ	24	7739	1273	2.15	.824E+ØØ	.3Ø7E-Ø4	Ø
GEARZ	25	6ØØ	2Ø2	.34	.752E+Ø2	.311E-Ø4	Ø
EPISODE	22	35Ø	11Ø	.16	.843E+Ø2	.338E-Ø4	Ø
RKFSAN	Ø	6ØØ6	4	1.15	.144E-1Ø	Ø.	1
ODESAN	Ø	1Ø58	443	.36	.628E-11	Ø.	1
GEAR	3	338Ø	5Ø9	.88	.1ØØE+Ø3	.4ØØE-Ø1	Ø
GEAR	5	299	123	.14	.64ØE+Ø2	.315E-Ø4	Ø
DIFSUB	2	51	1	.Ø2	.1ØØE-11	Ø.	1

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 38
CONTAINING 2 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	2338	1387	.95	.768E+ØØ	.922E-Ø4	Ø
GEARZ	11	2159	791	1.35	.131E+ØØ	.19ØE-Ø3	Ø
GEARZ	12	277	122	.28	.3Ø6E+Ø1	.869E-Ø4	Ø
GEARZ	13	415	117	.33	.12ØE+Ø2	.141E-Ø2	Ø
GEARZ	14	277	122	.28	.3Ø6E+Ø1	.869E-Ø4	Ø
GEARZ	15	283	122	.35	.3Ø6E+Ø1	.869E-Ø4	Ø
GEARZ	21	2336	828	1.34	.1Ø8E+Ø1	.354E-Ø3	Ø
GEARZ	22	216	99	.24	.184E+Ø2	.9ØØE-Ø4	Ø
GEARZ	23	783	171	.51	.619E+ØØ	.459E-Ø3	Ø
GEARZ	24	216	99	.25	.184E+Ø2	.9ØØE-Ø4	Ø
GEARZ	25	222	99	.3Ø	.184E+Ø2	.9ØØE-Ø4	Ø
EPISODE	22	199	79	.24	.365E+Ø2	.899E-Ø4	Ø
RKFSAN	Ø	1727	1Ø54	.71	.117E+Ø1	.9Ø3E-Ø4	Ø
ODESAN	Ø	1413	596	1.14	.345E+ØØ	.9Ø9E-Ø4	Ø
GEAR	3	394	136	.31	.599E+Ø1	.14ØE-Ø2	Ø
GEAR	5	178	1Ø6	.22	.192E+Ø2	.9Ø1E-Ø4	Ø
DIFSUB	2	3444	287	1.7Ø	.168E+Ø1	.168E-Ø3	Ø
SODE	Ø	1224	575	1.Ø8	.39ØE+ØØ	.9Ø1E-Ø4	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 39
CONTAINING 3 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	Ø	323Ø2	18754	1Ø.97	.22ØE+ØØ	.277E-Ø4	Ø
GEARZ	11	3Ø53Ø	9355	17.85	.666E-Ø1	.219E-Ø4	Ø
GEARZ	12	841	367	.72	.581E+ØØ	.242E-Ø4	Ø
GEARZ	13	1188	4Ø4	.8Ø	.617E+ØØ	.615E-Ø4	Ø
GEARZ	14	846	365	.73	.743E+ØØ	.166E-Ø4	Ø
GEARZ	15	796	338	.85	.932E+ØØ	.171E-Ø4	Ø
GEARZ	21	3Ø4Ø6	9Ø73	18.Ø6	.156E+ØØ	.218E-Ø4	Ø
GEARZ	22	435	181	.43	.144E+Ø1	.884E-Ø5	Ø
GEARZ	23	1515	5Ø1	1.Ø3	.383E+ØØ	.337E-Ø4	Ø
GEARZ	24	435	181	.43	.144E+Ø1	.884E-Ø5	Ø
GEARZ	25	447	181	.56	.144E+Ø1	.884E-Ø5	Ø
EPISODE	22	316	93	.34	.169E+Ø1	.214E-Ø4	Ø
RKFSAN	Ø	6ØØ5	3956	2.21	.7Ø6E-Ø1	Ø.	1
ODESAN	Ø	1Ø55	461	.86	.158E-Ø1	Ø.	1
GEAR	3	1155	461	.77	.24ØE+ØØ	.152E-Ø3	Ø
GEAR	5	451	195	.44	.145E+Ø1	.198E-Ø4	Ø
DIFSUB	2	76931	8939	36.31	.572E+ØØ	.274E-Ø4	Ø
SODE	Ø	786Ø	3869	7.52	.128E+ØØ	.275E-Ø4	Ø

ERROR FLAGS

Ø	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

ODE INTEGRATION ALGORITHM TEST RESULTS
TEST EQUATION SET 40
CONTAINING 4 EQUATIONS

ROUTINE	OPTION	NFE	STEPS	TIME	H	ERROR	FLAG
RKFINT	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	12	792	44	.67	.100E+02	.997E-05	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	15	830	44	1.11	.100E+02	.995E-05	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	22	501	56	.47	.100E+02	.997E-05	0
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	0	0	0	0.00	0.	0.	-2
GEARZ	25	517	56	.72	.100E+02	.997E-05	0
EPISODE	22	558	174	.55	.392E+02	.998E-05	0
RKFSAN	0	6005	4	2.27	.850E-09	0.	1
ODESAN	0	1061	352	.89	.388E-09	0.	1
GEAR	0	0	0	0.00	0.	0.	-2
GEAR	5	487	60	.46	.100E+02	.997E-05	0
DIFSUB	2	48	1	.04	.100E-11	0.	1

ERROR FLAGS

0	NO ERRORS REPORTED
+1	ERROR REPORTED
-1	NUMERIC OVERFLOW
-2	TIME LIMIT

APPENDIX 2
DOCUMENTATION FOR GEARZ

The following pages from the AECL subroutine library manual [7], describe in some detail the procedures for using STIFFZ, the executive routine for GEARZ.

TITLE	Error Controlled Integration of Ordinary Differential Equations, using a Modified Gear Algorithm. A package of subroutines to integrate large, non-linear, or stiff sets of ordinary differential equations.
ENTRY	CALL STIFFZ(EQNSF,<u>Y</u>,N,<u>T</u>,DTINT,EPS,<u>DTUSED</u>,MF,<u>INOUT</u>,WS)

EQNSF	The auxiliary subroutine supplied by the user, and declared external in the routine which calls STIFFZ. Its parameter sequence is EQNSF(N,T,Y,DY) and it must specify each of the N differential equations by defining each of the derivatives DY(I) in equation 17-1.
Y	Real input/output array of dimension N. Each element must be set to the initial value $y_i(0)$ prior to the first call to STIFFZ, and will thereafter contain current values $y_i(t)$ of the dependent variables.
N	Integer input variable to be set to the number of differential equations N.
T	Real input/output variable to be set to the initial value of the independent variable, t, prior to the first call to STIFFZ and will thereafter contain the current value of t.
DTINT	Real input variable to be set to the interval in t after which STIFFZ is to return to the calling routine.
EPS	Real input variable to be set to the acceptable error tolerance required by the user, and used by STIFFZ to govern step size used, DTUSED. A step is deemed acceptable if the estimated relative local truncation error is less than EPS for each y_i. Realistic values are $1.0\,E^{-10} \leq EPS \leq 1.0E^{-2}$.
DTUSED	Real output parameter which holds the current stepsize used by STIFFZ. The user may monitor DTUSED but may not change it, except in special

cases where discontinuities are to be handled via the common blocks /DISCO/ and /STIFS/ as discussed below.

MF Two-digit input integer, specifying

(a) Method indicator, nm=|MF|, defined as follows:

If n=1, use Adam's Bashforth predictor corrector (for non-linear equation systems).

n=2, use Gear's stiff system predictor corrector (for stiff equation systems).

m=1, use explicit method, no Jacobian analysis.

m=2, use implicit method, full Jacobian analysis.

m=3, use implicit method, diagonal Jacobian approximation.

m=4, use implicit method, banded Jacobian analysis.

m=5, use implicit method, sparse Jacobian analysis.

The full and sparse Jacobian analyses are the most accurate and permit the greatest efficiency of the routine. Banded approximation is next, followed by the diagonal and explicit methods, unless the Jacobian happens to be truly banded, in which case m=3 is ideal. The choioe is normally a trade off between efficiency and storage required in the WS array.

(b) Printing Control: Negative values of MF suppress printing of messages concerning non fatal errors.

INOUT integer input/output variable, the operate flag, which must be set = 0 for the first call of STIFFZ and which returns the current order of the method when integration is successful, or a negative value when an error has occurred. (See EXIT below.)

WS(NWS) real working storage array used by STIFFZ chiefly to hold the Jacobian matrix $\partial f/\partial y$. To permit bounds to be checked, WS(1) should contain the value of NWS. The dimension requirements depend on the method indicator, nm, as follows:

m=1	NWS $\geq$ 10N+1
2	$\geq$ N(N+11)+1
3	$\geq$ 11*N+1
4	$\geq$ N(12+2ML+MU)+1
5	$\geq$ 17 + N(26+4ML)

where ML and MU are defined below.

ADDITIONAL ENTRY INFO

1) The subroutine EQNSF must be provided to define the equations(17-1) above as follows:

```
SUBROUTINE EQNSF(N,T,Y,DY)
REAL Y(N),DY(N)

DY(1)=...
DY(2)=...
   .
   .
   .
END
```

This will be called repeatedly at times T determined by STIFFZ to evaluate the derivatives DY_i in terms of Y_i and T as integration proceeds. It may call other routines as required.

2) For the first call to STIFFZ, set INOUT=0, T to T_o and the Y_i to $Y_i(o)$. STIFFZ will attempt to integrate from time T_0 to $T_1 = T_0$ + DTINT. If it is unable to do so, diagnostics will be printed if not suppressed and INOUT will be returned negative. If the step is successful, INOUT will be returned positive and equal to the current order of approximation used by the algorithm. The control program should then arrange for printout of relevant variables, make any change required to DTINT, and return to STIFFZ to integrate from T_1 to T_1 + DTINT.

The actual step size DTUSED taken by the routine must not be controlled by the user other than indirectly via DTMAX,DTMIN and EPS, but it should normally be printed out to assess the progression of the integration.

COMMON BLOCKS USED

The common blocks need not be included in the calling routine unless the user wishes to change default values in STIFFZ or more thoroughly examine the solution.

(a) COMMON/BASINT/YCUT,DYCUT,DTMAX,DTMIN,ML,MU

YCUT	Lower bound of significance of Y, used for computing the relative error.
DYCUT	Lower bound of significance of DY, used for computing the relative error. To avoid problems approaching, leaving, or crossing zero the relative error is based on EPS*AMAX1(|Y|,YCUT,DYCUT-|DY|). Defaults are YCUT=1.0E-14,DYCUT=1.0E-9, which are appropriate for Y values of the order of unity. For very small or large Y values, default values should be modified in ratio.
DTMAX DTMIN	Upper and lower bounds permitted in DTUSED, default values 10^{15},10^{-15}.
ML,MU	integer variables whose use depends on the method, nm, default values are 1. For m $\leq$ 3, ML and MU are unused.

For m=4, ML and MU are the width of the upper and lower bands excluding the diagonal.

For m=5, ML is the sparsity indicator of the Jacobian, and the total number of non-zero elements permitted is ML*N. The structure of the Jacobian is reassessed every MU evaluations in case new non-zero terms have evolved. Suggest ML=MU=10. If MU is negative, only one initial evaluation of the Jacobian structure is performed. Obviously for m=4 or 5 to be useful, the resulting Jacobian array must be considerably smaller than it would be at m=2.

(b) COMMON/STIFS/RESTART,JSTART,MAXDER

is optional but may be included to give the user finer control over known discontinuities in the definition of the equation system.

RESTART	Normally when RESTART has its default value of 0, the algorithm permits T to exceed T + DTINT and then interpolates. A discontinuity normally occurs in the middle of a step size and will cause STIFFZ to adjust step size accordingly to compute the transition, but this may be done inefficiently. If a discontinuity is to occur at a known T=T*, it is more efficient to forbid T to exceed T* by setting RESTART negative. For the call to STIFFZ immediately prior to T*, set DTUSED so that the next value is T*, and set RESTART positive. STIFFZ will then return the exact values at T*. The discontinuity may then be introduced without causing problems if the next entry is with JSTART=0. This effectively starts a new problem from initial values at T*.
JSTART	JSTART is normally 1, but if the user wishes to change DTUSED, MF or EPS during a run, he must also set JSTART = -1 when the change is made. To restart the problem from current values, set JSTART = 0.

(c) COMMON/DISCO/DISC1,DISC2

This block is used for control of discontinuities which occur at arbitrary times depending on the evolution of the integration. If DISC1 is set .TRUE. by the user, STIFFZ will return once, after every successfully completed step with DISC2 set .TRUE. The user then may check the definition of discontinuity functions and direct the integration using JSTART if necessary. The theory of discontinuities is too complex to be further discussed here, but is covered adequately by references 2 and 3.

(d) COMMON/STAT/KOUNT(7)

This block may be used to track the progress in integration, its elements monitor the following:

1) The total number of calls to EQNSF
2) The number of successful steps taken
3) The number of calls to EQNSF used for Jacobian evalution
4) The number of Jacobian evaluations
5) The number of steps at DTMIN failing the error
6) The number of steps at DTMIN failing the convergence test
7) The number of steps at DTMAX

Steps at DTMIN are accepted but rated as unsuccessful.

(e) The following three common blocks are used to communicate between various modules in STIFFZ but are not required by the user

CNTROL,SPARS,INT1

ROUTINES CALLED

STIFFZ loads and calls a number of auxiliary routines, some of which may also be used independently. They are:

GEARZ	-	Gear's algorithm, all options
COSET	-	Coefficients for GEARS, all options
DECOMP	-	Decompose a full matrix, m=2
SOLVE	-	Solve equations from DECOMP, m=2
DECB	-	Decompose a banded matrix, m=4
SOLB	-	Solve equations from DECB, m=4
JACOB	-	Determine and pack a sparse Jacobian, m=5
SPARSE	-	Decompose a sparse matrix, m=5
SPARSEB	-	Solve equations from SPARSE, m=5
SORTAG	-	Sort an array of numbers, m=5
DIFFUM	-	Interface for Jacob, m=5
MSCALE	-	Scaling routine for SPARSE, m=5

In the event that a particular option is decided upon, one may prevent the unwanted routines from loading by including dummy subroutines of the same name in the user's deck.

STORAGE

1400 including all routines.

EXIT

STIFFZ returns the current values of T and Y. If INOUT returns a positive value, this is the current order used by the method (maximum 5), and the Y values have been obtained within a per step relative accuracy of EPS.

INOUT less than -1 indicates that the returned solutions may be inaccurate, as steps which do not satisfy error criteria may have been accepted for the following reasons:

INOUT	REASON
-1	A number of steps (KOUNT(7)) were taken at DTMAX so efficiency has been degraded.
-2	Integration failed to satisfy the error test at DTMIN at a total number of KOUNT(5) steps.
-3	Corrector convergence was not achieved at DTMIN at a total number of KOUNT(6) steps.

-4	Trouble with the sparse matrix option, but sparsity was detected to be decreasing, further matrix restructuring was prohibited. The user should increase ML and NWS.
-5	The error criterion imposed appears to be entirely too strict for this problem as 10 consecutive steps have generated INOUT = -2 or -3 having failed the above criteria
-6	Illegal value of N, EPS, or MF probably due to a user blunder or over-write.
-7	The declared working storage contains insufficient room for the requested Jacobian option. If this occurs at T=0, user should change options or increase NWS. It may also occur in the sparse matrix option when sparsity decreases are not detected in advance. In this case increase ML and NWS.

The last three problems are fatal, and if STIFFZ is entered again the program is stopped by the system as further results would be meaningless. Appropriate non-fatal error messages are printed unless MF is entered negative to suppress printing. Messages are always printed for fatal errors.

EXAMPLE

The following routines use STIFFZ to compute the solution to a set of three ordinary differential equations:

$$y_1' = 4.5(y_2-y_3) - 5.5\ y_1$$

$$y_2' = 49.5(y_1-y_3) - 50.5\ y_2$$

$$y_3' = 45.0(y_1-y_2) - 55.0\ y_3$$

with initial condition $y_1 = y_2 = y_3 = 2.0$. The calling program EG sets up parameters, performs printout and calls STIFFZ. The routine EQNSF defines the equations.

```
      PROGRAM EG(OUTPUT,TAPE6=OUTPUT)
C
C        EXAMPLE PROGRAM FOR STIFFZ
C
      DIMENSION Y(13),WS(155)
      COMMON/STAT/KOUNT(7)
      COMMON/BASINT/YCUT,DYCUT,DTMAX,DTMIN,ML,MU
      EXTERNAL EQNSF
C
C        SET INITIAL VALUES AND CONTROLS
C
      PRINT 101
      T=0.0
      EPS=1.0E-05
      N=3
      MF=25
      INOUT=0
      DTINT=5.
      ML=5
      MU=5
      WS(1)=155.
      DO 10 I=1,N
   10 Y(I)=2.
C
C        TRANSFER TO STIFFZ
C
   20 CALL STIFFZ(EQNSF,Y,N,T,DTINT,EPS,H,MF,INOUT,WS)
C
C        TEST ERROR FLAG IF LT 0 STOP
C
      IF(INOUT.LT.0) GO TO 40
C
C        PRINT RESULTS
C
   30 WRITE(6,100) T,H,(Y(I),I=1,N)
C
C        CONTINUE UNTIL T = 10
C
      IF(T.LT.10.) GO TO 20
      GO TO 50
C
C     ERROR FLAG LT 0 PRINT INOUT AND KOUNT THEN STOP
   40 WRITE(6,110) INOUT,KOUNT
      STOP
C
C
   50 CONTINUE
      PRINT 120
C
  100 FORMAT(* TIME*G10.3* STEP*G10.3* Y1*G10.3* Y2*G10.3* Y3*G10.3)
  101 FORMAT(1H1)
  110 FORMAT(* INOUT = *,I3* KOUNT =*,7I5)
  120 FORMAT(* STIFFZ SUCCESSFUL  FINISHED AT TIME = 10.*)
      END
      SUBROUTINE EQNSF(N,T,Y,DY)
      REAL Y(N),DY(N)
C
C        DEFINE THE EQUATIONS HERE
C
      DY(1)=4.5*(Y(2)-Y(3)-Y(1))-Y(1)
      DY(2)=49.5*(Y(1)-Y(2)-Y(3))-Y(2)
      DY(3)=45.Ø*(Y(1)-Y(2))-55.*Y(3)
C
      END
```

```
TIME  5.ØØ     STEP  .111     Y1  .674E-Ø2 Y2  .674E-Ø2 Y3  .252E-14
TIME  1Ø.Ø     STEP  .148     Y1  .454E-Ø4 Y2  .454E-Ø4 Y3 -.617E-16
STIFFZ SUCCESSFUL  FINISHED AT TIME = 1Ø.
```

REFERENCES

[1] M.B. Carver, "Efficient Handling of Discontinuities in Ordinary Differential Equation Simulation" in press for Mathematics & Computers in Simulation, 1978.

[2] M.B. Carver and S.R. MacEwan, "Simulation of an Implicitly defined Differential Equation System Subject to Numerous Discontinuities", in press for Applied Mathematical Modelling, 1978.

AUTHORS

STIFFZ was written by M.B. Carver and D.G. Stewart and incorporates modifications of the GEAR routines of A.C. Hindmarsh, LRL, and the sparse matrix routines of A.C. Curtis and J.K. Reid, Harwell.

DATE

July 1978

ISSN 0067 - 0367

To identify individual documents in the series we have assigned an AECL- number to each.

Please refer to the AECL- number when requesting additional copies of this document

from

Scientific Document Distribution Office
Atomic Energy of Canada Limited
Chalk River, Ontario, Canada
K0J 1J0

Price $5.00 per copy

ISSN 0067 - 0367

Pour identifier les rapports individuels faisant partie de cette série nous avons assigné un numéro AECL- à chacun.

Veuillez faire mention du numéro AECL- si vous demandez d'autres exemplaires de ce rapport

au

Service de Distribution des Documents Officiels
L'Energie Atomique du Canada Limitée
Chalk River, Ontario, Canada
K0J 1J0

Prix $5.00 par exemplaire

2816–79