

UIUCDCS-R-80-1045

UIIU-ENG 80 1743

~~COO-2383-0076~~

DOE/ER/02383-T1

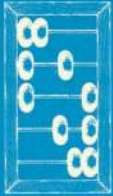
DETECTION AND INTEGRATION OF  
OSCILLATORY DIFFERENTIAL EQUATIONS WITH  
INITIAL STEPSIZE, ORDER AND METHOD SELECTION

by

Kyle A. Gallivan

**MASTER**

December 1980



DEPARTMENT OF COMPUTER SCIENCE  
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

## **DISCLAIMER**

**This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.**

## **DISCLAIMER**

**Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.**

UIUCDCS-R-80-1045

DETECTION AND INTEGRATION OF  
OSCILLATORY DIFFERENTIAL EQUATIONS WITH  
INITIAL STEPSIZE, ORDER AND METHOD SELECTION

by

Kyle A. Gallivan

December 1980

DEPARTMENT OF COMPUTER SCIENCE  
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN  
URBANA, ILLINOIS 61801

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Supported in part by the Department of Energy grant DE-AC02-76ER02383.A003  
and submitted in partial fulfillment of the requirements of the Graduate  
College for the degree of Master of Science in Computer Science.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

leg

THIS PAGE  
WAS INTENTIONALLY  
LEFT BLANK

## ACKNOWLEDGEMENT

The author wishes to acknowledge and thank Dr. C. W. Gear for his suggestions and guidance during the production of this thesis. Also, the author would like to thank all of the members of Dr. Gear's research group for providing an environment conducive to study and research.

## TABLE OF CONTENTS

1.	INTRODUCTION .....	1
	1.1. Nature of the Problem .....	1
	1.2. Method of Solution .....	2
2.	NUMERICAL INTEGRATION OF HIGHLY OSCILLATORY EQUATIONS .....	5
	2.1. Method of Petzold .....	5
	2.1.1. Motivation .....	5
	2.1.2. The Quasi-envelope .....	7
	2.1.3. Generalized Adams Methods .....	11
	2.2. Stiff Oscillating Problems .....	13
	2.2.1. Definition of Stiffness .....	13
	2.2.2. Generalized BDF's and the Jacobian .....	14
3.	DEFINITION OF NEARLY PERIODIC BEHAVIOR .....	16
	3.1. Variable Period Alterations .....	16
	3.2. Petzold's Definition of Nearly Periodic Behavior .....	17
	3.3. A Second Definition of Nearly Periodic Behavior .....	19
	3.4. A Third Definition of Nearly Periodic Behavior .....	21
4.	DETECTION OF NEARLY PERIODIC BEHAVIOR .....	26
	4.1. Phase 0 .....	26
	4.2. Phase 1 .....	26
	4.3. Phase 2 .....	28
	4.4. Phase 3 .....	29
5.	TRANSITION DECISION AND METHOD SELECTION .....	31
	5.1. Description of the Decision Process .....	31
	5.2. Estimation of the Lipschitz Constant .....	33
	5.2.1. First Strategy .....	34
	5.2.2. An Analytical Jacobian Possibility .....	34
	5.2.3. Present Strategy .....	37
6.	IMPLEMENTATION .....	43
	6.1. An Overview .....	43
	6.2. The Basic Routines .....	47
	6.2.1. ZCROSS .....	47
	6.2.2. FNDPER .....	50
	6.2.3. NORDSK .....	52
7.	EXPERIMENTAL RESULTS .....	54
	7.1. Test 1 .....	54
	7.2. Test 2 .....	54
	7.3. Test 3 .....	56

7.4. Test 4 .....	57
7.5. Test 5 .....	60
8. CONCLUSION .....	62
LIST OF REFERENCES .....	64



## 1. INTRODUCTION

### 1.1. Nature of the Problem

Within any general class of problems there typically exist subclasses possessed of characteristics which can be exploited to create techniques more efficient than general methods applied to these subclasses. Two such subclasses of initial value problems in ordinary differential equations are stiff and oscillatory problems. Indeed, the subclass of oscillatory problems can be further refined into stiff and nonstiff oscillatory problems. This refinement will be discussed in detail later.

The efficacy of such specialized techniques can be reduced to nil if the user is not aware of the existence of the special property in his problem. Further, if a method of detection is postulated it would be desirable for this method to have the capability of determining the lack of the special property since using a special technique on a problem to which it does not apply can be at best inefficient (as with stiff methods) or at worst impossible (as with oscillatory methods). It would seem likely that within the method of detection it would be possible to not only detect a certain characteristic but to also accumulate information about the nature of the characteristic, thus enabling a better performance from the special technique to be applied.

Once a method of detection has been proposed and demonstrated to be feasible, a control structure must be developed upon which a production code could be based. This control structure would involve such aspects

as interfacing the method of detection with the special technique, and providing for consistency of common processes within the detection and special technique phases. This aspect of consistency is crucial since the method of detection must use the same criteria for evaluating the characteristic as the special technique does lest the method of detection decide the characteristic exists and the special technique deny it.

This thesis addresses the problem of developing a method of detection for nonstiff and stiff oscillatory behavior in initial value problems. Given this method of detection a control structure is proposed upon which a production code could be based. An experimental code using this control structure will be described and results of numerical tests will be presented.

## 1.2. Method of Solution

In 1957, astronomers introduced methods for the calculation of orbits of artificial satellites known as multirevolutionary methods. Details can be found in Mace and Thomas [1], Graff [2], and Graff and Bettis [3]. The first step in these methods is to identify a certain point on the orbit using knowledge of the physical phenomena being modelled. After this point is identified a small step integration takes place from one such reference point to the following one. The information obtained from this integration is used to extrapolate to future orbits by means of formulae similar to those discussed in section 2.

Petzold [4] combined the idea of the multirevolutionary methods and a concept she called the quasi-envelope of the solution to form a new strategy for highly oscillatory problems. This quasi-envelope will be discussed in section 2. Further, she generalized the notion of the period of oscillation which liberated the problem from its physical knowledge prerequisite and vastly increased the range of applicable problems. Her definition of the period was based upon minimization of the integral of a norm of the difference of the function over two successive periods. Petzold's original definition of the period and possible alternatives will be discussed in section 3. This implementation of multirevolutionary methods, with slight modifications, will serve as the special technique used once highly oscillatory behavior has been detected.

Gear [5] proposed an alternate definition of the period of oscillation, which could be used in conjunction with Petzold's method, based upon zero crossings of a linear combination of the derivatives of the system. Using this definition of the period, he presented a method of detection which performed satisfactorily on a fairly complex numerical example. Also, he implemented the suggestion of Petzold to generalize the backward differentiation formulae to facilitate the integration of stiff oscillatory equations. This alternate definition of the period and the generalized BDF's will be discussed in sections 3 and 2 respectively. Lastly, Gear suggested that this method of detection could be used to provide a higher order start to Petzold's

method.

The work of Gear demonstrates the feasibility of a detection strategy based upon a linear combination of the derivatives. However, since it was only a feasibility study it did not address the attributes of a detection strategy which were deemed desirable in section 1.1. By using the proposed strategy of Gear as a base, it is possible to develop a more useful and sophisticated algorithm. The extensions explored in this thesis are as follows:

1. providing up to a third order start for the oscillatory integrator,
2. selecting initial stepsize, order, and method for the oscillatory integrator,
3. invoking the oscillatory integrator only if it is more efficient than nonoscillatory techniques,
4. reverting to nonoscillatory techniques when periodic behavior is no longer evident or when it is no longer efficient to use oscillatory techniques,
5. providing a control structure upon which a production code can be based.

The numerical test results presented seem to indicate that the above extensions are feasible and that the control structure presented could be used as the basis for an efficient production code.

## 2. NUMERICAL INTEGRATION OF HIGHLY OSCILLATORY EQUATIONS

In this section, Petzold's strategy for integrating highly oscillatory equations is considered in some detail along with the problem of stiffness in oscillatory problems. Section 2.1 discusses the development and motivation of the generalized Adams methods given in Petzold's thesis. Section 2.2 covers the definition of stiffness and the derivation of the generalized BDF's. Also covered in 2.2 is the definition of the Jacobian of the quasi-envelope and techniques for its evaluation.

### 2.1. Method of Petzold

#### 2.1.1. Motivation

This section will describe the motivation behind the strategy which will be used to integrate the system of equations.

Assume that the system of ordinary differential equations under consideration is known to have a solution which is nearly periodic in  $t$ . Further assume that the period, denoted  $T$ , is either constant or slowly varying with time. Given the previous assumptions it is fairly clear that the result of integrating from 0 to  $kT$ , where  $k$  is some integer, can be approximated by

$$k \int_0^T f(y(t), t) dt. \quad (2.1)$$

This could be written as

$$y(H+t) - y(t) \approx H\bar{f}(t), \quad (2.2)$$

where  $H = kT$ ,

$$\bar{F}(t) = \frac{1}{T} \int_0^T f(\tilde{y}(t+s), t+s) ds,$$

and  $\tilde{y}$  satisfies,

$$\frac{d}{ds} \tilde{y}(\tau+s, \tau) = f(\tilde{y}(\tau+s, \tau), \tau+s),$$

$$\tilde{y}(\tau, \tau) = y(\tau).$$

If  $y$  is replaced by  $z$  in (2.2) the result can be viewed as one step of Euler's method applied to the initial value problem

$$z'(t) = \bar{F}(t), \quad z(0) = y(0), \quad (2.3)$$

where  $\bar{F}(t)$  is as defined in (2.2).

The integration of this initial value problem can be viewed as the approximate construction of what Petzold calls the quasi-envelope. Notice that each step of length  $H$  in the integration of  $z$  calls for a small step integration (known as the inner integration) over one period of the function  $y$ . In this way, the small step integration can be considered a type of derivative evaluation for the quasi-envelope  $z$ . Also, since it is desired to use this method on functions with slowly varying periods, it becomes necessary to develop an algorithm for determining the local period of the function  $y$  starting at some initial point on  $z$ . After integrating  $y$  over one period and determining the slope of the line connecting the two extreme  $y$  values, the value of  $z(t+H)$  is determined by extrapolation with the formulae of the type described in section 2.1.3 or 2.2.2. This extrapolation is known as the outer integration since, as previously noted, it resembles the application of an integration method to the problem (2.3).

### 2.1.2. The Quasi-envelope

The function  $z$  of the previous section can be viewed intuitively by passing a smooth curve through points on  $y$  which are at intervals of length  $T$ . (See figure 1). Taking this view of the quasi-envelope allows the process of the outer integration to become fairly obvious. It is another matter entirely to present some sort of analytical definition of the quasi-envelope. However, the definition of Petzold will be mentioned briefly to indicate some of the difficulties involved and an alternative definition is proposed for a case where smoothness presents a problem.

The function  $z$  can be defined in terms of its values on  $[0, T]$  by

$$z(t+T) = z(t) + Tg(z, t), \quad z(0) = y(0), \quad (2.4)$$

where  $g(z, t) = \frac{1}{T}[\bar{y}(t+T, t) - \bar{y}(t, t)]$ , and  $\bar{y}$  satisfies

$$\frac{d}{ds} \bar{y}(t+s, t) = f(\bar{y}(t+s, t), t+s), \quad \bar{y}(t, t) = z(t).$$

Obviously, the above definition only specifies  $z$  at integer multiples of  $T$ . In order to complete the definition of  $z$  on  $[0, T]$  and satisfy the fact that  $z$  must be a smooth curve, it is necessary to define a sequence of functions, starting with the above function, which can be used to approximate  $z$ . For the definition of the sequence and related theorems the reader is referred to Petold [4].

The purpose of the outer integration is to construct the quasi-envelope and not to follow the underlying solution. Notice that this implies that the underlying function could be integrated, when

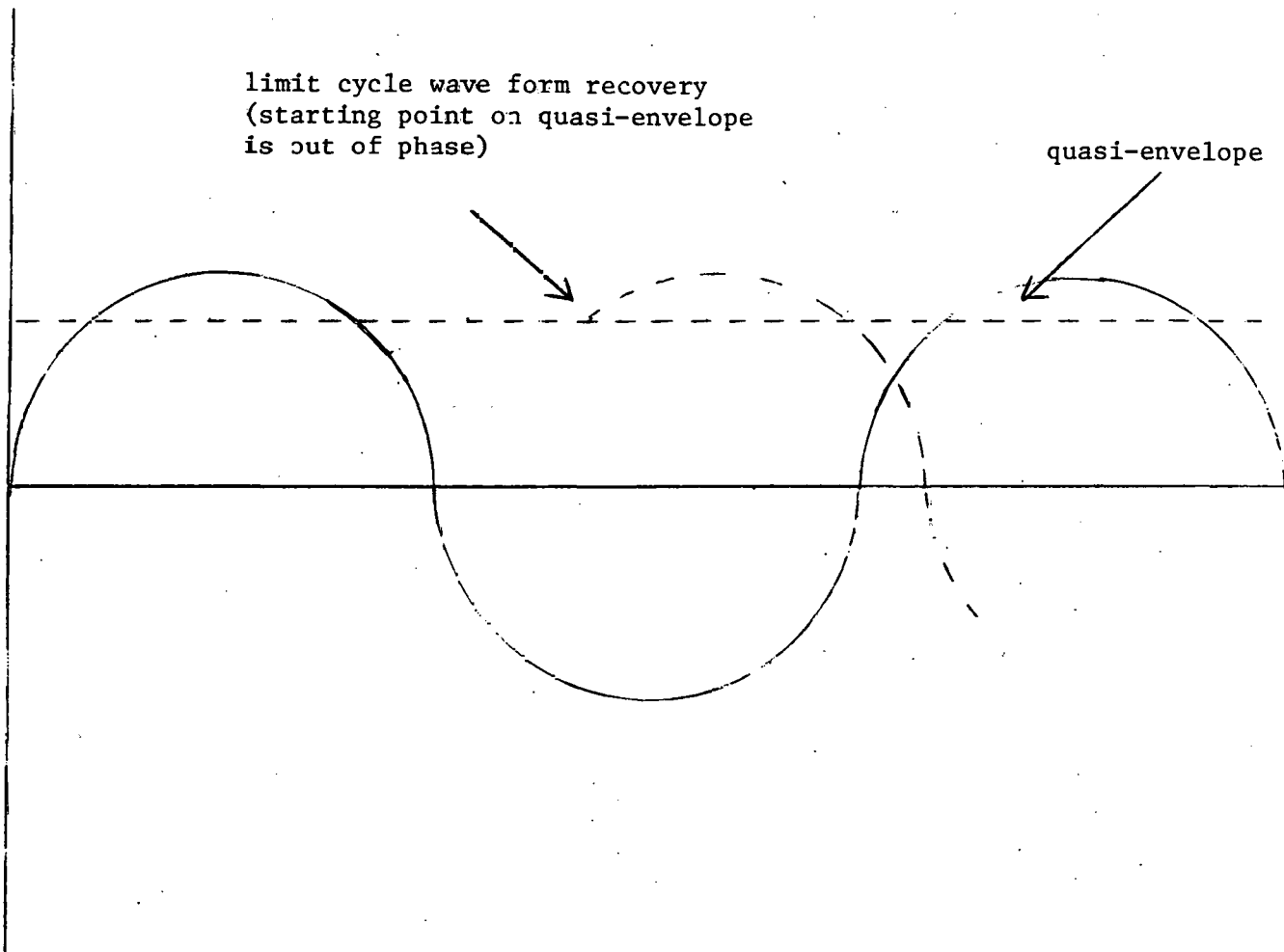


Figure 1. Quasi-Envelope



determining the local period, with initial conditions which are out of phase with the solution to the given problem. Hence, it is expected that problems which are sensitive to phase changes may generate problems in constructing a smooth quasi-envelope. If the problem is autonomous this is not a problem since  $f(y)$  is phase insensitive. If, however, the oscillation is caused by a driving term, an attempt to integrate in the period algorithm from a point on  $z$  with initial conditions not in phase with the underlying solution can cause anything from nonsmoothness of  $z$  to the nonexistence of periodic behavior on that particular integral curve.

As an example of a problem which can have trouble with the smoothness of the quasi-envelope consider,

$$y' = -y + \sin \lambda t. \quad (2.5)$$

The solution of (2.5) is

$$y(t) = Ce^{-t} + \frac{1}{1+\lambda^2}(\sin \lambda t - \lambda \cos \lambda t).$$

Obviously, the limit cycle of the solution is a sinusoidal curve. Notice that all around this limit cycle are integral curves which are sinusoidal waves imposed on exponentials. Suppose that  $z(t) = 0$  is the quasi-envelope of interest and is such that  $\sin \lambda \tau - \lambda \cos \lambda \tau = 0$ . Then, if the  $(\tau, y(\tau))$  given to the period algorithm is  $(\tau, 0)$  the small step integrator will produce a function in phase with the underlying solution. (In fact, this function will be identical to the limit cycle). After integrating through one period the  $y$  value would once

again be back to  $y = 0$ , as expected. Now suppose that the period algorithm is given initial conditions  $(\tau', 0)$  where  $(\tau', 0)$  is not on the limit cycle. This has the effect of placing the inner integrator on a completely different solution and after integrating for one period the  $y$  value will not be at zero, so the slope of the line connecting the extreme  $y$  values will be far from the zero slope present on the limit cycle. This example shows fairly clearly how the integral curve may no longer be periodic. In this case, the integral curve is imposed on an exponential which does not lend itself to a definition of periodicity if the exponential is steep enough.

It seems reasonable to attempt to form a new definition of the quasi-envelope which will keep the driving term in phase with the solution of interest as much as possible. Problems of the above type can be characterized by the fact that the derivatives can be separated into a nearly autonomous term and a term periodic in  $t$ . (Nearly autonomous is taken to mean that the partials with respect to time of this term are small). Assuming this has been done to the system under consideration, it is then possible to propose the following definition of the quasi-envelope.

The derivative can be written

$$f(y,t) = f_A(y,t) + f_p(y,t), \quad (2.6)$$

where  $f_A$  is nearly autonomous and  $f_p$  is periodic in  $t$ . Given  $z(t)$ , define  $z(t+T) = \hat{y}(T)$ , where  $\hat{y}(s)$  is given by

$$\frac{dy}{ds} = f_A(t+s, \hat{y}(s)) + f_p(s, \hat{y}(s)),$$

$$\hat{y}(0) = z(t).$$

This definition of the quasi-envelope allows the recovery of a wave form similar to the solution from any point on  $z$ . This is the same capability which is present when the system is autonomous and produces a smooth quasi-envelope.

Notice that the above definition of the quasi-envelope does not require a special procedure to be added to the algorithm for determining the local period of the function. The same effect can be achieved by requiring that the outer integrator take steps which are integral multiples of the period. This forces the driving term to always start in the same phase as the solution being followed. Petzold called this mode of operation the synchronized mode. In general, it is safer to run in the synchronized mode since it will minimize the loss of phase information. The nonsynchronized mode, that is allowing nonintegral multiples of the period to be used as outer stepsizes, can be used successfully for autonomous problems with slowly varying periods and nonautonomous problems which are not highly dependent on  $t$ .

### 2.1.3. Generalized Adams Methods

It is now time to consider exactly what type of method is to be used to extrapolate values of  $z$ . As seen in the definition of  $z$  in formula (2.4),  $g$  can be viewed as a type of derivative. One would suspect that some sort of linear multistep formula could be adapted to perform the outer integration. This is indeed the case and one such set

of generalized multistep methods are the generalized Adams formulae. The derivation presented here is due to Petzold. The following operator relationships are needed.

$$\nabla z(t) = z(t) - z(t-H)$$

$$\nabla = I - e^{-HD}$$

$$\Delta z(t) = z(t+H) - z(t)$$

$$\Delta = e^{HD} - I$$

$$Ez(t) = z(t+H)$$

$$E = e^{HD},$$

where  $D$  is the differentiation operator. Formally,

$$z_N - z_{N-1} = (I - e^{-HD})z_N \quad (2.7)$$

and

$$z_N = \left( \frac{e^{TD} - I}{T} \right)^{-1} \xi_N.$$

Substituting the latter relation into (2.7) and using the expression for  $\nabla$  yields

$$z_N - z_{N-1} = \nabla \left( \frac{e^{TD} - I}{T} \right)^{-1} \xi_N. \quad (2.8)$$

Note  $HD = -\log(I-\nabla)$ , so  $TD = -\frac{T}{H}\log(I-\nabla)$ . Hence,

$$\left( \frac{e^{TD} - I}{T} \right)^{-1} = \left( -\frac{1}{H} \log(I-\nabla) + \frac{1}{H} \frac{T}{H} \frac{1}{2!} \log^2(I-\nabla) + \dots \right)^{-1}.$$

Combining this and (2.8) and inverting the series representation of

$$\left( \frac{e^{TD} - I}{T} \right)^{-1} \text{ gives}$$

$$z_N - z_{N-1} = \Psi(-I - \frac{T}{H} \frac{1}{2!} \log(I-\nabla) - \frac{1}{12} (\frac{T}{H})^2 \log^2(I-\nabla) + \dots) g_N,$$

where  $\Psi = \nabla H(\log(I-\nabla))^{-1}$ . This can be written as

$$z_N - z_{N-1} = H[(I - \frac{1}{2}\nabla - \frac{1}{12}\nabla^2 - \dots) - \frac{1}{2} \frac{T}{H}\nabla + \frac{1}{12} (\frac{T}{H})^2 \nabla + \dots] g_N, \quad (2.9)$$

which is the formula for the generalized Adams methods.

## 2.2. Stiff Oscillating Problems

### 2.2.1. Definition of Stiffness

The generalized Adams formulae work well with many oscillating problems, however, as with conventional initial value problems, there are certain oscillating problems which can not be solved efficiently by Adams type methods. These oscillating problems are called 'stiff'. This concept of generalized stiffness is very similar to the conventional concept of stiffness but it is not identical.

Recall that a conventional stiff problem is one in which integral curves tend rapidly to a slowly varying solution. This forces most conventional integrators to use small stepsizes even though the solution appears to be smooth enough to allow large steps. The same type of behavior is possible with oscillating problems. A stiff oscillating problem is characterized by the fact that integral curves tend very rapidly to the limit cycle. This can be expressed alternately in the following way. If the quasi-envelope is graphed along with the quasi-envelopes of the integral curves of the problem the resulting graph would look very much like a conventional stiff problem since the quasi-envelopes of the integral curves would tend very rapidly to the quasi-

envelope of the solution. For a discussion of stiffness and a related discussion of the stability regions for generalized methods the interested reader is, once again, directed to Petzold [4].

### 2.2.2. Generalized BDF's and the Jacobian

Since the quasi-envelopes of the stiff oscillating problem look like the integral curves of a conventional stiff problem it would be expected that a stiff oscillating problem could be handled with a generalized form of a conventional stiff method such as the BDF's. This was suggested by Petzold [4] and implemented by Gear [5].

The generalized BDF's are fairly easy to derive if one takes the viewpoint of passing an interpolating polynomial through past values on the quasi-envelope and then setting the forward difference over one period equal to  $g$ . Or more formally, let  $p(t)$  be the polynomial which interpolates  $z$  over  $k$  past values. Setting the forward difference of  $p$  over one period to  $g$  yields

$$\frac{p(t_n + T) - p(t_n)}{T} = g(z_n, t_n).$$

Expanding  $p$  in terms of backward differences gives,

$$Hg_n = \sum_{i=1}^{i=k} (-1)^i \binom{-r-1}{i} \frac{1}{r+i} \nabla^i z_n, \quad (2.10)$$

where  $r = \frac{T}{H}$ ,

$$\binom{-r-1}{i} = \frac{(-r-1)(-r-2)\dots(-r-i)}{i!},$$

and the backward differences are over an interval of length  $H$ . This is the formula for the generalized BDF's.

One major difficulty with the above methods is that the solution of the corrector equation now requires knowledge of the Jacobian,  $J = \frac{\partial g}{\partial z}$ , of the quasi-envelope because equation (2.10) is implicit in  $z_n$  and  $J$  is large. This Jacobian indicates how much the slope of the line connecting the two  $y$  values at each end of a period will change if the  $y$  values at the beginning of the integration over that period are changed.

Some means of evaluating this Jacobian at occasional outer integration steps is required. Gear [5] used numerical differencing to accomplish the evaluation. This was done by perturbing each of the initial  $y$ 's then calling the period algorithm to evaluate the  $g$ 's. The changes in  $g$  versus the resulting changes in the  $y$ 's form the Jacobian. Obviously, this technique is a bit expensive since it requires the integration of  $n$  systems, each with  $n$  equations, at each Jacobian evaluation. Further, the choice of the size of the perturbation is critical. Integral curves may be nearly periodic only when very close to the oscillatory solution and since the period finding is done with a numerical algorithm it is quite possible that it will be impossible to evaluate the Jacobian by differencing. In practice, however, numerical differencing seems to work fairly well, albeit expensively. In the section concerning the decision of whether or not to invoke oscillatory techniques after finding oscillatory behavior, a possible alternative to the use of numerical differencing for evaluating the Jacobian is discussed.

### 3. DEFINITION OF NEARLY PERIODIC BEHAVIOR

In this section, the possible definitions for the local period finding algorithm are discussed along with the change of variables necessary to facilitate the integration of problems with slowly varying periods.

#### 3.1. Variable Period Alterations

The derivations of both the generalized Adams and BDF's assumed that the problem under consideration had a constant period. Since the integration technique is to be used with varying period problems also, some alterations must be made. Petzold provided for the integration of varying period problems by proposing a change of variables to an independent variable with a constant period. This can be accomplished by appending to the system an extra equation relating  $t$  to this new independent variable. The constant period has been normalized to length one in the algorithm presented in this thesis. This was done in order to allow easy preparation of information for the outer integrator in the detection program and to force the new independent variable to correspond to the number of periods which have been integrated. With this correspondence, it seems that it might be possible, for autonomous problems operating in the nonsynchronized mode, to place the quasi-envelope back into phase with the underlying solution by forcing the independent variable to assume integral values in any region in which the phase is of interest.

Let  $T(t)$  be the slowly varying period with respect to  $t$  and let  $\tau$



be the new independent variable with a constant period of 1. The equations

$$t(\tau+1) - t(\tau) = T(t(\tau)),$$

$$t(0) = t_0$$

accomplish the change of variables. The system to be integrated by the outer integrator is now

$$z(\tau+1) = z(\tau) + g(\tau)$$

$$t(\tau+1) = t(\tau) + T(t(\tau)),$$

where

$$g(\tau) = y(t(\tau)+T(t(\tau))) - y(t(\tau)),$$

and  $y$  is the function integrated by the period algorithm.

### 3.2. Petzold's Definition of Nearly Periodic Behavior

In this section, the definition of nearly periodic behavior used by Petzold is discussed.

Suppose the function  $y$  is known to be periodic in  $t$  with period  $T$ . Then, the difference of the functions evaluated a period apart would be zero, so the integrals of its norm would be zero. Petzold used this fact as the motivation for her definition of nearly periodic and the algorithm to find the local period.

The local period, in this definition, is taken to be the value of  $T$  which minimizes

$$I(t, T) = \int_t^{t+T} \|y(\tau+T) - y(\tau)\|_2^2 dt$$

where  $T_n$  is the last estimate of the period. The minimization is done by Newton's method so,

$$T_{n+1} = T_n - \frac{F(T_n)}{F'(T_n)},$$

where

$$F(T_n) = \sum_{i=1}^N \int_0^{T_n} (y_i(t) - y_i(t+T)) y_i'(t+T) dt,$$

and

$$F'(T_n) = \sum_{i=1}^N \int_0^{T_n} y_i''(t+T)(y_i(t) - y_i(t+T)) - (y_i'(t+T))^2 dt.$$

In her code, Petzold accomplished the evaluation of  $F(T)$  and  $F'(T)$  by initially integrating over one period and then for every Newton step integrating the function over the second period and performing the quadrature on the differences as this integration progressed. While this algorithm performed satisfactorily as far as finding a period is concerned, it has its problems. The first of these problems is the fact that it is so expensive. The reintegration of the second period at each step is costly in time. If values were stored for the integration over the two periods and the quadrature performed by interpolation, the storage costs would rise to unacceptable levels. Secondly, the algorithm uses Newton's method for finding the minimum, hence, it requires a fairly good initial guess at the period. In the outer integration phase, this is acceptable since the predicted difference of the appended equation is the expected value of the period. However,

this dashes all hopes of attempting to modify this algorithm into some sort of detection strategy. A third point is not really a major drawback but it can be bothersome from a user's point of view. Since the function values must be stored for the first period (or both if interpolation/quadrature is used) it is necessary to specify the expected number of steps to be used by the inner integrator to integrate over one period. This quantity may fluctuate throughout the problem, hence, wasting space, or the user may have absolutely no idea as to what amount to use.

### 3.3. A Second Definition of Nearly Periodic Behavior

To motivate the next definition of nearly periodic note the fact that two points with an interval of a period between them have exactly the same function and first derivative values. (Obviously this is true for all derivatives but only up to the first will be considered since these values are the more accurately known than the higher derivatives in a numerical integration scheme). The algorithm based on this fact would first integrate to a point on the curve which is near the expected period. Next, it would begin comparing the function values and first derivative values at the present point to those at the left hand side of the period. When both values agree, within some tolerance, the period has been found and the algorithm can return the period and the required  $g$  values.

The above algorithm still requires a good estimate of the period but, it does not have the time and storage expense of Petzold's

algorithm. In the implementation of this algorithm which was tested, the values were not compared with the end points until 0.9 of the expected period was reached. (If the new period is less than 0.9 of the estimate, it suggests that the period is changing too rapidly and the outer stepsize should be reduced). It was also found that the better values to compare were the first derivatives because these are periodic even if the  $y$  values have a superimposed linear term.

The above comparison of derivative values can be viewed as the minimization of

$$I(t) = \sum_{i=1}^n \beta_i (\bar{y}'_i - y'_i)^2,$$

where  $\bar{y}'_i$  is the derivative value of the  $i$ th component at the point integrated to and  $y'_i$  is the corresponding derivative value at the left side of the interval.

Hence, the root of

$$F(t) = \sum_{i=1}^n \beta_i \bar{y}'_i (\bar{y}'_i - y'_i)$$

must be found.

$$F'(t) = \sum_{i=1}^n \beta_i (\bar{y}'_i)'^2 + \beta_i \bar{y}'_i (\bar{y}'_i)'' (\bar{y}'_i - y'_i)$$

but, the term involving  $\bar{y}'_i$  is ignored since, for a periodic function, the term it multiplies would be zero. Further,  $\bar{y}'_i$  would be equal to  $y'_i$  for a periodic function so,  $F'(t)$  is evaluated only at the beginning of the algorithm. Note that the second derivative is needed. This is

taken care of by a higher order start routine, Gear [6], and maintaining at least a second order integration.

The algorithm proceeds:

1. Integrate to 0.9 of the expected period.
2. Calculate  $I(t)$ .
3. Integrate until  $I(t)$  begins to increase.
4. Perform a Newton iteration to find the minimum of  $I(t)$  and hence the period.
5. Interpolate to find the function values at the period point.

The above algorithm does not lend itself that easily to a detection strategy. This can be seen from the fact that the algorithm is fairly dependent on the left hand endpoint being fixed. In the detection process, this is not the case. So any time a new  $y(t)$  is being considered it must be compared against some chosen left hand point  $y(s)$ . Of course, the location of  $s$  is not known ahead of time and guesses at its location must be made. The resulting necessity of heuristic programming precludes the modification of this algorithm for detection purposes.

### 3.4. A Third Definition of Nearly Periodic Behavior

Any linear combination of a set of periodic functions, all with the same period, is periodic. The derivative of a periodic function must change sign at least twice in each period. Suppose this linear

combination was chosen such that its derivative was zero at the endpoints of the period under consideration. Obviously, such a linear combination could be used as the basis for an algorithm to determine the period of a function.

The above linear combination can indeed be defined for a system of ordinary differential equations in the following way. Let  $u$  be the vector of function values given as the initial conditions of the equation to be integrated for one period. Define a function  $P$  as follows,

$$P(y) = \langle c, y' \rangle$$

for  $y$  produced while integrating in the interval and  $P(u) = 0$ . Obviously, the key is the choice of the vector  $c$ . Since the object of the algorithm will be to find a zero crossing of  $P$ , choose  $c$  such that  $P(u) = 0$ ,  $\|c\| = 1$ , and  $P'(u)$  is maximized. It can be shown that the unscaled form of  $c$  is

$$c_1 = u_1'' - \alpha u_1'$$

where

$$\alpha = \frac{\langle u', u'' \rangle}{\langle u', u' \rangle}$$

The setting of  $c$  requires knowledge of  $u''$  and the evaluation of  $P(y)$  requires  $y''$ . As before, the necessary information will be obtained from the higher order starting routine.

Notice that  $P(y)$  alone will not be sufficient to determine the period of the function since even the simplest function will have two

crossings per period. Since  $P(u)$  is a positive going crossing, only positive crossings will be considered. Also, at each positive going crossing  $D(y) = || y'(t_1) - y'(t_2) ||$  will be computed, where  $t_1$ , and  $t_2$  are the crossing times. If this norm is small the point will be accepted as a periodic point.

There are certain situations which can cause trouble for the above definition of the  $c$  vector. If  $u'' = 0$  the  $c$  vector is all zeros. This, of course, defeats the entire purpose of the algorithm. In the detection phase this does not cause that much trouble since it is possible to just continue a small step integration until the situation no longer exists. It is not clear what should be done if this occurs in the oscillatory integration phase. The easiest thing to do would be to merely behave as if no period was found and reduce the outer stepsize. The  $c$  vector will also be zero in the cases when  $u'$  and  $u''$  are in the same direction or there is only one equation in the system. If  $|| u' || = 0$  then  $\alpha$  is undefined and consequently,  $c$  is undefined.

All of the above cases, except  $u'' = 0$ , can be handled by the following change to the formulation of  $c$ . Append to  $u'$  a component whose value is  $R$ , where  $R$  is the predicted period at this step.  $R$  can be viewed as an approximation of the derivative of the appended time equation. Also, since time is linear, append to  $u''$  a component whose value is 0. The new form of  $P(y)$  is

$$P(y) = \langle c, y' \rangle - R^2 \alpha,$$

where

$$c_i = u_i'' - \alpha u_i'$$

and  $\alpha$  is now given by

$$\alpha = \frac{\langle u', u'' \rangle}{R^2 + \langle u', u' \rangle}$$

There is another situation which can cause trouble for the detection and period finding routines. If there is a point in the interval of interest which has the same value of  $y'$  as the initial point or last crossing and this point is not a periodic point,  $D(y)$  will not be able to distinguish it from a periodic point. In the case of an autonomous problem, this can be handled by looking at  $y$  values since they must differ at all but periodic points when  $y'$  is the same.

The algorithm based on the above linear combination can be summarized as follows. Upon entering the algorithm, set the value of the vector  $c$ . (Actually this is done after the first small step integration which allows the starter to calculate the necessary derivatives). Next, locate a positive going sign change of  $P(y)$ . After finding such a change, use a Newton algorithm to find the values of the function and its first and second derivatives at the crossing point. Compute  $D(y)$ . If  $D(y)$  is small accept this crossing as the periodic point and calculate  $g$ . If it is not small find the next positive crossing and repeat the process.

The above algorithm functions well as a local period finder and it is possible to create a detection strategy from it, as will be seen in the next section. Note that it is much more efficient than the original definition proposed by Petzold since it only integrates through one



period and it does not require excessive amounts of extra storage. Also, it does not depend upon an initial estimate of the period to converge to a value. The present implementation does make use of such an estimate, however, to terminate, when it has become apparent that the function is no longer periodic, in which case the outer stepsize should be reduced.

#### 4. DETECTION OF NEARLY PERIODIC BEHAVIOR

This section concerns the detection algorithm generated from the third definition of nearly periodic behavior of the last section. Each phase of the strategy will be discussed in its own subsection. Throughout the following discussions references will be made to key actions without any detail being given as to the way these actions are to be performed. All of these key actions, such as 'find the next zero crossing', will be discussed in detail in the section dealing with implementation. Any reference to a zero crossing of  $P(y)$  is understood to be a positive going zero crossing. Figure 2 shows a state diagram relating the phases of the algorithm.

##### 4.1. Phase 0

This phase is a start up phase in which nothing is known or assumed about the behavior of  $P(y)$ . The  $c$  vector is set to all ones. An initial integration is made and  $P(y)$  is evaluated to determine its sign. After this has been done, an initial zero crossing is found and control passes to Phase 1.

##### 4.2. Phase 1

Having found an initial zero crossing in Phase 0, it is the job of Phase 1 to find the first set of crossings which seem to constitute a period. This is accomplished in a loop which terminates once a start up period is found.

The loop consists of the following actions. The next zero crossing is found, and information about it is saved. (Presently, up to 10 past

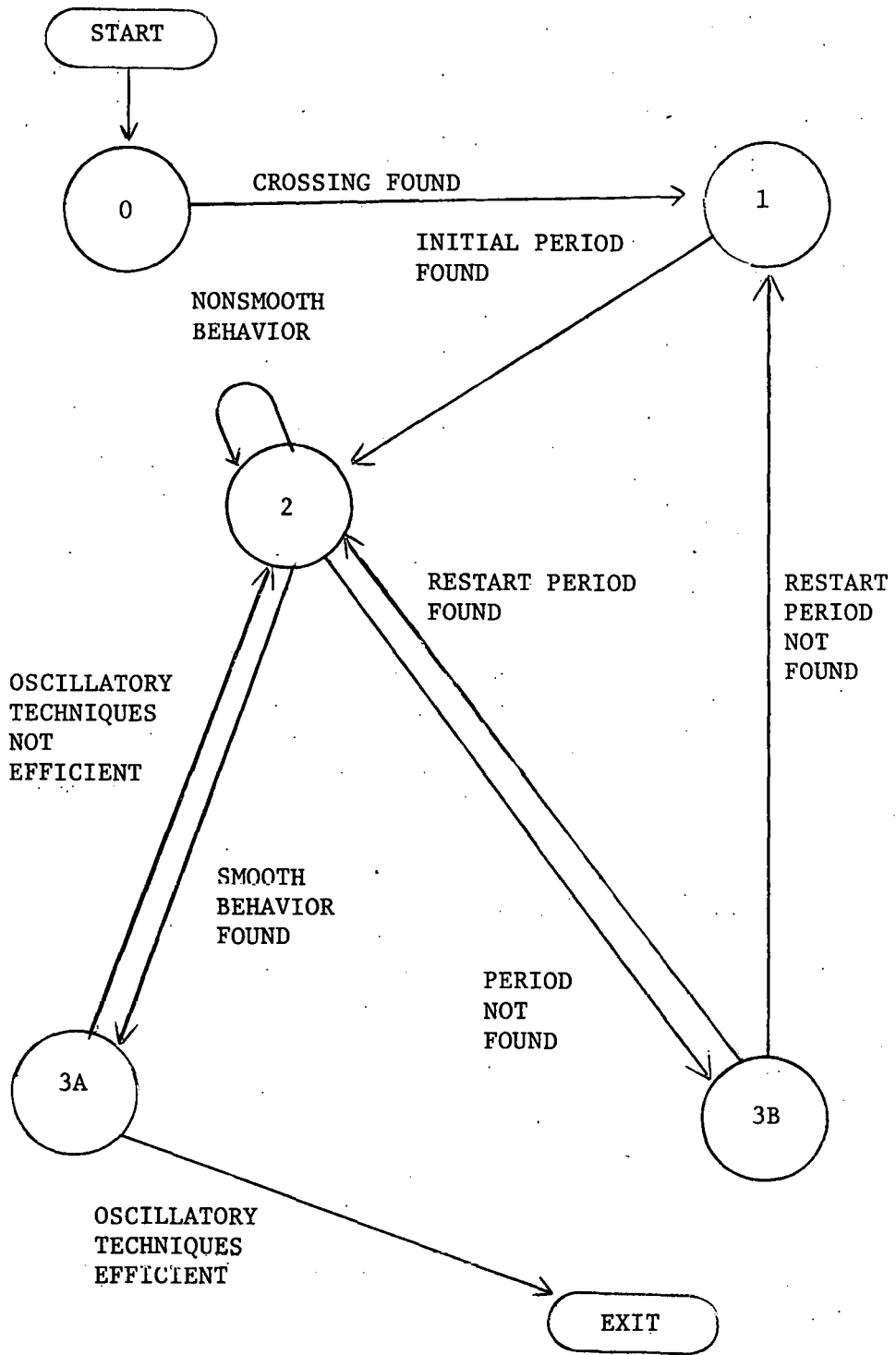


Figure 2. State Diagram of Phase Transition

crossings are saved for use). The last crossing generated is then compared with each of the previously generated crossings by way of an evaluation of  $D(y)$ . If any of the evaluations turn out to be small, that particular pair of crossings is taken to be a potential period and information concerning the period is initialized. Otherwise, the loop is repeated until such a pair of crossings is found.

The information on the periods which is saved includes the endpoints of the periods, the first three backward differences of periods generated, and the time components of the period endpoints. Upon initializing the above information, control passes to Phase 2.

#### 4.3. Phase 2

This phase is very similar to a call to the period algorithm from the outer integrator. The left endpoint of the period is fixed at the right endpoint of the period generated in Phase 1 and an attempt is made to find a period which is close to the previous one.

First, the  $c$  vector is set to the values which maximize  $P'$  and make the initial point a zero (or, in this case, keep it a zero). Now an attempt is made to produce a period with this fixed left point. If no such period is found control transfers to Phase 3 B. If the search is successful then the information concerning the periods generated so far is updated. The backward differences of the periods are then checked to see if any one is small indicating a smoothly varying period. (This is necessary since variable periods are allowed. If a constant period was expected only the first difference need be considered). If a smoothly

varying period is found control transfers to Phase 3 A. If the periodic behavior is not smooth control remains in Phase 2 and the attempt is repeated with the right side of this period now serving as the fixed left point.

#### 4.4. Phase 3

As indicated in the previous section, action in this phase is dependent upon the result of Phase 2. If smoothly varying periodic behavior has been found 3 A is invoked. If Phase 2 failed to find a period 3 B is invoked.

Phase 3 A makes the decision concerning the feasibility of using oscillatory techniques. This is done by predicting the stepsize, order and method to be used in the outer integration based upon the information accumulated about the last periods generated. Given the stepsize and method a decision is made as to whether or not the use of oscillatory techniques would be efficient. If efficiency is found to be the case then information is prepared for the outer integrator and it is invoked. If the action is deemed not efficient then control transfers to Phase 2.

Phase 3 B attempts to restore an initial period so that Phase 2 may be restarted. This is done by comparing the last crossing generated in Phase 2 to all those previously generated. (The comparison is, of course, an evaluation of  $D(y)$ ). If a small  $D(y)$  value is found then Phase 2 is restarted. If no such pairing is found then control

transfers to Phase 1 with the last crossing acting as if it was generated in Phase 0.

## 5. TRANSITION DECISION AND METHOD SELECTION

5.1. Description of the Decision Process

Two formulae of concern in this section are

$$E = C_{k+1}^* \left(\frac{H}{T}\right)^{k+1} \Delta^{k+1}_z, \quad (5.1)$$

and

$$H < \frac{1}{L|\beta_0^*|},$$

where  $L$  is the Lipschitz constant of  $g$ . (Recall that  $T = 1$  by definition).

The first is the expression given by Petzold as the local truncation error for a generalized method. The second is the condition on  $H$  which must be satisfied for functional iteration to converge. These two relations will serve as the basis for the transition decision and the method selection. For simplicity's sake, the second relation will be changed to restrict  $H$  to be less than the reciprocal of the Lipschitz constant. (This is a more stringent requirement on  $H$ ).

The major concern at this point is how the decision phase will use these relations to decide when to use oscillatory techniques. After that question is answered, it is necessary to consider how the outer integrator will decide that the detection program should be recalled or that stiffness has set in. The latter question is highly dependent on the method used to estimate the Lipschitz constant. This issue is considered and the full strategy used in the outer integrator is presented in the next section.

The detection routine saves backward differences of the period and the  $z$  values for smoothness considerations. Up to four periods are kept in the form of the first through fourth backward difference of  $z$ . Therefore, all the information needed to estimate the step which could be taken by a generalized method via (5.1) is available. Further, if an estimate of the Lipchitz constant were available it would be possible to adjust the stepsize of the generalized Adams methods to take into account the stiffness (or nonstiffness) of the problem. This can be done for both Adams and BDF methods. Hence, the order and method which yields the most efficient integration of the problem can be determined. After the proper order is chosen, the backward differences of  $z$  can be transformed into the form required by the particular outer integrator in order to provide for starting at the above order.

The efficiency of the outer integrator is obviously dependent upon the outer stepsize. The minimal efficient step would be dependent upon the implementation and the method. In the present implementation, the minimal step was set to three periods for both Adams and BDF's. In actuality, the minimal step should be larger for the BDF's since they are more expensive to integrate with. The prospective efficiency of the chosen Adams stepsize and BDF stepsize could then be compared by dividing each of them by their respective minimal stepsizes. The larger of the two ratios would determine which method is to be used. Further, the outer integrator would not be invoked if the larger ratio was not greater than one.



The strategy used in the detection routine to make the transition decision can be summarized as follows:

1. Using the expression for the local truncation error and the stored backward differences of  $z$ , estimate the stepsizes for all of the prospective starting orders for both Adams and BDF methods.
2. Choose the order for each method which yields the largest stepsize.
3. Adjust the Adams stepsize for stiffness by setting it to the minimum of the present  $H$  and  $\frac{1}{L}$ .
4. Divide each of the stepsizes by their respective minimal stepsizes and choose the method with the larger ratio.
5. If the chosen method has an efficiency ratio larger than one invoke the outer integrator, otherwise return to the detection routine.

#### 5.2. Estimation of the Lipschitz Constant

In this section, ways to estimate the Lipschitz constant are investigated. First, a method which would estimate the constant during the detection routine is presented. Unfortunately, this particular strategy performs very poorly and is included here for the sake of completeness. The second method presented is a possible analytical form of the outer Jacobian which may allow the estimation of the Lipschitz constant directly from a norm or eigenvalue calculation. The last subsection presents the strategy which was implemented to estimate the

constant and its resulting strategy for detecting stiffness and/or reverting to nonoscillatory techniques.

#### 5.2.1. First Strategy

Recall that the detection routine saves up to four periods to determine if the period is smoothly varying. Associated with each of these periods is, of course, a  $g$  value and a  $z$  value at the period's left endpoint. Given this information, it would seem that a crude estimate of the Lipschitz constant might be obtained by looking at the change in two of the  $g$ 's divided by the change in the corresponding  $z$  values. The implementation which tested this strategy actually computed all of the various first order differences of  $g$  and  $z$  and took the maximum quotient to be the estimate of the Lipschitz constant.

This strategy's performance could be described as being, at best, dismal. The reason for this is fairly obvious. The fact that each of the  $z$  values are at different times adds an extra degree of change to the approximation which is not present in the Jacobian. This change in time is not present in using two successive corrector  $g$  values in the outer integrator, as suggested by Gear, but, since the outer integrator has not yet been invoked, no such  $g$ 's exist. Hence, if a means of estimating the Lipschitz constant is desired in the detection routine it would seem that it must be based on something other than the  $g$  and  $z$  values generated by successive periods.

#### 5.2.2. An Analytical Jacobian Possibility

Since the  $g$  and  $z$  values do not look very promising as a basis for

the estimation of the Lipschitz constant in the detection phase, another possibility would be to somehow estimate the Jacobian and then use an eigenvalue computation to give a lower bound for the Lipschitz constant or a norm calculation to give a local upper bound. Numerical differencing is a possibility but, it is quite expensive and presents many implementation difficulties when taken out of the realm of the outer integrator and placed into the detection routine. Therefore, it seems desirable to get an analytical Jacobian estimate. If the system is autonomous then the strategy developed below seems plausible.

$\frac{\partial g}{\partial z}$  evaluated at some point  $z(t)$  is exactly  $\frac{\partial g}{\partial u}$ , where  $u = z(t)$ . So, let  $\hat{y}(0) = u$ ,  $\hat{y}' = f(\hat{y})$ , and let  $T$  be the period with respect to  $t$ . (Recall that the period with respect to the transformed independent variable is 1).  $\hat{y}(t)$  is then the function integrated over one period by the period routine. The  $g$  calculated by the period routine is dependent upon  $T$  and  $u$  and is defined to be

$$g = \hat{y}(u, T(u)) - u.$$

Now consider,

$$\frac{\partial g}{\partial u} = \frac{\partial}{\partial u} \hat{y}(u, T(u)) - I.$$

Carrying out the differentiation yields

$$\frac{\partial g}{\partial u} = \frac{\partial \hat{y}}{\partial u}(u, T) + \frac{d\hat{y}}{dT}(u, T) \frac{\partial T}{\partial u}(u) - I.$$

Note that  $\frac{d\hat{y}}{dT}(u, T) = f(\hat{y}(T))$ . Hence, if  $W(T) = \frac{\partial \hat{y}}{\partial u}(u, T)$  then

$$\frac{\partial g}{\partial u} = W(T) + f(\hat{y}(T)) \frac{\partial T}{\partial u}(u) - I. \quad (5.2)$$

$W(T)$  satisfies

$$\frac{dW}{dt} = \frac{\partial}{\partial u} \dot{y}'(u, t) = \frac{\partial}{\partial u} f(y(t)) = f_y W(t).$$

Therefore,  $W(T)$  can be approximated by integrating  $W'(t)$  while the inner integrator is finding the period. Using backward Euler,  $W'(t)$  can be integrated one step by

$$(I - hf_y)W_{k+1} = W_k, \quad W_0 = I,$$

where  $f_y$  is the inner Jacobian and  $h$  is the inner stepsize.

To find  $\frac{\partial T}{\partial u}$  consider  $P(y) = c^T \dot{y}'$ . By definition,  $P(y(T)) = 0$ . So differentiating with respect to  $u$  and setting the expression equal to 0 yields

$$0 = \frac{\partial c^T}{\partial u} \dot{y}'(T) + c^T f_y(y(T)) \frac{dy}{dT}(T) \frac{\partial T}{\partial u}.$$

Note that

$$\frac{\partial c^T}{\partial u} f(y(T)) = f^T(y(T)) \frac{\partial c}{\partial u}.$$

Therefore,

$$\frac{\partial T}{\partial u} = -[c^T f_y(y(T)) f(y(T))]^{-1} f^T(y(T)) \frac{\partial c}{\partial u}.$$

(5.2) gives an expression for the  $n$  by  $n$  upper left corner of the outer Jacobian. The partials concerning the appended time equation must also be considered. Since the system was assumed to be autonomous, a column of zeros is added to the right. The remaining  $n$  elements in the  $(n+1)$ -st row are the elements of  $\frac{\partial T}{\partial u}$ .

The above strategy avoids the problem of the numerical algorithm

being placed on a nonoscillating integral curve when the initial conditions are perturbed for numerical differencing. It is much easier to adapt to use in both the detection phase and in the oscillatory phase. Two questions which must be considered are whether or not this method can be implemented more efficiently than numerical differencing and whether or not a simple counterpart exists for nonautonomous problems. The present implementation did not use this strategy.

### 5.2.3. Present Strategy

In this section, the strategy used in this implementation to estimate the Lipschitz constant is discussed as is the strategy which results for detecting stiffness and the end of oscillatory behavior.

It was noted in the previous sections that the accumulated  $z$  and  $g$  values in the detection phase did not lend themselves to use in estimating the Lipschitz constant. With this in mind, the strategy implemented ignores the estimation problem the first time periodic behavior is detected, and decides if the outer integrator should be invoked by looking at step estimates based only on truncation errors for Adams and BDF methods. Since Adams methods have smaller error coefficients, the Adams truncation error calculation will lead to larger steps and will be invoked if either BDF or Adams is possible. The successive corrector  $g$ 's generated by the outer integrator can then be used to estimate the Lipschitz constant and test for stiffness. If the outer integrator finds that the function can not yet be integrated properly by oscillatory techniques, the detection routine is recalled.

Notice that in all subsequent calls to the detection routine an estimate of the Lipschitz constant is available. The above description is an outline. Below a precise description of the estimation and general strategy for stiffness detection in the outer integrator is given.

First recall that the decision routine requires an estimate of the Lipschitz constant and estimates of the higher order derivatives. The case of the starting estimate of the Lipschitz constant for the detection routine has been covered above and the estimation of higher order derivatives in the detection routine has been dealt with in the previous section so the remainder of this section will concern the use of the decision routine in the outer integrator.

The outer integrator is a  $P(EC)^m$  predictor-corrector scheme for  $1 \leq m \leq 3$ . In the case of stiff equations, the corrector is a quasi-Newton step. Each of the evaluation steps is a call to the period routine and, when stiff methods are being used, there are additional calls to the period routine to estimate the Jacobian via numerical differencing. Hence, there are many places where stiffness and/or the breakdown of oscillatory behavior must be checked for. The following list indicates situations of interest in the outer integrator. The action in each case will be considered.

1. The corrector converges on the first iteration and the error is satisfactory.
2. The corrector converges on the second or third iteration and the

- error is satisfactory.
3. Same as 2 but the error is not satisfactory.
  4. The corrector fails to converge.
  5. The period algorithm fails on the first or second evaluation step or the period algorithm fails during a Jacobian evaluation.
  6. The period algorithm fails on the third evaluation step.

The Lipschitz constant will be estimated by the difference of the  $g$ 's of two successive corrector iterations divided by the difference of the corresponding  $z$  values. This implies that at least two evaluation steps must be successful in order to test for stiffness. The estimate of the  $(k+1)$ -st derivative for a  $k$ -th order method will be achieved by using the predictor-corrector difference. This can be done after the corrector converges. If the corrector does not converge the last estimate of the derivative generated is used. In the detection phase, the decision routine checks all possible orders for their stepsizes, but here only the present order of Adams and BDF will be checked. A recommended stepsize and a flag indicating whether or not the outer integration will be efficient are returned. Regardless of the case, if stiffness is detected the outer integrator will restart the present step with stiff methods or recall the detection routine depending on the flag. Further, the tests for stiffness will only be made if Adams methods are being used. The

question of how to detect the lack of stiffness has not been addressed here.

In case 1 above, the action taken would be none at all. Indeed, this case implies that everything is as it should be and the outer integrator is allowed to proceed as normal.

In case 2, a successful step has been taken, but the fact that two or three iterations were required implies that something may be wrong. Estimates for the Lipschitz constant and the derivative are available so the decision routine is called. If stiffness is not detected the integrator is allowed to proceed as normal. If stiffness is detected the actions described above are taken.

All the information needed to call the decision routine is available in case 3, so it is called. If the outer integration is to be maintained something must be done with the stepsize. Since the outer integrator has provisions for such a case, it is allowed to reduce the stepsize on its own.

In case 4, the fact that the corrector has failed to converge indicates a bit more serious trouble than case 3. If the flag indicates that the outer integration should be maintained the stepsize is set to the stepsize recommended by the decision routine regardless of whether or not stiffness is detected. The step is then restarted with the appropriate method.

In case 5, an estimate for the Lipschitz constant is not available. So, the stepsize is merely reduced by some factor and the



step is retried. If a period call does not produce a period it may be an indication that the stepsize is too large or that the function has ceased to be periodic. The latter conclusion is reached only if the step has already been reduced to some minimum outer stepsize. In the present implementation, this minimum was one period.

Case 6 differs from case 5 in the fact that an estimate for the Lipschitz constant is available. So, if Adams methods are being used the decision algorithm is called. If it is indicated that the outer integration is to be maintained the step is retried with the recommended stepsize and the appropriate method.

The above actions can be summarized as follows. If Adams methods are being used and trouble is encountered with convergence or single step error control and all of the needed information is available then the decision algorithm is called. If it is decided that the outer integration should be terminated then the detection routine is reinvoked. If the outer integration is to be maintained then the step is retried with the recommended stepsize and the appropriate method. If the period algorithm fails the step is reduced by a constant factor if the needed information is not available or, if BDF's are being used. If Adams methods are being used and the decision routine can be called, after a period failure, then it is called and the appropriate action with the stepsize and method is taken. The lack of periodic behavior is detected when, after a period failure, the stepsize is reduced below some minimum.

This reduction below a minimum stepsize also serves as a second way to detect inefficiency in the use of oscillatory techniques, although it is hoped that the decision routine would be invoked and come to that conclusion long before the reduction of the stepsize to the minimum.

## 6. IMPLEMENTATION

6.1. An Overview

It is desirable to have a control structure which can serve as a basis for a production code. The major characteristic of the control structure, in this case, is that it must provide consistency between the oscillatory integrator and the detection routine. Such a control structure is proposed in this section. First an overview of the oscillatory integrator and the major tasks of the detection routine is given. Then, each of the basic routines will be considered in order to indicate some of the details of the implementation.

The outer integrator used here is a modified version of Petzold's DIFF [4]. The major modifications were made to allow the use of information from the detection routine, the use of the generalized BDF's, and the detection of stiffness. The integrator uses a modified Nordsieck scheme to store past information about  $z$  in vectors of the form

$$\underline{a} = [z, Hg, H^2 g' / 2, \dots, H^k g^{(k-1)} / k!]^T.$$

The predictor takes the form

$$\underline{a}_{-n}(0) = A \underline{a}_{-n-1},$$

where  $A$  is Pascal's triangle with the exception of the first row which contains

$$[1, 1, \alpha_1(r), \dots, \alpha_{k-1}(r)],$$

where  $r = \frac{1}{H}$ . The corrector is of the form

$$\underline{a}_n = \underline{a}_n(0) + \underline{v}\omega.$$

The scalar  $\omega$  is chosen such that  $z(\tau_{n+1}) - z_n = g(\tau_n)$  and  $\underline{v}$  is the conventional corrector vector with the exception of the first component which is a function of  $r$ .

The outer integration proceeds much like an ordinary integrator. The calls to evaluate the derivatives in the ordinary integrator are replaced by calls to a period routine which in turn calls an inner integration routine. Single step errors are controlled by maintaining the stepsize such that the estimate of the local truncation error is kept below some user supplied tolerance. This is, of course, the same type of process used by the detection routine to predict the step to be taken by the generalized methods. The error coefficients in the expression for the local truncation error are functions of the stepsize and if the proper form of these coefficients were used, a nonlinear equation would have to be solved to control the local error. Petzold ignored this fact and used the constant error coefficients of the conventional Adams methods. This implementation does likewise for both the generalized Adams and BDF's.

In the control structure, most of the processes common to the different phases of detection and period finding were isolated and implemented in their own routines. The hierarchy of control is presented in figure 3. A brief description of each of the routines is given here and the next section discusses some of the basic routines in detail.

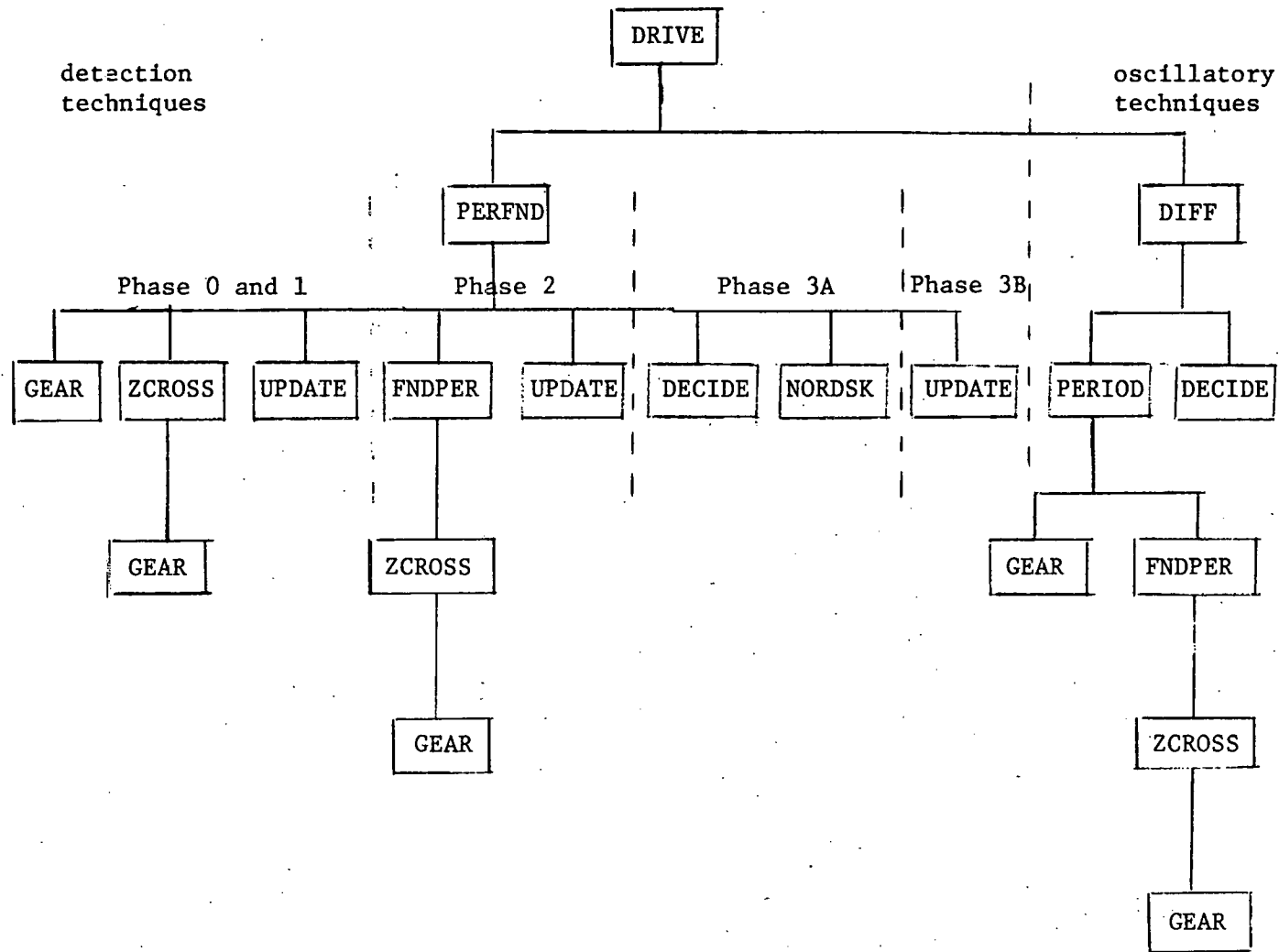


Figure 3. Control Structure of Experimental Code

1. DRIVE is the highest level driver. It is here that all parameters required of the user are set. The detection routine is called from here and upon detection of periodic behavior the outer integrator is called repeatedly until the desired time value is achieved.
2. PERFND is the detection routine. All of the different phases have their control statements here.
3. DIFF is the outer integrator. It performs one step of the integration of the quasi-envelope and then returns to the driver. DIFF calls the period routine and, of course, the decision routine.
4. PERIOD is the period routine. It serves as a driver for the FNDPER routine which does the majority of the period finding work. All of the weights and flags necessary for the period calculation are set in PERIOD and, after the local period is found, PERIOD calculates the values of  $g$  and returns to DIFF.
5. GEAR is the inner integrator and can be viewed as a 'black box' which performs one step of the inner integration and returns to the calling routine. It is a minor change to Hindmarsh's program of the same name [7].
6. ZCROSS is the routine which finds the next positive going zero crossing of the function  $P(y)$ .
7. UPDATE is a routine which does the bookkeeping for the detection

routine. It is responsible for the management of the information of past periods.

8. FNDPER is called from the period routine and from phase 2 of the detection routine. It is a routine which, given a fixed lefthand endpoint, finds the local period or decides that such a period does not exist.
9. DECIDE is the decision routine. It estimates the step which could be taken by the generalized methods of various orders and decides whether or not oscillatory techniques should be used. The routine requires an estimate of the Lipschitz constant and the appropriate higher order derivatives of the quasi-envelope.
10. NORDSK is the routine which converts the backward differences of  $z$  into estimates of the derivatives of  $g$  for use in the Nordsieck history vector.

## 6.2. The Basic Routines

Only three of the routines will be considered in this section. PERFND has, for all practical purposes, been described in the discussion of the detection strategy. The changes to DIFF required no radical alteration of Petzold's implementation. ZCROSS and FNDPER are the routines which do the most work, hence, they shall be considered along with the routine NORDSK which provides the information to DIFF in the proper format.

### 6.2.1. ZCROSS

This routine is the basis for the entire strategy. It is assumed that the  $c$  vector of weights has been set. The setting of these weights, in phase 2 of the detection routine and in the period routine, requires a knowledge of  $y''$ . In the first case, the values will be known from the inner integration which has been done previously. In the second case, an initial call to the inner integrator is made and the higher order starter supplies the estimates. On occasion, the starter is unable to form a higher order starting vector. When this occurs in the period routine,  $y''$  is estimated by the first divided difference of the initial derivative value and the derivative value returned on the initial step. (This estimate has turned out to be quite adequate since the inner integration step is typically fairly small). It is also assumed that the sign of  $P(y)$  is known at the point on  $y$  where the inner integrator is presently situated.

Once the above assumptions have been satisfied, the routine proceeds by calling the inner integrator and evaluating  $P(y)$  repeatedly. If a positive sign change is found the  $y$  values and the higher derivatives are saved to be used in the interpolation to follow. The actual location of the crossing of  $P(y)$  is located by a Newton iteration where each iterate is generated by the above mentioned interpolation. When the crossing is found, ZCROSS stores the information pertaining to the crossing and returns to the calling routine.

Two improvements to the above strategy were implemented. First, it is desirable to ignore oscillations which are smaller in magnitude than



the user specified inner integration tolerance. This being the case, a threshold was set on the negative side of zero such that the value of  $P(y)$  must cross this threshold in the negative direction before a positive to negative sign change was deemed to have taken place.

The second improvement concerns the distance above zero that the positive  $P(y)$  may be when a crossing is detected. It is conceivable that the last negative  $P(y)$  may be very close to zero thus making it a much more suitable base of interpolation than the succeeding positive position. To detect this condition, after each step, the next value of  $P(y)$  is predicted by a first order extrapolation using  $P'(y)$ . This prediction requires knowledge of  $y''$  at the present time. If the inner integrator is using a first order method this information is not contained in the  $y$  vector. The desired information can be obtained by using the predictor-corrector difference used in the inner integrator for error estimation purposes. If the predicted  $P(y)$  is positive the present negative position is used as the base of interpolation. Note, however, that just because the prediction states that the next step will produce a positive  $P(y)$  there is no guarantee that this is the case. Hence, after such a positive crossing has been found, the routine repeatedly calls the inner integrator until an actual positive value of  $P(y)$  is found. This avoids the rather nasty situation of having a false idea of what the sign of  $P(y)$  is on the next call to ZCROSS. (The routine has the unfortunate tendency to find the same crossing that was generated on the last call to ZCROSS when it is given false sign

information).

Of course, it is possible that zero crossings no longer exist past a certain point on a function. This is the case if the function becomes nonperiodic. In order to detect this, the routine checks the value of  $t$  after each inner integration against the final time the user has requested integration to. If  $t$  is passed the final value and the program is in the detection phase then execution is terminated. If the program is in the oscillatory phase this situation implies that a local period does not exist and a flag indicating such is returned to the period routine.

#### 6.2.2. FNDPER

FNDPER is used any time it is desired to find a local period with a fixed left endpoint. This is the case in phase 2 of the detection routine and in the period routine. FNDPER must also have some means of deciding whether or not a local period exists.

FNDPER has the same calling assumptions as ZCROSS since its job is basically to interpret output from ZCROSS. It accomplishes this interpretation in the following way. After each call to ZCROSS, the present and last crossings are compared. This comparison takes the form of evaluating the function  $D(y)$ , which was mentioned in section (3.4) on the period routine. This  $D(y)$  is just a weighted norm of the difference between the values of  $y'$  at the two crossings. The weights currently being used are the inverses of the maximum  $y$  values of the inner integration. If the value of  $D(y)$  is small enough a period is said to

exist and a return to the calling routine occurs.

FNDPER also has the capability of detecting the lack of local periodic behavior. This task is accomplished as follows. Of course, if ZCROSS returns a flag stating that it has gone beyond the final time searching for a crossing, FNDPER decides that a local period does not exist. There are two other means used in this implementation to determine the lack of a period. Recall that the detection routine stores several back crossings for use in finding an initial period or restarting after a local period is not found. This imposes a restriction on the number of crossings accepted per period. Since it is not likely that this number will change that drastically over the integration of the function, it seems reasonable that if no local period is found within that number of crossings, during the oscillatory phase, to conclude that something is amiss and return an indication that local periodic behavior of the type originally found and presently expected no longer exists.

The second technique is that of an expansion test. In the detection routine, successive periods are always compared. FNDPER is called from phase 2 where a period of approximately the same size as that which was just generated in phase 1 is expected. Likewise, in the oscillatory phase, an estimate for the size of the period to be found exists. It is carried along as the second component of the Nordsieck vector of the time equation. That is, the period is the derivative of the time component with respect to the normalized independent variable.

FNDPER uses these estimates by comparing the difference between the left endpoint and the location of each crossing generated that does not yield a small enough  $D(y)$ . This difference is allowed an expansion factor (presently 1.1) before it is concluded that the period is expanding too rapidly and the proper type of periodic behavior no longer exists.

### 6.2.3. NORDSK

The Nordsieck vector used in the outer integrator contains derivatives of  $g$  which are not known when the decision is made to invoke oscillatory techniques. Hence, some routine is needed to prepare the data required by DIFF in order to provide a higher order start. NORDSK is the routine which performs this task.

When entering DIFF, the outer integrator must be poised at the left endpoint of a local period which was just integrated through to supply starting  $g$  values. Further, if a  $k$ -th order start is to be accomplished up to the  $(k-1)$ -st derivative of  $g$  must be placed in the Nordsieck history vector. Also, note that the  $k$ -th derivative of  $g$  must be known for the step to be estimated. The detection routine has up to five values of  $z$  to work with. The fourth value of  $z$  is the point where all of the estimation must take place.

The solution of the above problem is, of course, to pass an interpolating polynomial through the  $z$  values which interpolates  $g$  at the fourth value of  $z$ . This is consistent with the strategy of using the  $i$ -th backward difference of  $z$  as an estimate of the derivative needed at the fourth  $z$  to predict the step. For, if the interpolating

polynomial was of order  $i - 1$  then the  $i$ -th derivative would indeed be the  $i$ -th backward difference. NORDSK uses this fact to approximate the various derivatives once an order is decided upon. If an  $i$ -th order start is accepted it is implicit that estimates of the derivatives of  $g$  up to the  $i$ -th exist. Hence, up to  $i$ -th backward difference of  $z$  can be transformed into a more accurate estimate of the  $(i-1)$ -st derivative of  $g$  by evaluating the proper derivative of the interpolating polynomial at the fourth  $z$ . The resulting estimates are then scaled by the predicted stepsize and placed in the Nordsieck vector for use by DIFF.

## 7. EXPERIMENTAL RESULTS

In this section, the results of the numerical testing are reported. The tests ranged from a simple sine-cosine system to the phase sensitive example discussed earlier. All examples were run in both the synchronized and nonsynchronized mode but, the difference will be reported only when significant.

### 7.1. Test 1

Regardless of the sophistication of an integrator, it should be able to handle trivial problems. The simplest autonomous problem in oscillations is the sine-cosine system of the form

$$\begin{aligned}y_1' &= \lambda y_2, & y_1(0) &= 0 \\y_2' &= -\lambda y_1, & y_2(0) &= 1.\end{aligned}$$

This problem is certainly not a stiff oscillatory problem, so it would be extremely unfortunate if the routine presented here decided to invoke the generalized BDF's to solve it.

The problem was integrated using an inner integration tolerance of  $10^{*-11}$  and an outer tolerance of  $10^{*-3}$ . Nonstiff methods were used in the inner integration. For both the synchronized and nonsynchronized case, the integrator took one outer step consisting of approximately 2386 periods and passed a smooth quasi-envelope through the solution.

### 7.2. Test 2

Petzold considered a contrived stiff oscillatory problem in her thesis to show the behavior of the Adams methods on such a problem. The problem is considered here in order to test the stiffness detection

routine on its ability to detect rather obvious stiffness.

The problem took the form

$$r' = \lambda(1-r^2), \quad r(0) = 1,$$

$$\theta' = -\mu, \quad \theta(0) = 0$$

in polar coordinates and

$$y_1' = \lambda y_1^\Psi + \mu y_2,$$

$$y_2' = \lambda y_2^\Psi - \mu y_1,$$

$$y_1(0) = 1, \quad y_2(0) = 0,$$

where

$$\Psi = (1 - y_1^2 - y_2^2)(y_1^2 + y_2^2)^{-1/2}$$

in rectangular coordinates.

Notice that  $\lambda$  regulates the stiffness. That is, the larger  $\lambda$  is the more stiff the problem.  $\mu$  regulates the frequency of oscillation about the unit circle. Petzold showed that the Adams method had trouble for both  $\lambda = 1500$  and  $\lambda = 10000$ . The problem was run using the same tolerances as test 1 and  $\lambda = 10000$ .

The detection routine had no trouble detecting the periodic behavior. The decision routine recommended a step of 967 periods in the synchronized case. (A fraction more in the nonsynchronized case). The first outer step succeeded, but when the integrator attempted to increase the step with the Adams method the decision routine called for the restarting of the outer integration with stiff techniques. (The decision routine was called after a failed period evaluation).

The results with the generalized BDF's were amazing. The outer integrator passed a constant quasi-envelope through points from  $t = 0$  to  $t = 643$  in just 5 outer integrations, including the aborted second step with the Adams method. The stepsize reached a maximum of 78607 periods!

### 7.3. Test 3

A less contrived example of an oscillating problem, which is difficult to solve with the Adams methods, is the Van der Pol oscillator problem.

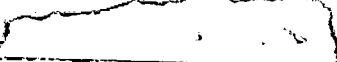
$$\begin{aligned}y_1' &= y_2 \\y_2' &= -\lambda y_1 + 2\lambda\mu y_2(1 - y_1^2) \\y_1(0) &= 2, y_2(0) = 0.\end{aligned}$$

The magnitude of  $\mu$  is the factor which determines how difficult the problem is to solve. Since this problem is already known to be periodic a detection routine is not strictly necessary. This allows comparison of the ability of the respective period finding routines. It was found that the routine described as the second method of finding a local period, in an earlier section, was unable to operate properly on the Van der Pol problem for any  $\mu$  slightly larger than .3.

The problem was considered with  $\mu = .5$ . Again the initial conditions place the solution almost on the limit cycle yielding the expectation of an almost constant quasi-envelope. While the detection routine had no trouble detecting periodic behavior, the decision routine required two calls before it could generate a step estimate larger than three periods. This is due to the fact that when the periodic behavior



is found immediately, a first order start is all there is information for, hence, if the problem is a bit more difficult more calls may be required to generate the necessary information for a higher order start. Once the outer integrator was called, two successful steps were taken. After the integrator attempted to increase the stepsize, stiffness was detected. This time the detection routine was recalled since the decision routine decided that the present information did not justify a large stepsize with the BDF's. After the detection routine recalled the decision routine a step of 1249 periods was recommended and was taken successfully. Indeed, the outer integrator passed a constant quasi-envelope through points at  $t = .3456$  to  $t = 41.95$  in this step.

The behavior of the decision routine recommending restarting of the detection routine and then on the next call recommending a large stepsize can probably be explained as follows. It seems that the outer integrator must have built up fairly large errors in its estimates of the derivatives of  therefore, the stepsize predicted by the decision routine is suitably small. Upon receiving more accurate information from the detection routine's integration over a local period of the function, it was able to see the true capability of the BDF's.

#### 7.4. Test 4

Gear [5] investigated a numerical example which was not periodic until after an initial phase to test the detection strategy. The example was partially based on a Van der Pol problem and took the following form.

$$\underline{u} = Q\underline{y}$$

$$u'_1 = u_2$$

$$u'_2 = -(u_1 - u_3) + 2(u_3 - (u_1 - u_3)^2)u_2$$

$$u'_3 = -10^{-3}(u_3 - 1)$$

$$u'_4 = 10^{-3} \sin 10^{-3} t.$$

The system integrated was

$$y' = Q^{-1} \underline{u}', \quad \underline{u}(0) = \underline{0},$$

$$Q = 1/2 \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix}$$

(Note:  $Q = Q^{-1}$  ).

The present implementation was run with the same error tolerances listed for test 1. The fact that the third and fourth component of the  $u$  system have analytical solutions allowed the absolute error in these components to be evaluated after each outer step. Using the above mentioned error tolerance and error calculation, it is possible to get a better idea of the behavior of this system than was possible with the untuned version of Gear.

Periodic behavior was detected much earlier than the  $t = 154$  reported by Gear. This was due to the fact that a much stricter inner integration tolerance was used. Initial periodic behavior was found at  $t = 50$ , but all of this behavior appeared to be fairly localized or not smooth enough since the detection routine had to be recalled almost immediately after the outer integrator was invoked due to the fact that

the corrector diverged. Stiffness was detected at this point.

From the point stiffness was detected until  $t = 94$ , the decision routine was recalled repeatedly but, it was unable to generate an adequate starting stepsize. At  $t = 94$  the detection routine was unable to generate successive periods which did not violate the expansion test. (Expansion factors of up to 12.0 were violated). This behavior is expected when the graphs of the solution presented in Gear [5] are consulted. There is an area of rapid expansion of both the period and the amplitude of the solution from approximately  $t = 90$  until  $t = 140$ . This was directly verified by the detection routine which did not call the decision routine again until  $t = 139$ . The outer integrator was finally called at  $t = 164$  with the recommendation that a second order BDF be used.

The outer integrator remained in control from  $t = 164$  until the end of the integration at  $t = 10105$ . The step used ranged from the initial 3 to a maximum of 69 periods. The absolute error in the fourth component of  $u$  ranged from  $0.6 * 10^{*-7}$  to  $0.1 * 10^{*-2}$ . A total of 48 outer integration steps were required. In all, 169 calls to the period routine were made.

The nonsynchronized case was slightly superior in performance. Of course, the detection phase proceeded in exactly the same manner, but, only 46 outer integration steps were required to go to  $t = 10357$ . 139 period routine calls were required. The error in the fourth component was comparable to the synchronized case. The step went from 3 to 66

periods but, it increased faster than the synchronized case, allowing the improvement in the number of outer steps. A final interesting statistic is the fact that the nonsynchronized mode required only 92839 evaluations of the systems derivatives while the synchronized case required well over 98000.

#### 7.5. Test 5

As a last example, the function which was cited earlier as being phase sensitive and therefore only appropriate to be integrated in the synchronized mode is considered. The function is

$$y' = -y + \sin \lambda t,$$

$$y(0) = -\frac{\lambda}{1+\lambda^2}.$$

$\lambda$  was taken to be 1000 in this example. Note that the initial conditions place the solution on the limit cycle so the detection routine should have no trouble detecting periodic behavior. (This was not the case for tests run with initial conditions off the limit cycle. In that case, nearly periodic behavior does not exist in a transient region).

The initial stepsize recommended by the decision routine was 1244 periods. Two successful steps were made with this size. However, when the integrator tried an excessively large step of 3180 periods the stiffness detection mechanism forced a restart with stiff methods. This was done solely because of the fact that the integrator wished to take a large step. The Lipschitz estimate was extremely small (.00626). The stiff method took one step at 526 periods and then immediately increased

the stepsize back to the desired 3180. A total of 4 outer integration steps were taken to go from  $t = 0$  to  $t = 22$ . 9 period routine calls were needed. The analytical solution was used to determine the error at the end of the integration to be  $.1 * 10^{*-7}$ .

Things were not as nice with the nonsynchronized mode. The detection routine was recalled several times in the course of the integration. There were an amazing number of period failures, as predicted in the earlier discussion of this function. A total of 66 outer steps were required to cover the same interval as the synchronized case. 406 period evaluations were made. This result confirms the necessity of the synchronized mode in oscillations which are sensitive to phase changes, such as driving term oscillations.

## 8. CONCLUSION

In this section, a brief summary of the topics presented is given along with an indication of some of the topics which are in need of further development.

First, a technique for integrating highly oscillatory equations was reviewed. This technique required the algorithmic definition of local nearly periodic behavior so three possible definitions were considered on the basis of their efficiency and possibility of generalization to use in a detection strategy.

The detection strategy which resulted from the third definition of nearly periodic behavior was considered in detail. This detection strategy was then enhanced to allow for the selection of an initial stepsize and method for the oscillatory integrator. Along with this selection mechanism, a strategy was proposed to determine the efficiency of the use of oscillatory techniques and the detection routine was designed such that the oscillatory techniques would not be invoked unless deemed efficient.

The above strategy for the calculation of efficiency and initial method selection was adapted for use in the oscillatory integrator to allow the detection of stiffness and loss of efficiency in the use of oscillatory techniques. Provisions were made to change to appropriate methods in the case of stiffness or to revert to the detection routine in the case of inefficiency.

A control structure was then proposed for use as a basis of an

experimental code and possibly a future production code. The control structure provided for the isolation into separate routines of all processes common to the detection and oscillatory phases of the integration. Numerical results were presented and indicated that a production code based upon the above strategy and control structure was feasible.

The major area of immediate interest is the efficient approximation of the outer Jacobian. The analytical formulation presented here is a possible starting point. Presently, a strategy based on a mixture of numerical differencing and analytical approximation which avoids the problem of the nonexistence of nearly periodic behavior on nearby integral curves is under investigation.

## LIST OF REFERENCES

- [1] Mace, D. and L. H. Thomas, An extrapolation method for stepping the calculations of the orbit of an artificial satellite several revolutions ahead at a time, Astronomical Journal 65 (5), #1280, June 1960.
- [2] Graff, O. F., Methods of orbit computation with multirevolution steps, Applied Mechanics Research Laboratory Report AMRL 1063, University of Texas, Austin, Texas, 1973.
- [3] Graff, O. F. and D. G. Bettis, Modified multirevolution integration methods for satellite orbit computation, Celestial Mechanics 11, 1975, 443-448.
- [4] Petzold, L. R., An efficient numerical method for highly oscillatory differential equations, Report UIUCDCS-R-78-933, Dept. Comp. Sci., Univ. Illinois at Urbana-Champaign, Illinois, August 1978.
- [5] Gear, C. W., Automatic detection and treatment of oscillatory and/or stiff differential equations, Report UIUCDCS-R-80-1019, Dept. Comp. Sci., Univ. Illinois at Urbana-Champaign, Illinois, June 1980.
- [6] Gear, C. W., Runge-Kutta starters for multistep methods, Report UIUCDCS-R-78-938, Dept. Comp. Sci., Univ. Illinois at Urbana-Champaign, Illinois, September 1978.
- [7] Hindmarsh, A. C., GEAR: Ordinary differential equation solver, UCID-30001 REV. 3, Lawrence Livermore Laboratory, Livermore, California, December 1974.



<b>BIBLIOGRAPHIC DATA SHEET</b>	1. Report No. UIUCDCS-R-80-1045	2.	3. Recipient's Accession No.
4. Title and Subtitle DETECTION AND INTEGRATION OF OSCILLATORY DIFFERENTIAL EQUATIONS WITH INITIAL STEPSIZE, ORDER AND METHOD SELECTION		5. Report Date December 1980	
7. Author(s) K. A. Gallivan .		8. Performing Organization Rept. No. R-80-1045	
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, IL 61801		10. Project/Task/Work Unit No.	
		11. Contract/Grant No. DE-AC02-76ERO2383.A003	
12. Sponsoring Organization Name and Address U.S. Department of Energy Chicago Operations Office Argonne, IL 60439		13. Type of Report & Period Covered M.S. Thesis	
		14.	
15. Supplementary Notes			
16. Abstracts Within any general class of problems there typically exist subclasses possessed of characteristics which can be exploited to create techniques more efficient than general methods applied to these subclasses. Two such subclasses of initial value problems in ordinary differential equations are stiff and oscillatory problems. Indeed, the subclass of oscillatory problems can be further refined into stiff and nonstiff oscillatory problems. This refinement will be discussed in detail. This paper addresses the problem of developing a method of detection for nonstiff and stiff oscillatory behavior in initial value problems. Given this method of detection a control structure is proposed upon which a production code could be based. An experimental code using this control structure will be described and results of numerical tests will be presented.			
17. Key Words and Document Analysis. 17a. Descriptors  ordinary differential equations oscillatory problems initial value problems detection method			
17b. Identifiers/Open-Ended Terms			
17c. COSATI Field/Group			
18. Availability Statement unlimited		19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 71
		20. Security Class (This Page) UNCLASSIFIED	22. Price