DYNAMIC SIMULATION OF

LARGE STIFF SYSTEMS

DYNAMIC SIMULATION
OF LARGE STIFF SYSTEMS
IN A MODULAR SIMULATION FRAMEWORK

by

JAMES ROBERT BARNEY, B.A.Sc. M.A.Sc.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

May 1975

DOCTOR OF PHILOSOPHY (1975) McMASTER UNIVERSITY

(Chemical Engineering) Hamilton, Ontario

TITLE: Dynamic Simulation of Large Stiff Systems in a Modular Simulation Framework

AUTHOR: James Robert Barney, B.A.Sc. (University of Waterloo)

M.A.Sc. (University of Waterloo)

SUPERVISOR: Professor A.I. Johnson

NUMBER OF PAGES: xiv, 362

ABSTRACT

"DYNSYS" is a digital simulation package for modelling the dynamic behaviour and automatic control of complex industrial systems. An acronym for Dynamic Systems Simulator, it was developed in the late 1960s in the chemical engineering department at McMaster University. The underlying principle of "DYNSYS" is that of modularity; i.e., the user assembles mathematical models of process units and control devices to build the process whose transient behaviour is to be studied. A fundamental aspect of dynamic simulation is the numerical solution of ordinary differential equations (o.d.e.s). The original version of "DYNSYS" used a third order Adams-Moulton-Shell routine; however, this is not sufficient to handle stiff systems, i.e., systems where the time constants differ greatly in magnitude. In chemical engineering, stiff o.d.e.s occur widely in reaction kinetics and to some extent in multistage systems.

Conventional numerical techniques are restricted by stability to using a very small step size resulting in large computer times. There have been many new numerical techniques published in the recent literature directed at

the efficient numerical solution of systems of stiff o.d.e.s. A literature survey of these has been made.

Numerical testing of several methods indicated Gear's method to be superior. It is a variable order, variable step, linear, multistep method.

Most stiff techniques are implicit and require a technique such as Newton-Raphson iteration to converge. Each iteration involves the solution of a system of linear algebraic equations (usually sparse) equal in size to the number of o.d.e.s. For a large stiff system, this requires considerable computer time. Various sparse linear equation solvers have been evaluated and that of Bending and Hutchison appears to be the most efficient. Their routine stores and operates on only the nonzero elements of the equations. When the equations are solved for the first time, a string of integers called the "operator list" is created which stores the particular solution process by Gaussian elimination. If the system is re-solved using the operator list, the amount of computer time required is greatly diminished. If the zero elements remain zero and the nonzero elements change, the same "operator list" can be used to solve the new system. This is essentially what occurs during numerical integration. The operator list could be set up on the first integration step and used on later steps to solve each new linear system.

Gear's integration algorithm in conjunction with the Bending-Hutchison linear equation solve has been implemented into DYNSYS version 2.0. An option for stiff systems with tridiagonal Jacobian matrix is also included. The procedure for writing modules is outlined.

Four small examples are presented to illustrate the new executive:

(1) The level control of a stirred tank system (nonstiff) with time delay.

(2) A network of 15 stirred tank reactors, stiff and nonstiff, 2 o.d.e.s per reactor.

(3) A tubular reactor with 222 stiff o.d.e.s resulting from the discretization of the partial differential equations.

(4) A tubular reactor with 49 stiff o.d.e.s with tridiagonal Jacobian matrix.

A simulation of a fictitious chemical plant proposed by Williams and Otto is also described.

ACKNOWLEDGEMENTS

The study described herein was supervised by Dr. A.I. Johnson, Dean of Engineering Science at The University of Western Ontario. The author is grateful to him and to many of his associates for their assistance.

Discussion and correspondence with D.M. Brandon of Control Data Corporation was very helpful.

The author was aided by a National Research Council Scholarship, a Shell Canada Engineering Fellowship and a grant from Control Data Corporation.

The computational work was done on the CDC 6400 at McMaster University and the CDC CYBER 73 at The University of Western Ontario.

Thanks are also due to J. Pulido for the distillation column work in the Williams-Otto plant, G. Lusk for the drawings and Miss T. Wellon for the typing.

TABLE OF CONTENTS

INDEX OF FIGURES

INDEX OF TABLES

1. INTRODUCTION TO DYNAMIC SIMULATION

Dynamic simulation in this thesis will refer to the study, via digital computer simulation, of the dynamic behaviour of a process which is changing with time. In our context, the process will be all or part of a chemical plant.

There are many analysis and design problems that may require a flexible dynamic model for their solution, for example:

(1) Difficulties in start up or shut down of a process.

(2) Control strategies including finding suitable controller settings.

(3) The influence of frequent disturbances or fluctuations in the process.

(4) The response of the plant to equipment failure and testing of corrective action.

(5) Operating strategies for multiproduct plants.

In addition there may be some steady state process problems whose solution can be obtained by driving a

dynamic model of the process to the steady state (thus avoiding numerical instabilities in a steady state iterative process, for example).

Various executive computer programs have been developed to handle the information transfer and calculations for performing a dynamic simulation.

1.1 Equation-Oriented Approach

The equation-oriented approach was developed first. Here, the user would write, possibly in some coded manner, all the algebraic and differential equations describing his process and the executive would then solve all the equations simultaneously over a period of time. There have been over thirty packages developed since 1955 (Franks, 1967). Table 1.1 lists some of the more common, recently developed programs: MIMIC (Northcott, 1967), CSMP (CSMP, 1967) and IMP (Brandon, 1972).

Table 1.1: Equation-Oriented Executive Programs For Dynamic Simulation

Acronym	Date	Full Name	Institution Where Developed
MIMIC	1965		Wright-Patterson Air Force Base, Ohio
CSMP	1967	Continuous System Modeling Program	IBM
IMP	1972	Implicit Solution Software System	University of Connecticut

1.2 Modular Approach

In the past few years, several executive programs have been developed, mostly by chemical engineers, which are more problem-oriented. A summary of these appears in Table 1.2: SWAPSO (Utsumi, 1969), KARDAZ (Brambilla *et al.*, 1971; Kardasz, 1969; Kardasz and Molnar, 1971, 1974), DYNSYS (Bobrow, Johnson and Ponton, 1970, 1971), FLEX (Shern and Petty, 1970), PRODYC (Ingels and Motard, 1970), REMUS (Ham, 1969), EARLYBIRD (Weaver, 1974), ACME (Loibl, Camp and Wilkins, 1973), DYFLO (Franks, 1972a) and OSUSIM (Koenig, 1972). More information on these packages appears in Section 3.4.

These problem-oriented programs use variations of what is known as the modular approach. A real plant is made up of processing units such as reactors, heat exchangers, compressors, etc. with connecting lines of material flow. The flows, temperatures, concentrations and other variables in the process may be measured and, based on these measurements, controllers can activate control devices in an attempt to achieve desired conditions in the plant. Figure 1.1 illustrates a very simple mixing operation which is made up of a processing unit with controllers and control valves in a process flow diagram. Each of the interconnected units can be described by mathematical relationships relating inputs and outputs.

Table 1.2: Problem-Oriented Executive Programs For Dynamic Simulation

Acronym	Date	Full Name	Institution Where Developed
SWAPSO	1969	Stone and Webster All Purpose Simulator and Optimizer	Stone and Webster
KARDAZ	1969	Kardasz Program	University of Pisa
DYNSYS	1970	Dynamic Systems Simulator	McMaster University
FLEX	1970		Procter and Gamble
PRODYC	1970	A System for Simulating Chemical Process Dynamics and Control	University of Houston
BEMUS	1970	Routine For Executive Multi-Unit Simulation	University of Pennsylvania
EARLYBIRD	1971		Tulane University
ACME	1972	Analyzer For Computer Modeling of Chemical Engineering Processes	University of Detroit
DYFLO	1972		DuPont
OSUSIM	1972	Ohio State University Simulator	Ohio State University

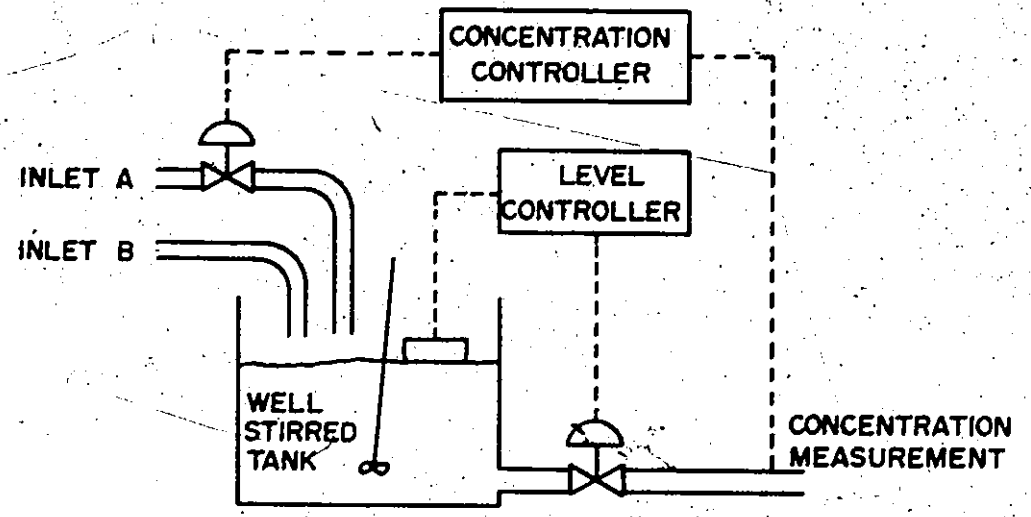


FIGURE 1.1 : PROCESS FLOW DIAGRAM

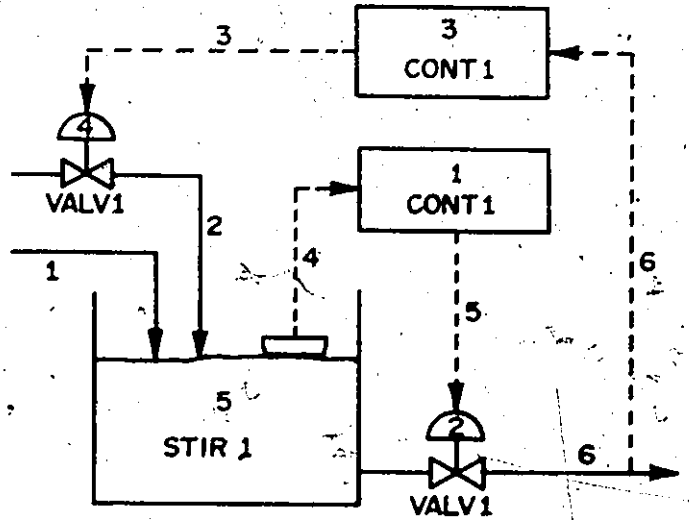


FIGURE 1.2 : DYNAMIC INFORMATION FLOW DIAGRAM

The assembly of the set of mathematical equations for simulating one of these process units, including controllers and control devices, in a computer program subroutine is called a unit computation or computation module.

Just as a chemical plant is made up of assemblies of the physical units, a mathematical model of a plant can be assembled as a network of modules, among which information will flow in a manner analogous to the material flow or control signals in the real plant. Figure 1.2 is a dynamic information flow diagram that corresponds to Figure 1.1. It can be seen that there is a very close similarity between the process flow diagram and the dynamic information flow diagram.

This building-block approach is very convenient for simulating systems which are very modular, i.e., which have many different pieces of physical equipment and where it is desired to study different combinations or configurations of these equipments and their effect on the dynamic behaviour. It is not as efficient computationally as the equation-oriented approach since there is more information transfer; however, it does allow the user to visualize his process more easily since each piece of equipment is usually represented by a corresponding module.

Several modules of different levels of sophistication can be developed for the same equipment. As a library of equipment and control modules becomes available, they may be readily used for the simulation of new plants, provided that the modules have a reasonable level of generality. This reduces the programming effort of new plant studies. A new simulation would require perhaps one third of the modules to be created, while the remaining two thirds could be taken from the library.

The modular approach deals with the real variables of the process, rather than transformed variables, and furthermore, the modules may be quite nonlinear in behaviour. This should encourage control studies by design and process engineers, even if they are unfamiliar with modern control systems theory and terminology.

1.3 Thesis Objectives

The objectives of this thesis are:

(1) to critically evaluate numerical techniques for the solution of systems of stiff ordinary differential equations such as those encountered in the analysis and design of complex chemical processes and to develop improved techniques for these situations.

(2) to critically analyze the DYNSYS package which is under development and to improve its approach for analyzing and designing complex systems.

(3) to demonstrate the application of an improved DYNSYS to several examples including a dynamic simulation of a realistic chemical process and to make recommendations for a library of unit computations.

2. NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS

A fundamental part of dynamic simulation is the numerical solution of ordinary differential equations (o.d.e.s). Partial differential equations can be broken down into o.d.e.s by discretizing one or more of the independent variables. Higher order o.d.e.s can usually be broken down into a set of first order equations. Thus we are primarily concerned with the numerical solution of a set of first order ordinary differential equations. These are usually initial value problems where each of the independent variables is specified at some initial time value (normally $t = 0$), rather than boundary value problems where the independent variables are not all specified at the same time value. There are a large number of numerical integration techniques for solving these equations. This chapter discusses some aspects of these techniques.

2.1 Introduction

We are considering the system:

$$\dot{y} = \frac{dy}{dt} = f(t,y) \qquad (2.1.1)$$

with initial condition:

$$y(t_o) = y_o \qquad (2.1.2)$$

where y may be scalar or vector.

$f(t,y)$ represents any function of the dependent variable y and the independent variable t.

t_o represents the initial value of t and

y_o is the initial value of $y(t)$ at $t = t_o$.

By a solution to the above system, we mean a curve in the $y(t)$ versus t domain which passes through the point (t_o, y_o) and which satisfies (2.1.1). By a numerical solution, we mean a discrete set of values of $y(t)$ called $\{y_n\}$ corresponding to a discrete set of t values called $\{t_n\}$ which approximate the equivalent continuous $y(t)$ versus t curve.

To obtain these discrete sequences, we consider a finite set of points $\{t_n\}$ which form a grid along the t coordinate. Each point in the sequence will be related to the previous point by the relationship:

$$t_{n+1} = t_n + h_n \quad n = 0, 1, \ldots, N \quad (2.1.3)$$

where h_n is the grid spacing.

Numerous techniques have been developed to solve o.d.e.s, the two principal types being single and multiple-step methods.

2.2 Existence and Uniqueness Theorem

Before we attempt to find a solution to (2.1.1) and (2.1.2), we must be assured that a unique solution does exist. The following two conditions are necessary and sufficient for a unique solution to exist in the time interval $[a,b]$ (Henrici, 1962)

1. $f(t,y)$ is defined and continuous in the interval $a \leq t \leq b$, $-\infty < y < \infty$ where a and b are finite.

2. There exists a Lipschitz constant L such that for any $t\epsilon[a,b]$ and any two numbers u and v:

$$|f(t,u) - f(t,v)| \leq L\,|u - v| \qquad (2.2.1)$$

In the nonscalar case, vector norms replace absolute values.

The author is not aware of any equations resulting from chemical engineering applications where these conditions are not satisfied.

2.3 Single-Step Methods

Single-step methods do not require any information prior to (t_n, y_n) to calculate y_{n+1}. The most common single-step methods are the Euler and Runge-Kutta types. The simple Euler method is:

$$y_{n+1} = y_n + h\, f(t_n, y_n) \qquad (2.3.1)$$

Runge-Kutta methods use evaluations of $f(t,y)$ within the interval (t_n, y_n) and (t_{n+1}, y_{n+1}).

The general single-step equation is:

$$y_{n+1} = y_n + \sum_{i=1}^{v} w_i\, k_i \qquad (2.3.2)$$

where w_i are weighting coefficients

v is the number of $f(t,y)$ substitutions

and $k_i = h\, f(t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij}\, k_j)$

$$c_1 = 0, \quad i = 1, 2, \ldots, v \qquad (2.3.3)$$

A typical fourth order formula is given by:

$$y_{n+1} = y_n + \frac{1}{6}\,[k_1 + 2k_2 + 2k_3 + k_4]$$

where $k_1 = h\, f(t_n, y_n)$

$$k_2 = h\, f\left(t_n + \frac{1}{2} h,\ y_n + \frac{1}{2} k_1\right) \qquad (2.3.4)$$

$$k_3 = h\, f\left(t_n + \frac{1}{2} h,\ y_n + \frac{1}{2} k_2\right)$$

$$k_4 = h\, f(t_n + h,\ y_n + k_3)$$

2.4 Multiple-Step Methods

A multiple-step or multistep method is one which uses information prior to y_n to calculate y_{n+1}. The general linear, k-step differential-difference equation with constant coefficients is:

$$y_{n+1} = \sum_{i=1}^{k} \alpha_i\, y_{n+1-i} + h \sum_{i=0}^{k} \beta_i\, \dot{y}_{n+1-i} \qquad (2.4.1)$$

If $\beta_o = 0$, the resulting equation is called an explicit or predictor formula. However, if $\beta_o \neq 0$, the resulting equation is referred to as an implicit or corrector equation.

The normal mode of usage is to use an explicit form of (2.4.1) to get a first approximation to y_{n+1} (the predicted value) and then use an implicit form of (2.4.1) to improve on this value with $\dot{y}_{n+1}$ evaluated using the predicted value of y_{n+1}. The process may be repeated using the new value of $\dot{y}_{n+1}$, calculated from the new

value of y_{n+1}. This may be done several times until y_{n+1} converges to a fixed value. The term PECE means a predictor step followed by a derivative evaluation, then a corrector step and another derivative evaluation. For $PE(CE)^s$ the latter step is repeated s times. These are also known as predictor-corrector methods although a single step method can also be a predictor-corrector.

A commonly used combination is the Adams-Bashforth predictor with the Adams-Moulton corrector equation. The third order equations are:

$$\text{PREDICTOR:} \quad y_{n+1} = y_n + \frac{h}{12}\left[23y_n - 16y_{n-1} + 5y_{n-2}\right]$$

$$\text{CORRECTOR:} \quad y_{n+1} = y_n + \frac{h}{12}\left[5y_{n+1} + 8y_n - y_{n-1}\right] \qquad (2.4.2)$$

2.5 Stability

In the numerical solution of an o.d.e., a sequence of approximations y_n to the true solution $y(t_n)$ is generated. The stability of a numerical method refers to the behaviour of the difference or accumulated error $y(t_n)-y_n$ as n becomes large.

Stability analysis is usually performed on the scalar equation:

$$\frac{dy}{dt} = \lambda y \qquad (2.5.1)$$

$$y(0) = 1 \qquad (2.5.2)$$

with analytical solution:

$$y(t) = \exp(\lambda t) \qquad (2.5.3)$$

For the numerical solution, if h is constant

$$t = nh \ , \ y_n = e^{nh\lambda} \quad n = 0, 1, 2, \ldots \qquad (2.5.4)$$

or

$$y_{n+1} = e^{h\lambda} y_n \qquad (2.5.5)$$

2.5.1 Stability Of Multistep Methods

Applying the general linear multistep method (2.4.1), we get the characteristic equation:

$$\mu^k - \sum_{i=1}^{k} \alpha_i \mu^{k-i} - h\lambda \sum_{i=0}^{k} \beta_i \mu^{k-i} = 0 \quad (2.5.6)$$

which has k characteristic roots μ_i, $i = 1, 2, \ldots, k$.

The numerical solution is thus:

$$y_n = \sum_{i=1}^{k} d_i \mu_i^n \quad (2.5.7)$$

where $d_1, \ldots, d_k$ are constants determined by the initial conditions.

One of the characteristic roots approximates the Taylor Series expansion of the true solution $y = \exp(\lambda t)$ with a truncation error corresponding to the order p of the method. If we let this root be μ_1, then as $h \to 0$

$$\mu_1 = \exp(h\lambda) + O(h^{p+1}) \quad (2.5.8)$$

This root, called the principal root, is the root which we wish to be represented in the numerical solution, since μ_1^n approximates $\exp(nh\lambda)$. The other $k-1$ roots are extraneous and are as a result of the use of a difference equation of degree k to represent a first order

differential equation. The extraneous roots have no relation to the exact solution, but nevertheless are unavoidable.

The characteristic roots of (2.5.7) are the same as those of the difference equation for the error:

$$\varepsilon_n = y_n - y(t_n) \qquad (2.5.9)$$

i.e.
$$\varepsilon_n = \sum_{i=1}^{k} c_i \mu_i^n \qquad (2.5.10)$$

For a valid numerical solution we require that ε_n not grow with n. A linear multistep method is called absolutely stable if $|\mu_i| \leq 1$, $i = 2, \ldots, k$.

The critical problems of numerical stability in o.d.e.s are associated with inherently stable o.d.e.s $(Re(\lambda) < 0)$ in which absolute stability is the important factor.

The value of $h\lambda$ for which $|\mu_i| = 1$ and for which a small increase in $h\lambda$ makes $|\mu_i| > 1$ is called the general stability boundary. Any method with a finite stability boundary is called conditionally stable, whereas any method with an infinite general stability boundary is called unconditionally stable or A-stable. The recent literature contains many new stability definitions (Lambert, 1973; Widlund, 1967; Cryer, 1973).

Most conventional methods have stability boundaries roughly like those in Figure 2.1. A-stability would require absolute stability for the entire left hand plane (Dahlquist, 1963).

Dahlquist (1963) has proven several important theorems for multistep methods:

(1) An explicit linear multistep method cannot be A-stable.

(2) The order of an A-stable implicit linear multistep method cannot exceed two.

(3) The second order A-stable implicit linear multistep method with the smallest error constant is the trapezoidal rule:

$$y_{n+1} = y_n + \frac{h}{2}(\dot{y}_n + \dot{y}_{n+1}) \qquad (2.5.11)$$

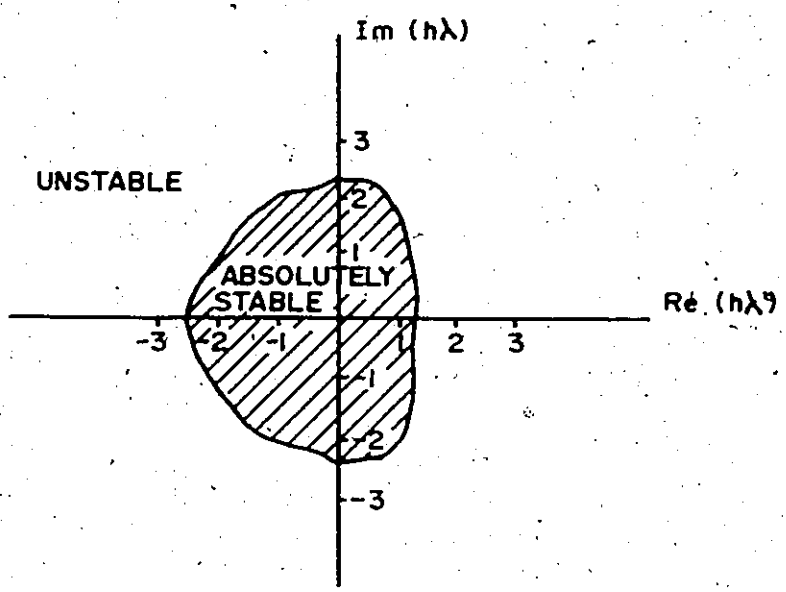


FIGURE 2.1 : TYPICAL STABILITY BOUNDARY FOR CONVENTIONAL INTEGRATION TECHNIQUE

2.5.2 Stability Of Runge-Kutta Methods

The difference equation resulting from an explicit Runge-Kutta method is:

$$y_{n+1} = \left[\sum_{i=0}^{p} \frac{(h\lambda)^i}{i!} + \sum_{i=p+1}^{v} a_i \frac{(h\lambda)^i}{i!} \right] y_n \qquad (2.5.12)$$

or

$$y_{n+1} = \mu_1 y_n \qquad (2.5.13)$$

where p is the order of the method

v the number of derivative substitutions

a_i are constants depending on the specific formula

and μ_1 is the characteristic root of the equation analogous to the principal root of a multistep method.

The numerical solution is thus:

$$y_n = c_1 \mu_1^n \qquad (2.5.14)$$

and as for the linear multistep equation, we require for absolute stability $|\mu_1| \leq 1$; this usually requires $|h\lambda| \leq k$ where k is of the order 1 to 10.

Note the difference in the basic causes of instability for the two types of solution. The difference equation for the multistep method yields extraneous roots

which can cause instability if any one of them is greater than 1. However, for the Runge-Kutta technique, there are no extraneous roots and instability is caused only by taking a step size which is too large for the series to represent the characteristic root accurately.

2.5.3 Nonlinear Systems

The preceding stability analysis has been applied to the linear system $\dot{y} = \lambda y$. There is no general stability theory for the nonlinear o.d.e.: $\dot{y} = f(t,y)$, but at any point in the integration, the equation can be linearized by a Taylor Series about that point, (t_n, y_n).

$$\dot{y} = f(t_n,y_n) + \left.\frac{\partial f}{\partial y}\right|_n (y-y_n) + O(h^2) \qquad (2.5.15)$$

Hildebrand (1956) has shown that the stability characteristics of the linearized o.d.e. are very similar to those of the original equation for small h. Thus the stability analysis for the linear case represents the nonlinear problem at least locally.

The eigenvalues of (2.5.15) corresponding to λ of $\dot{y} = \lambda y$ are the eigenvalues of the Jacobian matrix $\frac{\partial f}{\partial y}$.

For the system $\dot{\underline{y}} = \underline{f}(\underline{y})$ the Jacobian matrix is:

$$\underset{\approx}{J} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_1}{\partial y_2} & \cdots\cdots\cdots & \frac{\partial f_1}{\partial y_n} \\ \frac{\partial f_2}{\partial y_1} & \frac{\partial f_2}{\partial y_2} & \cdots\cdots\cdots & \frac{\partial f_2}{\partial y_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n}{\partial y_1} & \frac{\partial f_n}{\partial y_2} & \cdots\cdots\cdots & \frac{\partial f_n}{\partial y_n} \end{bmatrix} \qquad (2.5.16)$$

2.6 Stiffness

A stiff o.d.e. is one in which one component of the solution decays much faster than others. This may occur in a single o.d.e. or in a system of o.d.e.s. The term "stiff" probably arose from the study of mechanical or structural problems which had both soft and stiff spring constants (Steeper, 1970).

An example of a single stiff equation is:

$$\frac{dy}{dt} = -200\,[y - z(t)] + \dot{z}(t) \qquad (2.6.1)$$

$$y(0) = 10 \qquad (2.6.2)$$

$$z(t) = 10 - (10 + t)e^{-t} \qquad (2.6.3)$$

The analytical solution is:

$$y(t) = z(t) + 10e^{-200t} \qquad (2.6.4)$$

It is more common, however, for this condition to occur in systems of equations. As an example:

$$\dot{y} = \begin{bmatrix} -500.5 & 499.5 \\ 499.5 & -500.5 \end{bmatrix} y \qquad (2.6.5)$$

$$\underline{y}(0) = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \qquad (2.6.6)$$

with analytical solution:

$$y_1(t) = e^{-t} - e^{-1000t} \qquad (2.6.7)$$

$$y_2(t) = e^{-t} + e^{-1000t} \qquad (2.6.8)$$

The eigenvalues of the matrix of coefficients are $\lambda_1 = -1000$ and $\lambda_2 = -1$. In terms of time constants, they are $\tau_1 = 0.001$ and $\tau_2 = 1$; the eigenvalues and time constants are negative reciprocals.

Both y_1 and y_2 have a rapidly decaying component, corresponding to λ_1, which very quickly becomes insignificant. After the brief initial phase of the solution in which the λ_1 component is not negligible, we would like to use a step size h which is determined only by the component of the solution corresponding to λ_2. However, stability considerations demand that both $|h\lambda_1|$ and $|h\lambda_2|$ be bounded for the entire integration, usually by a number from 1 to 10. As an example, Euler's method requires $|h\lambda_2| = 2$.

Although the component of the solution corresponding to λ_1 is of no practical interest, the criterion of absolute stability forces us to use an extremely small

value of h over the entire range of integration. As a result, the computation time necessary to integrate a highly stiff system can become excessive. It is desirable then, to use a method which is not so restrictive in step size, for example, a method that is A-stable.

The ratio of the largest to the smallest eigenvalue in absolute value is called the stiffness ratio. In (2.6.5), the stiffness ratio is 1000, a moderate value. In practice, ratios of the order 10^6 have been encountered, for example, the hydrogen-bromine reaction (Creighton, 1971).

Mathematically, for a stiff system, the real part of the largest eigenvalue is much less than zero and the absolute value of the ratio of the real part of the largest eigenvalue to the real part of the smallest eigenvalue is much greater than one.

$$\text{i.e. } Re(\lambda_{max}) << 0 \qquad (2.6.9)$$

$$\left|\frac{Re(\lambda_{max})}{Re(\lambda_{min})}\right| >> 1 \qquad (2.6.10)$$

One way of determining stiffness is to calculate the eigenvalues of the Jacobian matrix, but for all but the most simple cases, this is very tedious. In practice, an easy way to identify a stiff system is to use a

conventional method to integrate it. If an extremely small value is needed for the step size (say $h < 0.01$) to avoid instability, the system is probably stiff and a special method should be used.

Stiff o.d.e.s occur commonly in reaction kinetics, control theory, circuit theory and to some extent in multistage systems. They are obtained in theoretical studies of reaction kinetic models of systems where the rate constants for the reactions involved are widely separated. Such systems occur in many fields of chemistry, particularly when radicals or quasistationary species are involved. The stiffness in control theory could, for instance, stem from the difference in time constants between the fast electrical control circuits and some slow mechanical devices. Stiffness in circuit theory often occurs in situations where transistors or other nonlinear semiconductors with very small time constants are connected to networks, often linear, with much larger time constants. Stiffness can arise in multistage systems, for example, in a distillation column because the time constants for the various components and plates may be very different and also since the time constant for the reboiler is large compared with the time constant of a plate.

Many chemical engineering systems give rise to stiff systems. Such situations occur in chemical reactors

(Amundson, 1965; Amundson and Luss, 1968; Creighton, 1971; DeGroat and Abbett, 1965; Emanuel, 1963; Eschenroeder, Boyer and Hall, 1962; Gelinas, 1972; Moretti, 1965; Robertson, 1967 and Williams and Otto, 1960) and in multistage systems (Davison, 1968; Distefano, 1968a,b,c; Haines, 1969; Mah, Michaelson and Sargent, 1962 and Rosenbrock, 1957).

2.7 Accuracy

In numerical integration there is usually an error associated with the method.

One source of error is the round-off error which is introduced by the machine due to the finite number of significant figures carried. Double precision arithmetic can be used, if it is felt round-off error is critical; however, it is very difficult to estimate and is usually neglected (Henrici, 1962).

The global truncation error of a method is the difference between the actual value and the calculated value:

$$G.T.E. = y_n^{ACT} - y_n^{CALC} \qquad (2.7.1)$$

It is usually very difficult or impossible to calculate.

The local truncation error is the error committed in one integration step from y_n to y_{n+1}. It is often of the form:

$$L.T.E. = Ch^{p+1} y^{(p+1)}(\xi) \quad (t_n < \xi < t_{n+1}) \qquad (2.7.2)$$

If as $h \to 0$, the $L.T.E.$ becomes $o(h^{p+1})$, p is called the order of the method.

The global truncation error is actually the accumulation of the local truncation errors. Because the global error is so difficult to calculate, the local truncation error is usually controlled. By keeping the local error below some specified value at each step, it is hoped that the global error is also kept under control. The step size can be chosen automatically to do this, if there is an estimate for the local error.

2.7.1 Single-step Methods

Among single-step methods, Euler's method has no error estimate, nor do most of the Runge-Kutta methods. Merson (1957) devised the first Runge-Kutta method with an error estimate:

$$k_1 = hf(t_n, y_n)$$

$$k_2 = hf(t_n + \frac{1}{3}h,\ y_n + \frac{1}{3}k_1)$$

$$k_3 = hf(t_n + \frac{1}{3}h,\ y_n + \frac{1}{6}k_1 + \frac{1}{6}k_2)$$

$$k_4 = hf(t_n + \frac{1}{2}h,\ y_n + \frac{1}{8}k_1 + \frac{3}{8}k_2)$$

$$k_5 = hf(t_n + h,\ y_n + \frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4)$$

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 4k_4 + k_5) \qquad (2.7.3)$$

$$L.T.E. = \frac{1}{30}(2k_1 - 9k_3 + 8k_4 - k_5) \qquad (2.7.4)$$

Other Runge-Kutta error estimators have also been developed. Lapidus and Seinfeld (1971) list a number of these.

2.7.2 Predictor-Corrector Methods

Predictor-corrector methods have very convenient local truncation error estimates, if the predictor and corrector are of the same order. For example, consider the third order Adams-Bashforth predictor, Adams-Moulton corrector (2.4.2).

$$\text{predictor error} = \frac{9}{24} h^4 y^{(4)}(\xi_p)$$

$$t_{n-1} < \xi_p < t_{n+1} \qquad (2.7.5)$$

$$\text{corrector error} = -\frac{1}{24} h^4 y^4(\xi_c)$$

$$t_n < \xi_c < t_{n+1} \qquad (2.7.6)$$

The exact value of y at t_{n+1} is given by:

$$y(t_{n+1}) = y_{n+1}^{PRED} + \frac{9}{24} h^4 y^{(4)}(\xi_p) \qquad (2.7.7)$$

and $$y(t_{n+1}) = y_{n+1}^{CORR} - \frac{1}{24} h^4 y^{(4)}(\xi_c) \qquad (2.7.8)$$

Thus $$y_{n+1}^{PRED} + \frac{9}{24} h^4 y^{(4)}(\xi_p) = y_{n+1}^{CORR} + \frac{1}{24} h^4 y^{(4)}(\xi_c) \qquad (2.7.9)$$

Assuming the fourth derivative remains fairly constant over the interval

$$y_{n+1}^{CORR} - y_{n+1}^{PRED} = \frac{5}{12} h^4 y^{(4)}(\xi) \quad t_{n-1} < \xi < t_{n+1} \qquad (2.7.10)$$

thus predictor error $$= \frac{9}{10} (y_{n+1}^{CORR} - y_{n+1}^{PRED}) \qquad (2.7.11)$$

$$\text{corrector error} = -\frac{1}{10} (y_{n+1}^{CORR} - y_{n+1}^{PRED}) \qquad (2.7.12)$$

A possible PECE algorithm is:

(1) calculate y_{n+1}^{PRED}

(2) add predictor error (from previous step, assuming little change)

(3) evaluate derivative

(4) calculate y_{n+1}^{CORR}

(5) add corrector error

(6) evaluate derivative

2.7.3 Changing Step Size

Usually the local error is kept below some specified tolerance. If the error is too large (greater than tolerance) or too small (less than say $\frac{1}{10}$ of the tolerance), the step size is adjusted. One common algorithm is to double the step when the error is too small and halve the step when the error is too large.

Another scheme if the error varies as say, h^4

$$E = Ch^4 \qquad (2.7.13)$$

Then if,

$$E_{ACTUAL} = C_1 h_{OLD}^4 > TOLERANCE \qquad (2.7.14)$$

If the desired error is say, half the tolerance,

$$E_{DESIRED} = 0.5(TOLERANCE) = C_2 h_{NEW}^4 \qquad (2.7.15)$$

Assuming $C_1 \doteq C_2$

$$\frac{E_{ACTUAL}}{E_{DESIRED}} = \left(\frac{h_{OLD}}{h_{NEW}} \right)^4 \qquad (2.7.16)$$

Thus the new step size is chosen as:

$$h_{NEW} = h_{OLD} \left[\frac{E_{DESIRED}}{E_{ACTUAL}} \right]^{\frac{1}{4}} \qquad (2.7.17)$$

$$= h_{OLD} \left[\frac{0.5(TOLERANCE)}{E_{ACTUAL}} \right]^{\frac{1}{4}} \qquad (2.7.18)$$

2.7.4 General Techniques

Usually relative error is controlled but if the independent variable becomes small (less than 1), it is better to control absolute error.

For a system of equations, the maximum error may be controlled. Gear (1971a) controls the Euclidean norm of the relative local truncation errors.

There are many other numerical integration techniques which may or may not have error estimators. An efficient and convenient local error estimate is a

desirable feature of any integration technique, assuming a great deal of computation time is not required for estimation and step-changing.

If the method does not have an estimate, one strategy is to integrate the method using a step h and then repeat the integration for two steps each of $\frac{1}{2}h$. By Richardson extrapolation, an expression for the local truncation error is (Carnahan, Luther and Wilkes, 1969):

$$L.T.E. = \frac{2^p (y_{n+1,2} - y_{n+1,1})}{2^p - 1} \qquad (2.7.19)$$

where p is the order of the method

$y_{n+1,1}$ is the value calculated by step h

$y_{n+1,2}$ is the value calculated by 2 steps of $\frac{h}{2}$

2.8 Accuracy Versus Stability

For a stable numerical solution, conventional methods for solving o.d.e.s require that $|h\lambda|$ for each eigenvalue be less than a number of the order 1 to 10 and hence the step size is restricted to be not much greater than the smallest time constant present. Ordinarily this restriction on h does not control the step size since the accuracy requirement, or the prescribed bound on the local truncation error, usually forces the step size to satisfy an even more stringent restriction. However, for stiff

equations, the bound on the step size necessary to maintain stability can be three to five orders of magnitude smaller than that necessary to maintain accuracy (Walters, 1972), making it uneconomical to solve stiff systems with standard methods.

Also in stiff systems, some components may be so small that it is not necessary to simulate them accurately to get an accurate solution for the main components. What is required is that the solution be stable.

As a result many numerical methods not subject to the above stability condition have been developed for solving stiff equations. A literature survey of these appears in the following chapter.

3. LITERATURE SURVEY OF NUMERICAL METHODS TO SOLVE STIFF SYSTEMS

The first mention in the literature of the stiffness problem was probably Curtiss and Hirschfelder (1952). Since then, the subject has received a great deal of attention, notably in the last five or ten years. This chapter surveys the methods which have been proposed to handle stiffness.

3.1 Conventional Methods

Conventional methods are restricted to a very small step size of the order $|h\lambda|$ less than 1 to 10. If a conventional method is to be chosen, it is reasonable to use one which requires the least work per time step, since they will use roughly the same step size and they will all be extremely accurate with such a small step. This method is the Euler method which has been recommended by Franks (1971, 1972b). However Euler's method will still use a great deal of computer time in solving stiff problems as shown in Chapter 6.

3.2 Pseudo Steady State Approach

Franks (1972a) suggests that the proper approach to stiff problems may be to simply eliminate those differential equations having small time constants and solve them as algebraic equations instead. This is generally known as the pseudo steady state approach.

Snow (1966) developed a computer program to calculate the product distribution in any homogeneous reaction mechanism. The program includes a numerical method to apply the steady state assumption when a mechanism involves intermediates present in low concentrations.

Brayton, Gustavson and Liniger (1966) analyze the transient behaviour of a transistor circuit. They reduce the original system of differential equations by setting some small resistors equal to zero, thereby eliminating the fast components. They point out that the new steady state differs from the original one. The error is roughly of the order of the ratio of the resistors neglected to those retained. For some problems this accuracy might be sufficient.

The pseudo steady state approach does introduce some large errors initially in some components. It is possible for these errors to be propogated throughout the solution so it is risky to use this procedure (Emanuel, 1967; Kolbrack, 1967; Snow, 1967; Gelinas, 1972). Also

with a complex nonlinear system, it may be very difficult or even impossible to eliminate the small time constants.

Perhaps future research will find the conditions under which the pseudo steady state approach is applicable and the amount of error incurred.

3.3 Stiff Techniques

In the past decade, many new sophisticated methods have been developed to overcome the instability problem.

The first major attempt to survey stiff techniques was Bjurel *et al.* (1970). There have been many review papers on a smaller scale (Lapidus and Seinfeld, 1971; Walters, 1971; Sigurdsson, 1970; Gear, 1969b; Steeper, 1970; Calahan, 1969; Hull, 1969).

This survey is not as detailed as that of Bjurel *et al.* but it extends the literature to early 1974, but since this is an area of current research, it should soon be obsolete.

Categorizing the methods is difficult since some methods would occur in more than one category; here we use several rough categories. Methods which are actually tested in Chapter 6 are explained in greater detail.

3.3.1 Extension Of Conventional Methods

Many authors have slightly extended the region of stability of conventional methods. Some of these are mentioned here.

Lawson (1966, 1967b) has developed fifth and sixth order Runge-Kutta formulae with extended ranges of stability.

Schoen (1971) develops fifth and sixth order predictors for use with Adams-Moulton correctors of the same order. The resulting PECE algorithms have larger regions of absolute stability than the Adams algorithms of corresponding order.

Emanuel (1964) examines the interaction between a stiff equation and several common integration procedures. He derives a convergence condition for the Runge-Kutta method which can be used to control the integration step size. He also introduces a technique for maximizing the step size and applies it to the generalized Adams predictor-corrector procedure.

Robertson (1966) combined linearity with a free parameter in the fourth order Simpsons Rule and the third order Adams-Moulton equation. For certain values of the parameter, he obtained stability regions superior

to those of either method (Sigurdsson, 1970).

3.3.2 Explicit Methods

Treanor (1966) has proposed a modified Runge-Kutta method. He observed that many stiff systems are of the form $\dot{y} = -P(y-z)$, where P is a large number and z is a slowly varying function of time. z can be approximated by a power series in time containing unknown parameters which are determined in the course of the integration. In particular, the method assumes that the derivative can be approximated in any interval by:

$$\dot{y}_i = -(P_i)_n y_i + (A_i)_n + (B_i)_n t + (C_i)_n t^2 \qquad (3.3.1)$$

The four constants A_i, B_i, C_i and P_i are evaluated by determining the value of the derivative at four points in the interval $[t_n, t_n + h]$ and solving for the constants from evaluation of the above equation at the four points. The four points are t_n, $t_n + \frac{1}{2}h$, $t_n + \frac{1}{2}h$ and $t_n + h$.

The following algorithm is obtained:

$$\begin{aligned} y^{(1)}_{n+\frac{1}{2}} &= y_n + \left(\frac{h}{2}\right) f_n \\ y^{(2)}_{n+\frac{1}{2}} &= y_n + \left(\frac{h}{2}\right) f^{(1)}_{n+\frac{1}{2}} \\ y^{(3)}_{n+1} &= y_n + h\left[2f^{(2)}_{n+\frac{1}{2}} F_2 + f^{(1)}_{n+\frac{1}{2}} PhF_2 + f_n(F_1 - 2F_2)\right] \end{aligned} \qquad (3.3.2)$$

$$y_{n+1} = y_n + h f_n F_1 + h v_3(Py_n + f_n) + h v_2(Py^{(1)}_{n+\frac{1}{2}} + f^{(1)}_{n+\frac{1}{2}}$$

$$+ h v_2(Py^{(2)}_{n+\frac{1}{2}} + f^{(2)}_{n+\frac{1}{2}}) + h v_1(Py^{(3)}_{n+1} + f^{(3)}_{n+1})$$

where

$$F_1 = \frac{e^{-Ph} - 1}{-Ph}, \quad F_2 = \frac{e^{-Ph} - 1 + Ph}{(Ph)^2}$$

$$F_3 = \frac{e^{-Ph} - 1 + Ph - \frac{1}{2}(Ph)^2}{(Ph)^3}$$

$$v_1 = -F_2 + 4F_3 \qquad (3.3.3)$$

$$v_2 = 2(F_2 - 2F_3)$$

$$v_3 = 4F_3 - 3F_2$$

$$P_i = \frac{f^{(2)}_{i,n+\frac{1}{2}} - f^{(1)}_{i,n+\frac{1}{2}}}{y^{(2)}_{i,n+\frac{1}{2}} - y^{(2)}_{i,n+\frac{1}{2}}}$$

The value of P is taken to be the largest value of P_i; if this value is negative, P is set to zero.

For small h, the method is identical to the fourth order Runge-Kutta method (Eqn. 2.3.4).

The step size control is to bound $\left| \frac{y_{n+1} - y_n}{y_{n+1}} \right|$ at each step between an upper and lower tolerance. Double-halving is used.

Jung (1967) has written a modified version of Treanor's method.

Lomax and Bailey (1967) have analyzed the method of Treanor and shown that it is efficient if there is only one large real negative eigenvalue or if all the large negative eigenvalues are close to each other and real. They also suggest some improvements of the method.

Richards, Lanning and Torrey (1965) devised a scheme for large stiff systems based on Euler's method. The method is based on two qualitative observations:

(1) It is possible to take a large integration step even with Euler's method, if the value of y is sufficiently close to the desired trajectory.

(2) The onset of instability is indicated by sudden reversals of the derivative.

The algorithm uses Euler's method with step size $h = \beta \frac{\| y_n \|}{\| \dot{y}_n \|}$, where the double bars indicate Euclidean norms. β is a parameter that can be adjusted for stability. Essentially it limits the fractional change in the large components of y. $\beta = 0.01$ is a reasonable value for most problems.

If the derivatives change greatly; i.e., if

$$\cos\theta = \frac{\dot{y}_n \cdot \dot{y}_{n+1}}{\| \dot{y}_n \| \cdot \| \dot{y}_{n+1} \|} < -\frac{1}{S} \qquad (3.3.4)$$

then the initial value of y_{n+1} is abandoned and replaced by an interpolated value:

$$t_{n+1} = t_n + S(\Delta t)$$

$$y_{n+1}^{NEW} = y_n + S(y_{n+1} - y_n) \qquad (3.3.5)$$

$$S = \frac{\dot{y}_n \cdot (\dot{y}_n - \dot{y}_{n+1})}{\| \dot{y}_{n+1} - \dot{y}_n \|^2}$$

Fowler and Warten (1967) derived an explicit second order method for large stiff systems, where the smallest time constant is real.

Consider the differential equation $\dot{y} = f(t,y)$ with solution y_T. At each step, let y_C be the computed solution. y_C is assumed to be the sum of two functions, y_A, an asymptotic part and y_{PE} a perturbation from the asymptote. y_A is to be determined from present and past values of y_T. In practice of course, y_T is not available and the approximations, y_C, obtained at the previous steps are used in the calculations instead of y_T.

Assume y_A and y_{PE} are of the form:

$$y_A(t + \xi) = y_A(t) + \xi \dot{y}_A(t)$$

$$y_{PE}(t + \xi) = y_{PE}(t) + \frac{e^{\lambda_P \xi} - 1}{\lambda_P} \dot{y}_{PE}(t) \qquad (3.3.6)$$

$0 \leq \xi \leq h$, thus:

$$y_C(t + \xi) = y_A(t + \xi) + y_{PE}(t + \xi)$$

$$= y_A(t) + y_{PE}(t) + \xi \dot{y}_A(t) + \frac{e^{\lambda_P \xi} - 1}{\lambda_P} \dot{y}_{PE}(t)$$

Five conditions are imposed to determine the five constants $y_A(t)$, $y_{PE}(t)$, $\dot{y}_A(t)$, $\dot{y}_{PE}(t)$, λ_P. The first three of these make the method second order exact and the last two determine the constants uniquely:

$$(1) \quad y_A(t) + y_{PE}(t) = y_T(t)$$

$$(2) \quad \dot{y}_A(t) + \dot{y}_{PE}(t) = \dot{y}_T(t)$$

$$(3) \quad \lambda_P \dot{y}_{PE}(t) = \ddot{y}_T(t) \qquad (3.3.7)$$

$$(4) \quad y_A(t) = y_T(t)$$

$$(5) \quad \dot{y}_A(t) = [y_T(t) - y_T(t - h_o)]/h_o$$

There are several possible modifications under certain conditions and a complex step size control.

Nigro (1969) uses multistep formulae of the form:

$$y_{n+1} = \sum_{i=1}^{3} \alpha_i y_{n+1-i} + h \sum_{i=1}^{3} \beta_i \dot{y}_{n+1-i} \qquad (3.3.8)$$

He constructs arbitrarily large stability intervals, $h\lambda_{max}$, by choosing appropriate values of the α and β. By perturbing the α and β, larger relative stability intervals can be created.

The method is not self-starting; it needs three calculated points before it can take over. The step size is fixed, as there is not a good error or step changing analysis.

Lomax (1968) proposed some second order explicit Runge-Kutta methods which extended the region of stability considerably.

Pope (1963) proposes the explicit difference equation:

$$y_{n+1} = y_n + h f_n + \ddot{y} f_y^{-2} [\exp(hf_y) - 1 - hf_y] \qquad (3.3.9)$$

Kubicek (1975) has developed a complex one-step integration method, based on boundary value technique. The algorithm is nonlinear, explicit and second order. Kubicek also claims it is A-stable.

Chu and Berman (1974) describe an explicit complex single-step method. It is second order.

3.3.3 Implicit Multistep Methods

Klopfenstein and Davis (1971) present a study of a class of PECE algorithms consisting of an application of a predictor followed by application of one iteration of a pseudo Newton-Raphson method to a corrector.

They use algorithms of the form:

$$p_{n+1} = y_n + h\,[\alpha\, f_n + \beta\, f_{n-1}]$$

$$c_{n+1} = y_n + h\,[v\, f(t_{n+1}, p_{n+1}) + u\, f_n] \qquad (3.3.10)$$

$$y_{n+1} = p_{n+1} + [\alpha\, I - v\, h\, J]^{-1}[c_{n+1} - p_{n+1}]$$

The α, β, u, v and α are real constants, I is the identity matrix and J approximates the Jacobian.

The algorithm with $\alpha = 1-\beta$, $v = 1-u$, $u < \frac{1}{2}$ is first order with a first order error estimate:

$$e_{n+1} = p_{n+1} - y_{n+1} + \frac{\beta + \frac{1}{2}}{\beta - u + 1}\,(c_{n+1} - p_{n+1}) \qquad (3.3.11)$$

$\alpha = 1-u$ will shift the stability region so that the centre of an inscribed circle is at the origin.

The algorithm with $\alpha = \frac{3}{2}$, $\beta = -\frac{1}{2}$, $v = u = \frac{1}{2}$ is second order with first order error estimate:

$$e_{n+1} = p_{n+1} - y_{n+1} + \frac{5}{6}(c_{n+1} - p_{n+1}) \qquad (3.3.12)$$

$a = 0.71$ will shift the stability region to the origin.

There are variants of the θ method, known for many years:

$$y_{n+1} = y_n + h[(1 - \theta)\dot{y}_{n+1} + \theta\, \dot{y}_n] \qquad (3.3.13)$$

Liniger and Willoughby (1970) and Brandon (1972) discuss methods of this type. See Appendix B for an analysis of Brandon's method. Zein and Hakimi (1971) use Liniger and Willoughby's method with Newton-Raphson iteration to solve circuit problems.

Sandberg and Shichman (1968), Shichman (1969) recommend taking one Newton-Raphson iteration of the implicit Euler method with y_n used as the estimate of y_{n+1}:

$$y_{n+1} = y_n + h\, \dot{y}_{n+1}$$

$$y_{n+1} = y_n + [1 - h \frac{\partial f}{\partial y}\Big|_{\quad}]^{-1}[h\, \dot{y}_n] \qquad (3.3.14)$$

A rough error estimate and step control scheme is given. Sloate (1970) uses an extension of this method.

Hodgkins (1969) presents an algorithm based on the trapezoidal rule in conjunction with global extrapolation.

Lindberg (1971a,b) gives a simple smoothing procedure which damps out the oscillation of the trapezoidal rule applied to stiff systems.

The FACE program (Price, 1970) uses a simple, first order, one-step algorithm, known as a linear extrapolation method. There is provision for using a special integrator which permits fast portions of the problem to be run with a reduced step size. This is an approximation which assumes the inputs from the main portion of the problem are constant during a normal time step.

Curtiss and Hirschfelder (1952), Krogh (1971, 1973) and Gear (Appendix A) have recommended the numerical differentiation formulae for stiff systems. Cryer (1972) discusses a stability theorem for the formulae. Calahan (1971) also discusses Gear's method. Dill and Gear (1971) used an interactive computer graphics program to find stiffly stable methods of orders seven and eight.

Branin *et al.* (1971) have developed a version of the numerical differentiation algorithm different from Gear's which they claim is more stable than his version under certain conditions.

Klopfenstein (1971) generalizes the numerical differentiation formulae of orders one through six to increase the angular width of the wedge of stability at only modest cost in increased local truncation error.

Bickart and Picel (1973) derive some high order stiffly stable composite multistep methods.

Cooke (1972, 1973) analyzes stiff stability.

Williams and de Hoog (1974) derive a class of A-stable advanced multistep methods with order up to ten. It compares favourably with Gear's method.

Brunner (1972) derives linear k-step methods ($k \geq 2$) with constant coefficients by choosing as the second characteristic polynomial of the method, a Schur polynomial whose coefficients depend on a certain set of parameters. For the case k = 2, the corresponding two-step methods are always A-stable.

Bjurel (1969, 1972) derives implicit and explicit Adams-like multistep formulae for equations of the type $P\dot{y} = f(t,y)$ where P is a polynomial with constant coefficients and $|\partial f/\partial y|$ is small compared with the roots of P.

Ehle (1968) shows how one can obtain one-step methods of arbitrarily high order which are A-stable. These are n-stage implicit Runge-Kutta processes of order $2n$ and a generalized class of linear one-step methods of the form:

$$y_{n+1} = y_n + \sum_{i=1}^{j} \alpha_{ij} h^i [(-1)^{i+1} y_{n+1}^{(i)} + y_n^{(i)}]$$

$(j=1,2,3,\ldots)$ (3.3.15)

Lindberg (1972) describes the package IMPEX which consists of two procedures for the treatment of the transient phase, and two algorithms, one of order two and one of order four, for computation of the smooth solution after the transient.

The Lipschitz constant at the starting point is estimated. From this estimate, a starting step size for the calculation of the transient phase is computed. Throughout the transient phase, a fourth order Runge-Kutta method is used. When the transients are negligible, the variation of the solution is examined in order to get a reasonable starting step size for the main program. The smooth solution is computed by the implicit midpoint rule with second order smoothing and fourth order extrapolation (Bulirsch and Stoer, 1966). The step size is controlled using an estimate of the global truncation error.

IMPEX is the only program found containing different algorithms for both the transient and smooth phases. However, in a practical simulation, there may be transients constantly arising from system disturbances. IMPEX assumes the transients arise only at the beginning of the simulation.

Zavorin and Khesina (1974) have developed three A-stable third order single-step methods.

Sloate (1971), Bickart, Burgess and Sloate (1971) and Sloate and Bickart (1973) use two simultaneous linear multistep difference equations to integrate stiff systems. In this way Dahlquist's limitation that no A-stable linear multistep formula can exceed order two is bypassed. They present a fourth order A-stable method.

Davison (1973) has devised a single-step implicit algorithm for integrating very large stiff differential equations of the type:

$$\dot{y} = Ay + Bu + f(t,y) \qquad (3.3.16)$$

where $f(t,y)$ has a small Lipschitz constant.

Roe (1967) describes a four-step, predictor-corrector algorithm which is derived by fitting the formula to an exponential plus a quadratic. The method works best if the output versus time is exponential-like and degenerates to the Adams-Moulton method, if the response is not similar to an exponential.

Guderley and Hsu (1972) studied a predictor-corrector method for systems of the form:

$$\dot{y} + Dy = -Ay + Bt \qquad (3.3.17)$$

where D is a diagonal matrix which may have some large

elements and the right hand side is considered as non-stiff. The operator on the left is inverted and the right hand side is approximated by Lagrangian interpolation polynomials at grid points. The integration of the exponential functions is done analytically.

Norsett (1969) gives new finite difference formulae which are exact for the problem $\dot{y} = Py + Q(t)$ where P is a constant and $Q(t)$ is a polynomial of degree q. When $P = 0$, the method is identical with the Adams-Bashforth formulae.

Stineman (1965) used an implicit one-step method for the integration of first and second order equations of the type:

$$a(t)\,\dot{y} + y = b(t)$$

$$c(t)\,\ddot{y} + d(t)\,\dot{y} + y = e(t) \qquad (3.3.18)$$

where the functions a, b, c, d and e are locally approximated by linear functions.

Sarkany and Ball (1969) treat the solution of $\dot{y} = -Dy + f(t,y)$ where D is a diagonal matrix. A predictor-corrector method is used.

Spicer (1968, 1969) defines exact stability as the extraneous roots being constrained to be zero. He derives

predictor-corrector formulae having this property.

Certaine (1960) has developed an iterative multi-step method for solving systems of the form $\dot{y} + Dy = f(t,y)$ where D is a constant diagonal matrix with large positive elements and $f(t,y)$ is a slowly varying function of time.

Snider and Fleming (1974) use aliasing to reduce the computation of finding an accurate trigonometric interpolation for a function with dominant high frequencies. The technique is applied to stiff differential equations, extending the applicability of the method of Certaine to systems with oscillatory forcing functions.

3.3.4 Runge-Kutta Methods

The basic Runge-Kutta equation is explicit:

$$y_{n+1} = y_n + \sum_{i=1}^{v} w_i k_i \qquad (3.3.19)$$

$$k_i = hf(x_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} k_j) \quad c_1=0, i=1,2,\ldots,v$$

Butcher (1964) introduced implicit Runge-Kutta methods. Ehle (1968) and Gear (1970) discuss these further:

$$k_i = hf(x_n + c_i h, y_n + \sum_{j=1}^{v} a_{ij} k_j) \quad c_1=0, i=1,2,\ldots,v \qquad (3.3.20)$$

Rosenbrock (1963), Calahan (1968b), Allen and Pottle (1968), Allen (1969), Haines (1969) and Caillaud and Padmanabhan (1971) introduce semi-implicit methods of the form:

$$k_i = hf(x_n + c_i h,\ y_n + \sum_{j=1}^{i} a_{ij} k_j) \quad c_1 = 0,\ i = 1, 2, \ldots, v \qquad (3.3.21)$$

The implicit and semi-implicit forms are A-stable.

Chipman (1971) defines the concept of strong A-stability. He then introduces a class of strongly A-stable Runge-Kutta processes.

Axelsson (1972) gives formulae for a class of A-stable quadrature methods or equivalently for a certain implicit Runge-Kutta scheme. The methods are strongly A-stable.

Lopes and Phares (1965) and Lawson (1967a) have developed an A-stable Runge-Kutta method. The system must first be transformed by $z(t) = \exp(-\lambda t)y(t)$.

Stanton and Talukdar (1970) use a fourth order Runge-Kutta and a complex transformation of variables to increase the stability interval and permit a large step size.

3.3.5 Miscellaneous Methods

Jain (1972) gives some A-stable methods of order $2n$, with variable coefficients based on Hermite interpolation polynomials, making use of n starting values.

Axelsson (1969) gives A-stable methods of a quadrature type:

$$^{o}y_{n,r} = y_{n,r-1} + h \sum_{k=1}^{n} a_{ik} f(y_{k,r}) \qquad i=1,2,\ldots,n \qquad (3.3.22)$$
$$r=1,2,\ldots,y_{n,0}$$

where a_{ik} are quadrature coefficients over the zeros of $P_n - P_{n-1}(v=1)$ or $P_n - P_{n-2}(v=2)$, where P_n is the Legendre polynomial orthogonal on $[0,1]$ and normalized such that $P_n(1) = 1$.

Dahlquist (1969) devised a method based on local polynomial approximations for two stiff o.d.e.s of particular type. Oden (1971) gives further details of the method.

Watts and Shampine (1972) develop r-block implicit one-step methods which compute a block of r new values simultaneously with each set of application. A subclass of formulae is derived which is related to Newton-Cotes quadrature and it is shown that for block sizes $r=1,2,\ldots 8$, these methods are A-stable, while those for $r=9,10$ are not. They construct A-stable and strongly A-stable

formulae having arbitrarily high orders of accuracy.

Giloi and Grebe (1968) discuss multistep algorithms and the computation of optimum coefficients. They point out that each problem has its own optimum formula, and z-transform techniques are used to study this. This approach may not be practical for general purpose use, since the optimum formula would be difficult or impossible to determine, and the method is limited to linear equations.

Andrus (1967) uses a quasi-analytical approach which takes the inverse Laplace transform of the system equations, using a summation of the partial fractions. Some approximate techniques for handling nonlinearities are discussed, but this appears to be a limitation.

Liniger and Willoughby (1970) introduced the concept of exponential fitting. They proposed single-step methods of the form:

$$y_{n+1} = \alpha_1 y_n + h[\beta_0 \dot{y}_{n+1} + \beta_1 \dot{y}_n] + h^2[\gamma_0 \ddot{y}_{n+1} + \gamma_1 \ddot{y}_n] \qquad (3.3.23)$$

If the characteristic root is chosen by adjusting the free parameters to be $\exp(q_o)$, then we say the method is exponentially fitted at q_o.

Liniger (1968) gives two easy-to-check conditions which together are sufficient and "almost necessary" for A-stability of linear multistep integration formulae.

Liniger (1969) discusses global accuracy and A-stability for one and two-step integration formulae.

Liniger (1971) gives a stopping criterion for the Newton-Raphson method in implicit multistep integration algorithms.

Odeh and Liniger (1972) discuss unconditional fixed step size stability of linear multistep formulae.

Prothero and Robinson (1974) discuss the stability and accuracy of implicit one-step methods to stiff equations of the form:

$$\dot{y} = \dot{g}(t) + \lambda(y - g(t))$$

They describe some new stability properties and recommend a family of methods based on a compromise between accuracy and stability considerations.

Brunner (1974) uses recursive collocation. He approximates the solution on each subinterval by a linear combination of exponential functions which involve only the significant eigenvalues of the Jacobian. The unknown vectors are computed recursively by requiring they satisfy the given system at certain suitable points (collocation),

with the additional condition that the collection of these functions represent a continuous function satisfying the given initial conditions.

Liniger and Odeh (1972) discuss an algorithm wherein several low-order solutions are computed by repeated integration using a multistep method with parameters. By forming suitable linear combinations of such solutions, higher order solutions are obtained. If the parameters are properly chosen, the underlying solutions, and thus the higher order one, can be made A-stable and strongly damping with respect to the stiff components of the system.

Blue and Gummel (1970) discuss rational function approximations for the matrix exponential $\exp(h\lambda)$.

The use of Padé approximations for $\exp(h\lambda)$ is discussed by Calahan (1967), Jung (1968) and Osborne (1969) for linear systems.

Cooper (1969) gives a brief review of stiffness, and an iteration procedure likely to give convergence, both in multistep methods and in the steady state approach.

Miller (1964) gives a second order stiff equation and suggests a scheme for solving it. Cash (1972) extends this method and develops an entirely new algorithm for differential equations of order three and higher, to enable these equations to be integrated with a reasonable step length.

Miranker (1971) gives a description and analysis of a class of matricial difference schemes. This class of schemes is based in part on a generalization of the feature of classical numerical methods of being characterized by approximations at a single point in the complex plane.

The following four methods are restricted to linear systems or systems which can be linearized easily.

Mah, Michaelson and Sargent (1962) developed a stable linearization technique for solving the dynamic behaviour of multistage systems. The coefficients of the differential system are taken as constant over a short time interval. The exact solution of the linearized system is used to predict the values at the end of the interval and hence the coefficients for the next interval.

Buffam and Kropholler (1969) extend the concept of network combing to give the concentration history at all the nodes in a flow network whose nodes have exponential dynamic mixing characteristics with arbitrary time constants. They show that the transition matrix for the flow network is a special form of stochastic matrix whose power series has absolute convergence. The generalization results in a very stable method for integrating linear systems of o.d.e.s.

Davison (1968) has considered the problem of the the numerical integration of large systems of constant

coefficient linear o.d.e.s. The poles and zeros of the solution are obtained and the solution constructed in terms of a sum of exponentials.

Moretti's (1965) technique requires nonlinear o.d.e.s to be reduced to linear form. The numerical solution then requires the evaluation of the complex eigenvalues of an n-th order matrix, thus limiting the analysis to fairly small systems. DeGroat and Abbett (1965) and Maheshwari (1968) have extended this approach.

The following four methods use state transition matrices.

Liou (1966) has devised a fast and accurate method. However, it is for linear systems only; one output is available and the problem must be formulated as a ratio of two polynomials in s (the Laplace operator). Nichol (1970) has a similar method, but simple nonlinearities can be handled.

Giese (1967) also uses state variable techniques and discusses their use in quasi-linear systems. He also discusses a combination approach for stiff equations, where this method is used for the fast portion of the problem and a conventional explicit numerical procedure is used for the slow portion.

Oswald and Smith (1970) also use state transition matrices in their method.

Lee (1967) uses matrix filters to separate the differential equations into slow and fast equations. These are then computed separately and the results are added together. It is limited to linear equations.

Allen and Fath (1969) propose techniques for systems which can be separated into a stiff linear part and a nonstiff linear part. Padé approximations are used.

Jain (1971) transforms the system by $x(t) = \exp(At)y(t)$, where A is a real square matrix and obtains modified Adams and linear multistep methods which are A-stable. Jain gives a working algorithm to obtain A and to calculate $\exp(hA)$.

Shannon (1971) developed a technique using a changeable independent variable of integration. It can be used with any integration method. Appendix C contains an analysis of the technique.

Singular perturbation theory is applicable when the system can be written as:

$$\begin{aligned} \varepsilon \dot{y}_1 &= f_1(t, y_1, y_2) \\ \dot{y}_2 &= f_2(t, y_1, y_2) \end{aligned} \qquad (3.3.24)$$

where ε is a small real parameter and the y_2 components are stable in the sense that if y_2 is fixed at

any time, then the y_1 components rapidly reach an equilibrium solution.

MacMillan (1968) proposes a method based upon this theory.

Miranker (1973) describes an equivalence between a subclass of stiff systems and differential equations subjected to singular perturbations. He uses the characterization of the solution of this class of equations in terms of boundary layers as a means of generating numerical procedures for solving the stiff equations. The numerical procedures have the desirable feature of improving with increasing stiffness.

Diperna (1971) describes a digital simulation method for interconnected continuous systems.

3.4 Methods Used In Dynamic Simulation Executive Packages

As mentioned in Chapter 1, there have been several new problem-oriented executive packages developed in the past five years. Almost all of these programs contain an algorithm for the numerical solution of ordinary differential equations; i.e., the system model contains ordinary differential equations. Table 3.1 summarizes some of the numerical methods used.

Table 3.1: Numerical Methods Used in Dynamic Simulation Packages

Package	Method Used To Integrate Differential Equations	Suited For Stiff Eqns.
MIMIC	4th order Runge-Kutta	No
CSMP	various conventional methods, latest version contains Gear's method	Yes
IMP	Brandon's method	Yes
KARDAZ	explicit finite difference method used to solve a set of quasilinear hyperbolic partial differential equations	No
DYNSYS	Adams-Moulton-Shell	No
PRODYC	any method in CSMP	Yes
REMUS	16 conventional options: Simpson's Rule, various Runge-Kutta, Milne and Adams-Moulton formulae	No
ACME	2nd and 4th order Runge-Kutta, Euler's method	No
DYFLO	2nd and 4th order Runge-Kutta, Euler's method	No
OSUSIM	Euler predictor, trapezoidal rule as corrector	Yes

MIMIC (Northcott, 1967) uses the fourth-order Runge-Kutta method.

CSMP (CSMP, 1967) offers a variety of conventional methods, but the latest version contains Gear's method.

IMP (Appendix B) offers Brandon's method, a new A-stable method very suitable for stiff equations.

Franks (1971) has made a detailed review of the packages DYNSYS, PRODYC, REMUS, DYFLO and OSUSIM.

DYNSYS (Bobrow, Johnson and Ponton, 1970, 1971) uses the Adams-Moulton-Shell method (Shell, 1958; Fairchild, Wengrow and May, 1965).

PRODYC (Ingels and Motard, 1970) has a preprocessor which converts the system model into CSMP statements after which any of the methods available in CSMP (CSMP, 1967) can be used to solve the equations.

REMUS (Ham, 1969) offers sixteen conventional options: Simpson's Rule, and various Runge-Kutta, Milne and Adams-Moulton formulae.

DYFLO (Franks, 1972) offers a choice of simple Euler or second or fourth order Runge-Kutta.

OSUSIM (Koenig, 1972) uses an Euler predictor and a trapezoidal rule as corrector. OSUSIM emphasizes the formation and solution of implicit and explicit algebraic equations.

ACME (Loibl, Camp and Wilkins, 1973) is a new modular package. It simulates both the steady state and dynamic behaviour of chemical engineering systems. It uses several options for solving o.d.e.s: second and fourth order Runge-Kutta and Euler's method.

The packages DYNSYS, REMUS, PRODYC, OSUSIM and ACME basically use the modular approach. All of these executives are highly structured and have their own input language which the user must master. The author has not noticed any particular advantage of any of these, but Franks claims DYFLO is the most convenient program from the user's point of view. DYFLO is not exactly modular; it is somewhere in between the modular approach and the equation-oriented approach. The user of DYFLO writes his own computer program using essentially a series of CALL statements to utilize a rather large number of service subroutines that Franks has provided. Culver (1972) used DYFLO to study distillation column dynamics and he attests to its convenience.

The IMP package requires a good deal of sophistication on the part of the user to set up the system equations for simulation and to select the best options for a particular study.

Kardasz (Brambilla *et al.*, 1971; Kardasz, 1969; Kardasz and Molnar, 1971, 1974) has developed a package

for the simulation of the dynamic behaviour of chemical plants. It is quite different than the other packages just discussed. The language used is not FORTRAN but is based on SIMULA-67. It is structure-oriented according to the flowsheet of the plant to be simulated. The method of simulation is based on the concept of a cell model. Under the assumption of a one-dimensional model, a chemical plant composed of a set of devices can be represented mathematically by a set of quasilinear hyperbolic partial differential equations of first order. After appropriate discretization, each device is subdivided into a finite number of cells. An explicit finite difference method is used to solve the equations.

Other attempts have been made to form dynamic simulation packages. Utsumi (1969) describes the SWAPSO package developed at Stone & Webster, but it has not been pursued (Crean, 1974). A package called EARLYBIRD has been developed at Tulane University but the dynamic simulation portion of it has been discontinued (Weaver, 1974). Procter & Gamble (Shorn and Petty, 1970) has a modular package called FLEX. It uses Euler's method for numerical integration.

As seen in Table 3.1, only several of the packages contain a numerical method suitable for stiff o.d.e.s. The latest version of CSMP contains Gear's method. The

IMP package contains Brandon's method. OSUSIM uses the trapezoidal rule which is A-stable. An equation-oriented package, FORSIM (Carver, 1972, 1973) developed at AECL uses Gear's and the Fowler-Warten method. Franks (1971, 1972b) recommends Euler's method for stiff systems. However, except for IMP, none of the packages are oriented to solving large systems of stiff o.d.e.s.. In Chapter 6, we shall see a comparison of some of these techniques.

4. CONVERGENCE OF IMPLICIT METHODS

Of the methods developed to integrate stiff systems, implicit methods are usually preferred, but particular problems arise in solving the corrector or implicit equation, especially for large stiff systems.

Consider the general linear multistep formula :

$$y_{n+1} = \alpha_1 y_n + \alpha_2 y_{n-1} + \dots + \alpha_k y_{n+1-k} + h[\beta_0 \dot{y}_{n+1} + \beta_1 \dot{y}_n + \dots + \beta_k \dot{y}_{n+1-k}] \quad (4.1)$$

or

$$y_{n+1} = \sum_{i=1}^{k} \alpha_i y_{n+1-i} + h \sum_{i=0}^{k} \beta_i \dot{y}_{n+1-i} \quad (4.2)$$

$$y_{n+1} = h\beta_0 \dot{y}_{n+1} + \sum_{i=1}^{k} (\alpha_i y_{n+1-i} + \beta_i \dot{y}_{n+1-i}) \quad (4.3)$$

Since the last term is known, we can replace it by ω_n :

$$y_{n+1} = h\beta_0 f(t_{n+1}, y_{n+1}) + \omega_n \quad (4.4)$$

$$y_{n+1} - h\beta_0 f(t_{n+1}, y_{n+1}) - \omega_n = 0 \quad (4.5)$$

From the above equation we wish to determine y_{n+1}.

For a system of m o.d.e.s, this is equivalent to solving m nonlinear equations. Several common methods of doing this are outlined in this chapter, but this is an ongoing area of research and perhaps better methods will be found.

4.1 Jacobi Iteration

The obvious way of solving this is by repeated substitution:

$$y_{n+1}^{(s+1)} - h\beta_0 f(t_{n+1}, y_{n+1}^{(s)}) - w_n = 0 \qquad (4.1.1)$$

until $y_{n+1}^{(s+1)}$ is equal to $y_{n+1}^{(s)}$ to the degree of accuracy desired.

This approach, called Jacobi iteration, is commonly used for nonstiff o.d.e.s, but for stiff problems, it does not converge unless the step size is very small.

4.2 Accelerated Iteration

A modification of Jacobi iteration is :

$$(1+\alpha)y_{n+1}^{(s+1)} - h\beta_0 f(t_{n+1}, y_{n+1}^{(s)}) - w_n - \alpha y_{n+1}^{(s)} = 0 \qquad (4.2.1)$$

where α is an acceleration parameter.

This will increase the rate of convergence, but the step size condition for convergence is just as stringent as Jacobi iteration.

4.3 Backward Iteration

A backward iteration with much better convergence properties is:

$$y_{n+1}^{(s)} - h\beta_0 f(t_{n+1}, y_{n+1}^{(s+1)}) - w_n = 0 \qquad (4.3.1)$$

however, this still requires the solution of nonlinear equations.

4.4 Newton-Raphson Iteration

The most commonly used method is Newton-Raphson iteration.

For $$g(y_{n+1}) = y_{n+1} - h\beta_o f(t_{n+1}, y_{n+1}) - w_n = 0 \quad (4.4.1)$$

The Newton-Raphson formula is:

$$y_{n+1}^{(s+1)} = y_{n+1}^{(s)} - \left[\left(\frac{\partial g}{\partial y}\right)_{n+1}^{(s)}\right]^{-1} g_{n+1}^{(s)} \quad (4.4.2)$$

$$y_{n+1}^{(s+1)} = y_{n+1}^{(s)} - \left[1 - h\beta_o \left(\frac{\partial f}{\partial y}\right)_{n+1}^{s}\right]^{-1} \left[y_{n+1}^{(s)} - h\beta_o f_{n+1}^{(s)} - w_n\right] \quad (4.4.3)$$

The theoretical convergence condition is quite conservative, but in practice, the method converges for fairly large h and does not slow down the integration process.

$y_{n+1}^{(o)}$ can be a predicted value or if no predictor equation is used, it can be taken as the previous value, y_n.

Note that if the term to be inverted is singular, the step size h can always be reduced to make it non-singular.

4.5 Quasilinearization

The IMP package (Appendix B) uses a method called quasilinearization.

For a single equation:

$$\dot{y} = f(t,y) \qquad (4.5.1)$$

Expand the right hand side in a Taylor Series about the present point,

i.e. $$\dot{y}_n = f(t_n,y_n) + \left.\frac{\partial f}{\partial y}\right|_n (y-y_n) + \ldots \qquad (4.5.2)$$

$$\dot{y}_n = \left.\frac{\partial f}{\partial y}\right|_n y + f(t_n,y_n) - \left.\frac{\partial f}{\partial y}\right|_n y_n \qquad (4.5.3)$$

$$\dot{y}_n = J\,y + B \qquad (4.5.4)$$

For a system of equations, J is the Jacobian matrix; while B is called the augmented constant vector.

When the system is linearized in this way, we no longer have to solve a set of nonlinear equations but a set of linear equations at each time step.

4.6 Large Stiff Systems

Two useful methods for solving stiff corrector equations are Newton-Raphson and quasilinearization. Whichever of the two is used, a set of linear algebraic equations equal in size to the number of o.d.e.s must be solved.

Lapidus and Seinfeld (1971) recommend Jacobi or accelerated iteration if the stiffness ratio is of the order of 10. If it is greater than the order of 10, they recommend Newton-Raphson or backward iteration with the step size selected on the basis of the number of iterations desired per step.

In quasilinearization, the linear equations arise directly from the Taylor Series linearization of the system.

In Newton-Raphson iteration, we must evaluate the term:

$$[I - h\beta_o \left(\frac{\partial f}{\partial y} \right)^{*}_{n+1}]^{-1} [y^{(e)}_{n+1} - h\beta_o f^{(e)}_{n+1} - w_n] \qquad (4.6.1)$$

In general this means solving a set of linear equations :

$$A X = b \qquad (4.6.2)$$

where

$$A = [I - h\beta_o \left(\frac{\partial f}{\partial y} \right)^{(e)}_{n+1}] \qquad (4.6.3)$$

$$b = [y^{(e)}_{n+1} - h\beta_o f^{(e)}_{n+1} - w_n] \qquad (4.6.4)$$

and $$X = A^{-1}b \qquad (4.6.5)$$

This set of linear equations must be solved for each corrector equation.

For large systems of o.d.e.s, the solution of this set of linear equations can require a large amount of computer time.

The matrix of coefficients is usually quite sparse; i.e., there are many zero elements. A method which takes advantage of the sparsity by storing and operating on only the nonzero elements will reduce the amount of computer time appreciably. Therefore we will examine ways of solving sparse sets of linear equations in the following chapter.

5. NUMERICAL SOLUTION OF SPARSE LINEAR ALGEBRAIC EQUATIONS

We saw in Chapter 4 that an implicit numerical integration technique for stiff o.d.e.s requires the solution of a set of linear algebraic equations equal in size to the number of o.d.e.s. For large systems of o.d.e.s, the solution of this set of linear equations can use a large amount of computer time.

The coefficient matrix of the linear equations is a function of the Jacobian matrix for the system. In general, the Jacobian matrix has many zeros so that using a method which stores and operates on only the nonzero elements will reduce the computer time greatly. Such methods are called sparse matrix techniques.

There are two major ways of solving linear equations: direct elimination techniques such as Gaussian elimination, and iterative methods such as Jacobi and Gauss-Seidel. Traditionally, iterative techniques have been used to handle sparse systems (Varga, 1962; Young, 1971); however, in recent years, elimination methods have been more commonly used (Reid, 1971; Rose and Willoughby, 1972; Westlake, 1968; Willoughby, 1969). Decomposition techniques (Himmelblau, 1973) where the system is broken into two or more subsystems have also been developed, but these require

extensive a priori analysis.

Several conventional methods and some sparse techniques were tested on a set of banded test examples. We will assume that our "sparse" equations contain at-most 15 nonzeros per equation.

5.1 Methods Tested For Solving Linear Algebraic Equations

5.1.1 Conventional Methods

Five conventional methods were studied:

(1)	MINV	An SSP matrix inversion algorithm (SSP, 1970)
(2)	SOLVE	Gaussian elimination with no pivoting
(3)	DECOMP-SOLVE	Gaussian elimination with partial pivoting (Forsythe and Moler, 1967)
(4)	JACOBI	Jacobi iterative method (initial estimate of zero used)
(5)	GAUSS-SEIDEL	Gauss-Seidel iterative method (initial estimate of zero used).

5.1.2 Key (1973)

Key (1973) has written a program called SIMULT based on Gauss-Jordan elimination with seven pivoting options:

(1) Simple Gauss-Jordan elimination. Diagonal elements are used in order as pivots.

(2) Gauss-Jordan partial pivoting. The largest coefficient in absolute value in each column in order is selected as pivot, provided this coefficient is not in a previously selected pivotal row.

(3) Gauss-Jordan full pivoting. The largest coefficient in the entire coefficient matrix that is not in a previously selected pivotal row or column is used.

(4) Minimum row-Minimum column. The coefficient matrix is searched to find the row with the least number of nonzero coefficients. Then for the nonzero elements in this row, the column with the most zeros is chosen as the pivotal column. If more than one row or column is chosen, the row or column with the smallest index is used.

(5) Minimum column-Minimum row. This is similar to the above except that the pivotal column is determined first.

(6) Maximum column-Minimum row. The column with the most nonzero elements is chosen, followed by the row with the least nonzero elements.

(7) Minimum of row entries times column entries. The pivot is chosen from the nonzero element with the smallest product of row and column entries.

The storage scheme is illustrated by the following example:

The matrix

$$\underset{\sim}{P} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 4 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 3 \end{bmatrix} \qquad (5.1.1)$$

is stored as

$$\underset{\sim}{A} = \begin{bmatrix} 3 & 0 \\ 2 & 1 \\ 2 & 4 \\ 1 & 2 \\ 2 & 3 \end{bmatrix} \quad \text{and} \quad \underset{\sim}{ICOL} = \begin{bmatrix} 1 & 0 \\ 2 & 5 \\ 3 & 3 \\ 1 & 4 \\ 3 & 5 \end{bmatrix} \qquad (5.1.2)$$

A contains the nonzero elements and *ICOL* the column index of the corresponding elements. The disadvantage of this is that one full row will make the *A* matrix as large as the original matrix.

5.1.3 IMP (1972)

IMP (Brandon, 1972) is a software system developed by D.M. Brandon at the University of Connecticut. It is now offered as a standard CDC software product through the Application Services Division of CDC. An object copy of IMP, Version 1.1, FTN version 4.0 compiler and optimization level 2 was used.

Five options with the IMP package were tested:

(1) Gauss-Seidel

(2) Gradient

(3) Crout elimination, fixed order, written for general sparse matrices

(4) Crout elimination, variable order, written for general sparse matrices

(5) Crout elimination, variable order, written for variable or constant banded matrices.

The Crout method is a variation of Gaussian elimination which uses less storage.

In addition, a program supplied by D.M. Brandon (Brandon, 1974c), which is a simplified version of the method (4) above, was used. It should run faster because there is none of the overhead associated with the IMP routines.

The storage scheme known as cumulative indexing is as follows:

The matrix

$$\underset{\approx}{P} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 4 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 3 \end{bmatrix} \qquad (5.1.3)$$

is stored as

$$\underset{\sim}{A} = \begin{bmatrix} 3 \\ 2 \\ 1 \\ 2 \\ 4 \\ 1 \\ 2 \\ 2 \\ 3 \end{bmatrix} \qquad \underset{\sim}{JHA} = \begin{bmatrix} 1 \\ 2 \\ 5 \\ 2 \\ 3 \\ 1 \\ 4 \\ 3 \\ 5 \end{bmatrix} \qquad \underset{\sim}{ICUMA} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \\ 9 \end{bmatrix} \qquad (5.1.4)$$

where A is the vector of nonzero entries of P by rows

JHA is the vector of column numbers of nonzero elements for each row

ICUMA is a cumulative vector which separates *JHA* by rows.

5.1.4 Schappelle (1967)

Schappelle (1967) developed a program called LINEQ4 which is available from VIM, the CDC 6000 Series User Association.

The method first performs a systematic rearrangement, called pre-triangularization, of the rows and columns of the coefficient matrix. The equations then appear as:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \qquad (5.1.5)$$

where A is upper triangular, B and D are rectangular, and C is quasi lower triangular meaning no lead element of any row of the matrix is further to the right than the lead element of its preceding row.

$$\begin{aligned} A\,X_1 + B\,X_2 &= C_1 \\ C\,X_1 + D\,X_2 &= C_2 \end{aligned} \qquad (5.1.6)$$

solving for X_2

$$(D - C A^{-1} B) X_2 = (C_2 - C A^{-1} C_1) \qquad (5.1.7)$$

X_2 is solved by Gaussian elimination with full pivoting

$$A X_1 = C_1 - B X_2 \qquad (5.1.8)$$

X_1 is solved by back-substitution, since A is upper triangular.

The solution is corrected by iterating on the residuals.

The storage scheme is as illustrated below:

The matrix

$$\underset{\approx}{P} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 4 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 3 \end{bmatrix} \qquad (5.1.9)$$

is stored as

$$\begin{bmatrix} 1 & 1 & 3 \\ 2 & 2 & 2 \\ 2 & 5 & 1 \\ 3 & 2 & 2 \\ 3 & 3 & 4 \\ 4 & 1 & 1 \\ 4 & 4 & 2 \\ 5 & 3 & 2 \\ 5 & 5 & 3 \end{bmatrix}$$

i.e., row number, column number, and value for each nonzero element in any order.

5.1.5 Bending-Hutchison (1973)

Bending and Hutchison (1973) developed the concept of an "operator list" to store the solution process by Gaussian elimination. The process consists of two stages, triangularization and back-substitution.

5.1.5.1 *Triangularization*

For the system

$$\begin{bmatrix} a_1 & 0 & a_9 & a_{11} \\ 0 & a_6 & 0 & a_2 \\ a_4 & 0 & a_8 & 0 \\ a_5 & 0 & 0 & a_3 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} a_7 \\ a_{10} \\ 0 \\ 0 \end{bmatrix} \qquad (5.1.10)$$

(1) eliminate a_2, row 2 = row 2 - $\frac{a_2}{a_3}$ row 4

$$\begin{matrix} a_1 & 0 & a_9 & a_{11} & a_7 \\ a_{12} & a_6 & 0 & 0 & a_{10} \\ a_4 & 0 & a_8 & 0 & 0 \\ a_5 & 0 & 0 & a_3 & 0 \end{matrix} \qquad a_{12} = 0 - \frac{a_2}{a_3} a_5 = -a_5 \frac{a_2}{a_3}$$

(5.1.11)

(2) eliminate a_{11}, row 1 = row 1 - $\frac{a_{11}}{a_3}$ row 4

$$\begin{matrix} a_1^* & 0 & a_9 & 0 & a_7 \\ a_{12} & a_6 & 0 & 0 & a_{10} \\ a_4 & 0 & a_8 & 0 & 0 \\ a_5 & 0 & 0 & a_3 & 0 \end{matrix} \qquad a_1^* = a_1 - a_5 \frac{a_{11}}{a_3} \qquad (5.1.12)$$

(3) eliminate a_9, row 1 = row 1 - $\frac{a_9}{a_8}$ row 3

$$\begin{matrix} a_1^{**} & 0 & 0 & 0 & a_7 \\ a_{12} & a_6 & 0 & 0 & a_{10} \\ a_4 & 0 & a_8 & 0 & 0 \\ a_5 & 0 & 0 & a_3 & 0 \end{matrix} \qquad a_1^{**} = a_1^* - a_4 \frac{a_9}{a_8} \qquad (5.1.13)$$

The above process can be represented by a string of integers:

New element $a_i = -a_j \frac{a_k}{a_l} \rightarrow -k\ l\ i\ j$ (5.1.14)

Element changed $a_i = a_i - a_j \frac{a_k}{a_l} \rightarrow -k\ l\ i\ j$ (5.1.15)

For this example:

-2 3 12 5 -11 3 1 5 -9 8 1 4 0 0 terminates string

5.1.5.2 *Back-Substitution*

$$
\begin{aligned}
X_0 &= -1 \quad \text{(dummy variable)} \\
X_1 &= -a_7 X_0 / a_1^{**} \\
X_2 &= (-a_{10} X_0 - a_{12} X_1)/a_8 \\
X_3 &= -a_4 X_1 / a_8 \\
X_4 &= -a_5 X_1 / a_3
\end{aligned}
\qquad (5.1.16)
$$

This process can also be characterized by a string of integers:

$$a_i = -a_j \frac{X_k}{a_l} \rightarrow j\ k\ -i\ l \qquad (5.1.17)$$

or

$$a_i = (-a_j X_k - a_m X_n)/a_l \rightarrow j\ k\ m\ n\ -i\ l \qquad (5.1.18)$$

For this example:

7 0 -1 1 10 0 12 1 -2 6 4 1 -3 8 5 1 -4 3 0 0

These two strings of integers form what is called the operator list, i.e., the particular solution process by Gaussian elimination. Subroutine TRGB solves the equations for the first time and creates the operator list which can be stored in core or on tape if the system is large. Subroutine TRGB2 can re-solve the system using only the operator list. If the zero elements remain zero and the nonzero elements change, the same operator list can be used to solve the new system. This is usually what occurs during numerical integration. The operator list could be set up on the first integration step by TRGB and used on later steps by TRGB2 to solve each new linear system. The operator list can change during the integration; however, in all examples tested, the original list could be used for the entire integration.

The pivot is chosen by the row and column each with the most zeros. The row is selected first. In case of ties, the largest element is chosen. The storage scheme is row number, column number, value.

5.2 Numerical Testing

Banded systems of 50, 100 and 200 linear equations with bandwidths 3, 5, 7, 9, 11 and 15 were used as test systems. The nonzero elements were generated randomly by rows from -10 to +10 using the FTN library random number generator RANF (Control Data, 1971). An initial seed of one was used. The right hand side was adjusted to yield a vector of ones as the solution.

The banded systems were used mainly for convenience; none of the algorithms tested takes advantage of this structure. However, with banded matrices, no new elements will be generated during the solution; thus the testing does not reflect how well the methods handle this problem. The Bending-Hutchison method, however, does handle this easily.

Execution times (See Appendix E) for the various methods are shown in Tables 5.2, 5.3 and 5.4. Table 5.1 provides a key to the methods.

The conventional methods are obviously unsuitable for these problems. Matrix inversion takes much time and Gaussian elimination is quite slow. The iterative methods, Jacobi iteration and Gauss-Seidel iteration both diverged from initial estimates of zero. None of the conventional techniques were able to solve 200 equations as this required more than the 49K of core available.

Of the sparse routines, many gave inaccurate answers. This may be attributed to the fact that the non-zero elements were generated over a wide range, producing badly conditioned systems. If instead the elements are generated from 1-2, these programs work quite well. The execution times were independent of the spread of the non-zero elements.

Of the seven pivoting options given by Key, the best was minimum-row, minimum-column which Key also recommends.

The Bending-Hutchison routine, TRGB2, gave the best performance of the sparse methods. This is almost to be expected since all of the work of determining the solution process has already been done by TRGB. The method is ideal for repeated solution of linear equations.

There is another program developed at IBM by Gustavson, Liniger and Willoughby (1970) in which the first run generates the FORTRAN code to solve the specific linear system. Further solutions of the same system structure can use the FORTRAN code already produced (Calahan, 1968a). This process has been implemented along with a modified version of Gear's method (Brayton, Gustavson and Hachtel (1972); Hachtel, Brayton and Gustavson, 1971) into the ECAP II Package (Branin *et al.*, 1971)."

The version of TRGB-TRGB2 used in the testing stored the operator list completely in core. For 200 equations, the method could not store the list for bandwidth of 11 or more. The list can be put into auxiliary storage however, but this will require more computer time to transfer the list or parts of it to and from core. If this is done, there would be no limit to the number of equations which can be solved. Table 5.5 gives the length of the operator list for the test systems.

As the bandwidth increases, one of the IMP options, method 17 of Table 5.1 becomes more competitive. Possibly as the bandwidth increases still further, the IMP method will become even faster than TRGB2. This opinion is held by Brandon (Brandon, 1974e).

Other algorithms not included in the numerical testing were a conjugate gradient program of Reid (1970), an elimination algorithm of Curtis and Reid (1971) and a program based on arc-graph structure (Rheinboldt, 1973).

Table 5.1: Key To Linear Equation Solvers

NUMBER	METHOD
	CONVENTIONAL METHODS
1	(1) Matrix Inversion (MINV)
2	(2) Gaussian Elimination, No Pivoting (SOLVE)
3	(3) Gaussian Elimination, Partial Pivoting (DECOMP-SOLVE)
4	(4) Jacobi Iteration
5	(5) Gauss Seidel Iteration
	KEY
6	(1) Simple Gauss-Jordan Elimination
7	(2) Gauss-Jordan Partial Pivoting
8	(3) Gauss-Jordan Full Pivoting
9	(4) Minimum Row-Minimum Column
10	(5) Minimum Column-Minimum Row
11	(6) Maximum Column-Minimum Row
12	(7) Minimum Of Row Entries Times Column Entries
	IMP
13	(1) Gauss-Seidel
14	(2) Gradient
15	(3) Crout, Fixed Order, General Sparse
16	(4) Crout, Variable Order, General Sparse
17	(5) Crout, Variable Order, Variable Or Constant Band
	MISCELLANEOUS
18	(1) Brandon, Crout, Variable Order, General Sparse
19	(2) LINEQ4
20	(3) TRGB
21	(4) TRGB2

Table 5.2 Execution Times For 50 Linear Equations

NO.	BANDWIDTH						
	3	5	7	9	11	13	15
1	5.17	5.15	5.13	5.15	5.18	5.20	5.16
2	0.838	0.850	0.816	0.833	0.819	0.818	0.818
3	0.790	0.791	0.780	0.779	0.755	0.764	0.756
4	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
5	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
6	0.647	0.798	0.935	1.09	1.27	1.44	1.62
7	0.748	0.986	1.23	1.38	1.64	1.91	2.14
8	0.523	1.79	2.30	3.27	5.05	5.39	5.15
9	0.215	0.327	0.495	0.691^{I}	0.897	1.12^{I}	1.36^{SI}
10	0.293	0.437	0.612	0.814^{I}	1.06	1.31^{SI}	1.60
11	0.768	0.970	1.22	1.49	1.78	2.06	2.39
12	0.320	0.538	0.736	0.994^{I}	1.28	1.58^{SI}	1.88
13	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
14	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
15	0.112	0.161	0.243	0.373	0.537	0.771	1.02
16	0.101	0.158	0.223	0.298	0.391	0.509	0.645
17	0.047	0.060	0.080	0.092	0.111	0.131	0.138
18	0.073	0.097	0.161	0.234	0.320	0.424	0.564
19	0.960^{I}	1.35^{I}	1.68^{I}	2.04^{I}	2.46^{I}	2.97^{I}	3.52^{I}
20	0.185	0.268	0.346	0.453	0.550	0.674	0.797
21	0.012	0.023	0.033	0.049	0.065	0.086	0.106

I - inaccurate SI - slightly inaccurate (1-3 figure accuracy)

DIV - diverged

Table 5.3: Execution Times For 100 Linear Equations

NO.	BANDWIDTH						
	3	5	7	9	11	13	15
1	40.1	39.8	40.0	39.8	39.7	39.8	40.6
2	6.45	6.43	6.37	6.28	6.32	6.26	6.26
3	5.81	5.78	5.72	5.67	5.62	5.59	5.54
4	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
5	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
6	2.60	3.15	3.71	4.32	4.94	5.57	6.20
7	2.07	3.88	4.79	5.69	6.40	7.47	8.33
8	1.91	8.68	11.6	15.1	24.7	24.9	34.9
9	0.722[I]	1.05[I]	1.49[I]	1.95[I]	2.48[I]	3.10[I]	3.76[I]
10	1.02[I]	1.38[I]	1.84[I]	2.35[I]	2.95[I]	3.65[I]	4.37[I]
11	3.05	3.81	4.64	5.51	6.43	7.42	8.47
12	1.22	1.81	2.51	3.25	4.08	4.97	5.86
13	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
14	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
15	0.179	0.286	0.463	0.741	1.10	1.58	2.20
16	0.277	0.379	0.518	0.674	0.879	1.13	1.43
17	0.118	0.144	0.178	0.216	0.249	0.295	0.327
18	0.152	0.247	0.350	0.503	0.690	0.923	1.20
19	2.36[I]	3.87[I]	5.03[I]	6.40[I]	7.95[I]	9.74[I]	11.7[I]
20	0.651	0.933	1.24	1.54	1.88	2.25	2.67
21	0.023	0.044	0.068	0.101	0.138	0.181	0.229

I - inaccurate SI - slightly inaccurate (1-3 figure accuracy)
DIV - diverged

Table 5.4: Execution Times For 200 Linear Equations

NO.	BANDWIDTH						
	3	5	7	9	11	13	15
1	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.
2	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.
3	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.
4	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.
5	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.	T.L.
6	10.4	12.5	14.8	17.1	19.4	21.8	24.2
7	6.17	15.5	20.0	23.3	26.4	29.9	33.8
8	6.64	33.5	60.3	95.2	110.0	147.7	148.2
9	2.60[I]	3.57[I]	4.70[I]	5.98[I]	7.43[I]	9.00[I]	10.8[I]
10	3.85[I]	4.85[I]	6.05[I]	7.51[I]	8.97[I]	10.7[I]	12.7[I]
11	12.2	15.0	18.0[I]	21.1[I]	24.4[I]	27.7[I]	31.1[I]
12	4.56[I]	6.59[I]	8.81[I]	11.2[I]	13.7[I]	16.5[I]	19.3[I]
13	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
14	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.	DIV.
15	0.293	0.530	0.897	1.46	2.22	3.24	4.52
16	0.816	1.02	1.28	1.62	2.04	2.56	3.17
17	0.315	0.372	0.440	0.506	0.577	0.650	0.743
18	0.417	0.587	0.824	1.15	1.53	2.00	2.59
19	7.26[I]	13.2[I]	17.5[I]	22.7[I]	28.5[I]	35.1[I]	42.3[I]
20	2.45	3.47	4.51	5.59	T.L.	T.L.	T.L.
21	0.046	0.086	0.137	0.201	T.L.	T.L.	T.L.

I - inaccurate SI - slightly inaccurate (1-3 figure accuracy)

T.L. - too large (requires more than 49K) DIV - diverged

Table 5.5: Length of TRGB Operator List

BANDWIDTH	NUMBER OF EQUATIONS		
	50	100	200
3	600	1200	2400
5	1176	2376	4776
7	1928	3928	7928
9	2848	5848	11848
11	3928	8128	-
13	5160	10760	-
15	6536	13736	-

6. NUMERICAL TESTING OF STIFF TECHNIQUES

6.1 Testing By Other Workers

Several numerical comparisons of methods for solving stiff o.d.e.s have been made.

Steeper (1970) compares Sloate's method (Sloate, 1970), Liou's method (Liou, 1966), Pope's method (Pope, 1963), second and fourth order Runge-Kutta, Jung's variation of Treanor's method (Jung, 1967), a linear extrapolation method from the FACE program (Price, 1970) and MINTA5, a modified version of Adams-Moulton-Shell (Shell, 1958). Based on two test examples, he recommends MINTA5.

Lapidus and Seinfeld (1971), also Seinfeld, Lapidus and Hwang (1970), have compared fourth order Runge-Kutta, fourth order Adams, Treanor's method (Treanor, 1966), the modified midpoint rule, the trapezoidal rule with and without extrapolation, Calahan's method (Calahan, 1968b) and two methods of Liniger and Willoughby (1970). They found the last four implicit methods to be superior and roughly comparable in terms of accuracy and computing time. For highly stiff systems, they recommend one of Liniger and Willoughby's techniques. Four test examples were used.

Brandon (1972, 1974f) compares his own method with the stiff techniques compared by Lapidus and Seinfeld and finds his method to be superior. He uses two test examples.

Hronsky and Martens (1973) compare Runge-Kutta-Newton (Kreyszig, 1967) which is the pseudo steady state approach used with the fourth order Runge-Kutta method, Treanor's method (Treanor, 1966), a method of Liniger and Willoughby (1970), and the backward differentiation formula (Brayton *et al.*, 1972) which is a modification of Gear's method (Appendix A). Based on five test examples, they recommend the methods in the order above, except if the number of equations is less than six, Liniger and Willoughby's method is recommended over that of Treanor.

Hull and Enright have been doing extensive comparisons of stiff techniques, but their final results are not yet available. Their preliminary testing indicated the three best methods to be Gear's method, trapezoidal rule with extrapolation and the second derivative multistep method (Enright, 1974).

Several other comparisons appear in the literature but, unfortunately, most authors of new techniques compare their algorithm only with an inefficient conventional method such as fourth order Runge-Kutta or with a stiff technique very similar to theirs, to demonstrate a

superior modification.

6.2 Test Methods and Examples

The following methods were selected for numerical testing:

four conventional methods none of which are oriented to solving stiff systems:

(1) Euler

(2) Adams-Moulton-Shell (Shell, 1958)

(3) Runge-Kutta-Merson (Merson, 1957)

(4) Gear's nonstiff option (Appendix A)

four explicit stiff techniques:

(1) Richards, Lanning and Torrey (1965)

(2) Nigro (1969)

(3) Treanor (1966)

(4) Fowler and Warten (1967)

four implicit stiff techniques:

(1) Klopfenstein and Davis (1971)

(2) Sandberg and Shichman (1968)

(3) Brandon (1972, 1974f)

(4) Gear's stiff option (Appendix A).

The IMP package (Appendix B) containing Brandon's method was also tested. Obviously it is impractical to test all the methods surveyed in Chapter 3, but these provide a cross-section of promising explicit and implicit techniques. The explicit methods have the advantage of avoiding the solution of linear algebraic equations.

The four implicit methods used matrix inversion (MINV of Chapter 5) in solving the corrector. Gear's method was also used with two other linear equation solvers: DECOMP-SOLVE and TRGB-TRGB2 described in Chapter 5. TRGB was used only on the first time step to set up the operator list and TRGB2 used thereafter.

Eleven test systems were used (Appendix D). They can be divided into three groups:

(1) small stiff systems: I, II, III, IV, V

(2) large stiff systems: VI, VII

(3) nonstiff systems : VIII, IX, X, XI

Each of the fifteen test programs were tested on each example. Ideally the same method could handle both nonstiff and stiff methods efficiently; otherwise, two methods would have to be used.

The two complex eigenvalue systems III and VIII had small imaginary parts to avoid introducing the problem

of simulating rapidly oscillating components.

Most of the methods used had a local truncation error estimate: Gear, IMP, Brandon, Fowler-Warten, Klopfenstein-Davis, Sandberg-Shichman, Adams-Moulton-Shell and Merson. Gear's error control is described in Gear (1971a,b) and briefly in Appendix A. IMP uses a complex algorithm (Brandon, 1972, 1974f) to calculate the step size. In all of these methods, however, the maximum relative local error was kept below an upper tolerance and usually above a lower tolerance (except for Gear and IMP). The upper tolerance used was 10^{-3} for systems I-X and 10^{-6} for system XI. The lower tolerance was taken as one-tenth of the upper tolerance. Relative errors were used with the absolute error estimate divided by the maximum value of the corresponding independent variable. This maximum value had a minimum of 1.0, so that in effect, relative errors were used for large y values and absolute errors for small y values. Fowler-Warten had a complex error control scheme, but it was not used here. The last six methods mentioned above used the double-halving approach described in Section 2.7.3.

Three methods had no error estimate: Euler, Nigro and Treanor. Lapidus and Seinfeld (1971) recommend the following error control scheme for Treanor's method. If the relative change in y, $|y_{n+1}-y_n|/|y_{n+1}|$, is greater than

some upper tolerance, the step size is halved and the step repeated. If it is greater than some lower tolerance, the step is doubled. Tolerances of 0.05 and 0.005 were used. This strategy gave unstable results for systems V, VI and VII. It was altered for them to keeping

$$0.001 < \left| \frac{y_{n+1}-y_n}{y_{n+1} + R} \right| < 0.01$$

where $R = 0.1$ if $y_{n+1} < 0.5$

$= 0.0$ otherwise

Euler and Nigro used a fixed step size. The critical step size was used for the stiff examples, i.e., the step size just before the onset of instability. For the nonstiff examples, the step size was adjusted to keep the final error below 1%.

For Nigro's method, the parameters corresponding to $h\lambda = 120$ were used. Euler's method was used to start the integration. A step size of 0.01 of the Nigro step size was used to calculate the starting points. This was not included in the execution time.

The first order Klopfenstein-Davis algorithm tested better than the second order. Tests with various coefficients of α, β, u, v and α yielded very similar results so the values $\alpha = 1.0$, $\beta = 0.0$, $u = 0.25$, $v = 0.75$ and $\alpha = 0.6667$ were used. See Equation (3.3.10).

6.3 Test Results

Global errors are given at the end of the simulation with the exception of systems II, V, VI and VII. There, the error is given after 10% of the simulation, since it was expected several methods could require near infinite computer time to reach the end of the integration.

Analytical solutions are available on systems I, II, III, IV, VIII and IX to calculate the error. For the remaining examples, V, VI, VII, X and XI, the Euler method with $h = 0.00005$ was used to get an accurate solution. This was arbitrarily taken as the "correct" solution. In each case with $h = 0.0001$, the results differed at worst in the fourth significant figure.

None of the examples came close to steady state, so an error taken at the end of the simulation will be reasonably indicative of the accuracy of the trajectory.

Table 6.1 shows the execution times and errors for the explicit methods and the Euler step size. Tables 6.2 and 6.3 show results for the explicit and implicit stiff methods respectively. Nigro's step size is also shown. Table 6.4 shows IMP and three different linear equation solvers in conjunction with Gear's method. Gear-MINV is repeated from Table 6.3. See Appendix E for a discussion on computer timings.

In general the conventional methods did not perform well on the stiff examples. As expected, Euler's method was the least inefficient, but obviously a method designed specifically for stiff systems is to be preferred. Some of the runs especially on systems II, V, VI and VII were a waste of computer time, as the methods were so inefficient. The conventional methods are not of great concern here except to note how well the stiff methods were able to solve the nonstiff examples by comparison. They do reasonably well with respect to execution time and error. Since the execution times are so small anyway, it would suffice to use a stiff method for both nonstiff and stiff examples.

Hull *et al.* (1972a) consider a method of Krogh (1969) very good for nonstiff systems, but they rate Gear's (nonstiff option) closely behind it. In our limited testing of nonstiff examples here, Runge-Kutta-Merson did very well. AMOS could not handle system XI, probably because of the extremely small tolerance demanded.

In general, the explicit stiff methods did not perform as well as the implicit methods. Of the stiff methods, Gear's was easily the best (Table 6.3). Brandon (1974d) has improved his error estimation procedure recently and it does much better, but it was not used in our testing. In particular, the latest version of the IMP package

integrates system VI and VII in 22.2 and 6.68 seconds respectively on the CDC 6600. These times would roughly be multiplied by three for comparable CYBER 73 times.

The Fowler-Warten method handled the complex eigenvalue systems III and VIII reasonably well in spite of the fact that it is intended for systems with the largest eigenvalue real.

Neither of the step-changing strategies used with Treanor's method were very efficient. Possibly a fixed step size would be a better strategy.

Some of the methods simulated only the extremely small components inaccurately especially in Systems V and X. In these cases, the second largest error is given in brackets.

Different methods of handling the corrector are illustrated in Table 6.4 with Gear's method. For less than about five equations, matrix inversion is fastest, then Gaussian elimination is preferable up to perhaps ten equations, but for large systems a sparse method such as the Bending-Hutchison algorithm is necessary, as shown by the results for Systems VI and VII. The difference in execution time is slight for the small examples.

System XI ran slightly faster with the tridiagonal option (0.753 seconds) rather than with TRGB-TRGB2 (0.980 seconds). The time savings will increase with the size of the system as shown in Section 8.4.

If we had to choose one method to handle all systems stiff and nonstiff, large and small, it would be Gear's method with stiff option and the Bending-Hutchison method to solve the corrector. It handles small stiff systems and nonstiff systems efficiently enough since little computer time is required anyway. However, the nonstiff option is also available in the same program and only a switch is required to change from one to the other, so that both stiff and nonstiff options are conveniently available. Different coefficients are used, but most of the code is similar for both options.

Table 6.1: Execution Times And Errors For Conventional Methods

TEST SYSTEM \ METHOD	EULER	STEP SIZE	ERROR	AMOS	ERROR	RKM	ERROR	GEAR NONSTIFF	ERROR
I	0.066	1.9×10^{-3}	9.50×10^{-4}	0.872	1.88×10^{-3}	0.301	3.72×10^{-4}	2.30	1.13×10^{-3}
II	71.7	1.9×10^{-6}	9.51×10^{-8}	866.0	1.01×10^{-3}	300.3	1.36×10^{-4}	2188.0	3.81×10^{-4}
III	0.325	1.9×10^{-3}	4.28×10^{-3}	2.06	4.29×10^{-3}	1.18	3.55×10^{-3}	4.90	4.05×10^{-3}
IV	1.47	1.9×10^{-3}	9.62×10^{-6}	9.09	5.65×10^{-4}	6.02	7.27×10^{-6}	36.4	2.89×10^{-5}
V	17.1	5.0×10^{-4}	3.91×10^{-5}	240.0	114.6 (4.70×10^{-3})	79.0	154.6 (5.21×10^{-4})	431.6	0.368 (3.95×10^{-4})
VI	U[1] 23.0 (0-10)	2.5×10^{-4} 1.0×10^{-3}	4.21×10^{-4}	>1500 101.0 (0-10)	9.95×10^{-3}	>1000 48.0 (0-10)	1.99×10^{-4}	1925.0	1.86×10^{-4}
VII	37.6	6.0×10^{-3}	4.55×10^{-3}	353.0	0.855[2]	167.8	7.94×10^{-2}	656.0	8.80×10^{-3}
VIII	0.530	2.0×10^{-3}	9.58×10^{-3}	0.059	6.69×10^{-3}	0.021	1.89×10^{-3}	0.101	8.15×10^{-4}
IX	1.04	5.0×10^{-3}	9.48×10^{-3}	0.107	9.62×10^{-3}	0.041	1.56×10^{-2}	0.213	6.67×10^{-3}
X	0.636	2.0×10^{-3}	9.75×10^{-3}	0.059	3.48 (1.81×10^{-4})	0.022	0.394 (6.05×10^{-4})	0.164	4.39×10^{-2}
XI	1.39	2.5×10^{-3}	9.45×10^{-3}	>320		0.263	4.87×10^{-4}	1.30	2.25×10^{-4}

[1] unstable

[2] only extremely small components inaccurate

Table 6.2: Execution Times And Errors For Explicit Stiff Techniques

TEST SYSTEM \ METHOD	RLT	ERROR	NIGRO	STEP SIZE	ERROR	TREANOR	ERROR	FOWLER WARTEN	ERROR
I	0.092	4.83×10^{-3}	0.144	1.8×10^{-3}	5.35×10^{-3}	0.384	2.24×10^{-4}	0.495	2.24×10^{-3}
II	0.068	5.03×10^{-4}	26.5	1.0×10^{-5}	2.15×10^{-2}	0.393	8.90×10^{-7}	49.1	3.98×10^{-2}
III	3.32	3.81×10^{-3}	0.498	1.8×10^{-3}	6.13×10^{-3}	1.38	5.09×10^{-4}	1.48	7.49×10^{-5}
IV	0.356	5.03×10^{-3}	0.817	5.0×10^{-3}	6.88×10^{-3}	6.90	1.85×10^{-3}	0.317	1.79×10^{-2}
V	0.078	343.0 (6.58×10^{-4})	18.5	1.0×10^{-3}	1.06×10^{-2}	0.505	3.56×10^{-3}	0.624	8.42×10^{-3}
VI	36.5	8.86[2]	464.0	1.0×10^{-3}	3.35×10^{-2}	U[1] 44.5 (0-10)	4.41×10^{-2}	117.0	2.48×10^{-2}
VII	3.58	2.03[2]	46.5	1.0×10^{-2}	1.73×10^{-2}	219.0	8.24×10^{-2}	75.2	1.41×10^{-2}
VIII	0.159	8.61×10^{-2}	1.24	2.0×10^{-3}	9.72×10^{-3}	0.927	1.97×10^{-5}	0.080	2.64×10^{-2}
IX	0.317	5.45×10^{-2}	2.02	4.0×10^{-3}	1.05×10^{-2}	2.69	3.13×10^{-9}	0.977	7.07×10^{-3}
X	0.118	0.630 (2.81×10^{-2})	1.36	2.5×10^{-3}	2.64×10^{-2}	2.36	2.46×10^{-4}	0.060	1.83×10^{-3}
XI	0.744	3.00×10^{-2}	5.65	1.0×10^{-2}	1.63×10^{-2}	5.07	4.06×10^{-5}	0.666	1.86×10^{-4}

[1] unstable

[2] large components reasonably accurate

Table 6.3: Execution Times And Errors For Implicit Stiff Techniques

TEST SYSTEM \ METHOD	KLOP DAVIS	ERROR	SAND SCH.	ERROR	BRANDON	ERROR	GEAR	ERROR
I	2.27	5.85×10^{-4}	0.274	1.21×10^{-2}	0.124	1.22×10^{-3}	0.139	1.07×10^{-3}
II	>1000 385.0 (0-10)	5.00×10^{-8}	0.276	7.10×10^{-4}	0.268	3.98×10^{-4}	0.196	1.01×10^{-4}
III	7.11	2.67×10^{-3}	0.788	4.64×10^{-2}	4.24	3.97×10^{-3}	0.318	3.77×10^{-3}
IV	39.2	5.90×10^{-6}	1.75	9.41×10^{-4}	1.11	5.66×10^{-4}	0.505	1.16×10^{-5}
V	845.0	0.267 (7.56×10^{-3})	0.117	1.27×10^{-4}	0.459	3.10×10^{-5}	0.168	4.57×10^{-5}
VI	>1000 (0-10)		>1000 626.0 (0-10)	6.83×10^{-3}	>1500 761.0 (0-10)	7.72×10^{-4}	81.7	8.47×10^{-3}
VII	>1000 (0-10)		442.0	0.133	299.0	1.86×10^{-2}	42.0	1.71×10^{-2}
VIII	0.760	0.206	0.721	0.204	0.294	0.308 (0.023)	0.078	3.23×10^{-3}
IX	0.788	0.335	0.830	0.316	0.546	9.68×10^{-3}	0.192	8.07×10^{-2}
X	0.473	0.936 (3.0×10^{-2})	0.455	0.870 (2.94×10^{-2})	0.313	8.13×10^{-3}	0.119	1.38 (2.74×10^{-3})
XI	29.1	3.78×10^{-2}	29.1	3.75×10^{-2}	2.44	1.77×10^{-4}	1.03	4.85×10^{-5}

Table 6.4: Execution Times And Errors For Gear's Method And IMP

TEST SYSTEM \ METHOD	GEAR MINV	GEAR DECOMP-SOLVE	GEAR TRGB	ERROR	IMP	ERROR
I	0.139	0.148	0.162	1.07×10^{-3}	0.139[1]	1.85×10^{-3}
II	0.196	0.216	0.235	1.01×10^{-4}	0.373[1]	2.84×10^{-4}
III	0.318	0.294	0.315	3.77×10^{-3}	3.72[1]	1.06×10^{-5}
IV	0.505	0.489	0.656	1.16×10^{-5}	0.967[1]	2.69×10^{-2}
V	0.168	0.165	0.182	4.57×10^{-5}	27.3[2]	3.91×10^{-5}
VI	81.7	20.0	7.58	8.47×10^{-3}	>1000	
VII	42.0	12.3	6.05	1.71×10^{-2}	30.2[2]	0.169
VIII	0.078	0.079	0.080	3.23×10^{-3}	0.258[1]	0.344
IX	0.192	0.197	0.292	8.07×10^{-2}	0.584[2]	2.37×10^{-3}
X	0.119	0.128	0.130	1.38 (2.74×10^{-3})	1.12[1]	5.27×10^{-3}
XI	1.03	0.897	0.980 0.753(TR1)	4.85×10^{-5}	11.5[1]	0.967

[1]Crout elimination, variable order, variable or constant banded matrices

[2]Gauss-Seidel

7. THE DYNSYS 2.0 EXECUTIVE PROGRAM

We have tested several integration techniques for handling stiff ordinary differential equations. Of these, Gear's method appears to be the most effective. We have also tested various techniques for solving large systems of linear algebraic equations which arise in solving the implicit part of a stiff numerical integration method. The Bending-Hutchison approach was found to be particularly appropriate.

We would now like to implement these features into a dynamic simulation executive program. Either the equation-oriented or problem-oriented approaches can be used, but we believe the modular approach is especially suited to the engineer, since many engineering processes are modular in structure; i.e., they consist of many different pieces of equipment. It was decided to implement the new techniques into the DYNSYS package. The resulting program is called DYNSYS 2.0.

This chapter describes the new features of DYNSYS 2.0 and how they affect the user. A listing of DYNSYS 2.0 appears in Appendix I.

7.1 Numerical Integration of O.D.E.s

Both the stiff and nonstiff options of Gear's method are available; however, the same option must be used for the entire simulation.

A simulation will often contain both stiff and nonstiff modules. In this case the stiff option must be used and all modules will use the stiff coefficients of Gear's method. We saw in Chapter 6 that the stiff option handles nonstiff problems reasonably well. It is not possible to use the nonstiff and stiff coefficients in the same simulation as they have different error constants and the error and step-changing analysis could not be done.

The stiff option is assumed by the executive. If the user wishes to have the nonstiff system, i.e., the system is completely nonstiff, he must specify this through the data set. Section 2.6 describes how to identify a stiff system.

The algorithm used differs from Gear's original DIFSUB (Appendix A) because of the modular approach. A predictor step is taken through all of the modules followed by a corrector step. Up to three corrector iterations are made. The derivatives are re-evaluated by returning to the module. If convergence does not occur for any module, the integration step size is reduced to 1/4 of its present size and the step repeated for all modules. The factor 1/4

is given by Gear (1971c) and is empirical.

Another way of using the predictor-corrector would be to predict and correct for each module in turn. However, the manner actually chosen helps to converge any recycle in the system. As yet DYNSYS has no other way of handling recycle convergence. This is a possible item for future work.

After the corrector pass, the error analysis is done (Gear, 1971c; Appendix A) considering the equations from every module as one system.

7.2 Corrector Convergence of Stiff Equations

Gear's algorithm contains an option for both stiff and nonstiff equations. If the nonstiff option is used, the corrector equations will be solved by repeated iteration only. For the stiff option, Newton-Raphson iteration is generally used, but a special option for tridiagonal matrices (Carnahan, Luther and Wilkes, 1969) was included. This was done because tridiagonal systems arise often in chemical engineering in the discretization of partial differential equations and in countercurrent stagewise processes. A special algorithm for the tridiagonal system is much faster than using the Bending-Hutchison solver. See example #4 of Chapter 8.

In a system of nonstiff and stiff modules, all the modules will use the stiff coefficients of Gear's method to facilitate the error analysis, but the nonstiff modules can use Jacobi iteration of the corrector.

Each set of stiff o.d.e.s in each module will generate an operator list. All the lists are created on the first time step and are stored sequentially in core with 10% extra storage in case the list is re-created.

The routine contains a minimum pivot presently set at 10^{-6} which is reasonable for 14 figure single precision accuracy as on CDC machines. This figure may be altered through the data set for other computers (for example, perhaps 10^{-3} for single precision IBM computers). If, during the integration, one of the pivots falls below 10^{-6}, the operator list will be re-calculated using different elements as pivots. It is assumed that the new operator list will not be more than 10% longer than the original one. Several tests were made where the operator list was re-calculated every time step. The greatest change in length was 5% longer than the original length; however, in every example tested, the original operator list was sufficient for the entire simulation. Different operator lists can be generated as the integration proceeds, in addition to the possibility of a pivot falling below 10^{-6}, because at $t = 0$, many nonzero elements of the

Jacobian may actually be zero because of the initial conditions, their values may become large enough that they are available as pivots. The present executive does not, however, re-evaluate the operator list for this reason as it is unnecessary and would require too much computer time.

DIFSUB can use a numerical or analytical (user supplied) Jacobian matrix. It was decided to use an analytical Jacobian for DYNSYS as it is more efficient and it would be extremely difficult to compute a sparse numerical Jacobian and use it with the Bending-Hutchison routine.

Gear re-evaluated the Jacobian only if the corrector did not converge in three iterations or less or if a change in order was made. The present method in DYNSYS evaluates the Jacobian every time step to avoid storing all the Jacobians together in core at the same time since storage space is critical.

The DYNSYS version of DIFSUB has about the same execution speed as Gear's published version. The Jacobian is evaluated more often, but this is balanced by fewer corrector iterations being required.

7.3 Using the DYNSYS 2.0 Executive Program

7.3.1 New Parameters in the DYNSYS 2.0 Data Set

Several new data set parameters relating to the new integration method have been introduced. They are the smallest and largest permissible integration step size (HMIN, HMAX), the option, stiff or nonstiff to be used (NONSTIFF), the maximum permissible order of the integration (ORDER), the error tolerance (EPS), and the minimum permissible pivot element (MINPIVOT). These are summarized in Table 7.1. The other data set parameters are described in the DYNSYS Manual.

7.3.2 Writing Modules For DYNSYS 2.0

The user will usually write a module corresponding to each of his pieces of equipment. A general skeleton of such a module is shown in Figure 7.1. The user should study this and the examples in Chapter 8 before he writes his modules.

As shown in the skeleton, a module can be divided into five basic sections:

(1) calculate module parameters, stream input, initial conditions, ITER, etc.

Table 7.1: New Parameters In the DYNSYS 2.0 Data Set

DATA INPUT	DEFAULT VALUE	DESCRIPTION
HMIN	HMIN=10^{-6}	HMIN: smallest permissible step size to be used by DIFSUB
HMAX	HMAX=0.05	HMAX: largest permissible step size to be used by DIFSUB
NONSTIFF	ISTIFF=1	ISTIFF=0: nonstiff option, i.e., nonstiff coefficients are used in Gear's method ISTIFF=1: stiff option
ORDER	IORDER=6	IORDER: maximum permissible order of integration method to be used IORDER cannot exceed 6 for stiff option or 7 for nonstiff option
TOLERANCE	EPS=0.001	EPS: Euclidean norm of relative local truncation errors
MINPIVOT	XMIN=10^{-6}	XMIN: minimum permissible value of pivot element in TRGB, TRGB2

FIGURE 7.1: SKELETON OF MODULE FOR DYNSYS 2.0

```
      SUBROUTINE TYPE2.0
C
C     COMMENTS DESCRIBING MODULE
C
C     THE FOLLOWING COMMON BLOCKS ARE OR MAY BE REQUIRED:
C     MAT,CON,PTAB,UNIT,GEAR2,MODULE,ROW,COLUMN,JACOB,SUBDI,DIAG,SUPERD
C
      COMMON/MAT/MP( , ),EP( , ),S( , , ),EX( )
      COMMON/CON/IG,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON/PTAB/IGFLAG,PP( , )
      COMMON/UNIT/IM,NMP
      COMMON/GEAR2/EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF
      COMMON/MODULE/IDERY,ITER,ITRI,NZERO
C
C     NZERO IS NUMBER OF NONZERO ELEMENTS IN JACOBIAN
C
      COMMON/ROW/IROW(NZERO)          (NORMAL OPTION)
      COMMON/COLUMN/JCOL(NZERO)       (NORMAL OPTION)
      COMMON/JACOB/XJACOB(NZERO)      (NORMAL OPTION)
C
C     N IS NUMBER OF ODES
C
      COMMON/SUBDI/A(N)      (TRIDIAGONAL OPTION)
      COMMON/DIAG/B(N)       (TRIDIAGONAL OPTION)
      COMMON/SUPERD/C(N)     (TRIDIAGONAL OPTION)
C
C*************************************************************************
C                  SECTION #1 : PARAMETER CALCULATIONS
C*************************************************************************
C
C     CALCULATE MODULE PARAMETERS : STREAM INPUT,INITIAL CONDITIONS,
C                                   VALUES OF ITER ETC.
C
C     INPUT STREAM INFORMATION IS OBTAINED FROM S(IG, , )
C     VALUE OF INDEPENDENT VARIABLE (Y) NEED ONLY BE SPECIFIED ON FIRST
C        INTEGRATION STEP (I.E. INITIAL CONDITIONS)
C     JSTART=0 ON FIRST INTEGRATION STEP
C           =CURRENT ORDER OF INTEGRATION TECHNIQUE ON LATER STEPS
C
C     IF ISTIFF=0,NONSTIFF COEFFICIENTS WILL BE USED IN INTEGRATION ALGORITHM
C                 (DIFSUB) FOR ALL MODULES,
C                 JACOBIAN MATRIX IS NOT REQUIRED
C
C     IF ISTIFF=1,STIFF COEFFICIENTS WILL BE USED FOR ALL MODULES
C                 THEN,IF ITER=0 DIRECT ITERATION OF CORRECTOR WILL BE USED
C                                (NONSTIFF MODULE)
C                                JACOBIAN IS NOT REQUIRED
C                      IF ITER=1 NEWTON-RAPHSON ITERATION OF CORRECTOR WILL
C                                BE USED (STIFF MODULE)
```

```
C                              JACOBIAN MATRIX MUST BE SUPPLIED
C
C     ITER MUST BE SPECIFIED 0 OR 1
C     IT IS NOT USED UNLESS ISTIFF=1
C
      IF(IG.EQ.2) GO TO 2
C
C*****************************************************************************
C                    SECTION #2 : JACOBIAN EVALUATION
C*****************************************************************************
C
C     SECTION #2 IS OMITTED FOR NONSTIFF MODULE
C
C     JACOBIAN MATRIX IS REQUIRED ONLY IF MODULE EQUATIONS ARE STIFF
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS ONLY
C     JACOBIAN NEED NOT BE VERY ACCURATE,AS IT IS USED ONLY FOR CONVERGENCE
C        OF CORRECTOR
C     ONLY NONZERO ELEMENTS ARE CALCULATED,BUT ALL DIAGONAL ELEMENTS MUST
C        BE STORED WHETHER OR NOT THEY ARE ZERO (NORMAL OPTION)
C
C              NORMAL OPTION
C
C     IF JACOBIAN MATRIX IS NOT TRIDIAGONAL
C     ROW NUMBER,COLUMN NUMBER AND VALUE OF EACH NONZERO ELEMENT ARE STORED IN
C        ARRAYS IROW,JCOL AND XJACOB FROM COMMON BLOCKS ROW,COLUMN AND JACOB
C        RESPECTIVELY
C     ELEMENTS MAY BE STORED IN ANY ORDER
C
C     NZERO=     - NUMBER OF NONZERO ELEMENTS IN JACOBIAN
C                  INCLUDING ANY ZERO DIAGONAL ELEMENTS
C     IROW(I)    - ROW NUMBER OF NONZERO ELEMENT I
C     JCOL(I)    - COLUMN NUMBER OF NONZERO ELEMENT I
C     XJACOB(I)  - VALUE OF NONZERO ELEMENT I
C
      NZERO=
      IROW(1)=
      JCOL(1)=
      XJACOB(1)=
          .
          .
          .
      IROW(NZERO)=
      JCOL(NZERO)=
      XJACOB(NZERO)=
C
C             TRIDIAGONAL OPTION
C
C     FOR TRIDIAGONAL JACOBIAN MATRIX,THE SUBDIAGONAL,DIAGONAL AND SUPERDIAGONAL
C        ELEMENT VALUES ARE STORED IN ARRAYS A,B AND C FROM COMMON BLOCKS SUBDI,
C        DIAG AND SUPERD RESPECTIVELY
C
C     N - NUMBER OF ODES
```

```
C      A - VALUES OF SUBDIAGONAL ELEMENTS ARE STORED IN A(2)...A(N)
C          A(1) IS NOT USED
C      B - VALUES OF DIAGONAL ELEMENTS
C      C - VALUES OF SUPERDIAGONAL ELEMENTS ARE STORED IN C(1)...C(N-1)
C          C(N) IS NOT USED
C
       B(1)=
       C(1)=
       A(2)=
       B(2)=
       C(2)=
          .
          .
          .
       A(N-1)=
       B(N-1)=
       C(N-1)=
       A(N)=
       B(N)=
     1 CONTINUE
C
C      IF TRIDIAGONAL OPTION IS BEING USED,ITRI MUST BE SET TO 1 HERE,
C      IF NORMAL OPTION IS USED ITRI MAY BE IGNORED
C
C      ITRI=1
C
     2 CONTINUE
C
C*****************************************************************************
C                    SECTION #3 : DERIVATIVE CALCULATION
C*****************************************************************************
C
C      CALCULATE DERIVATIVES
C
       DERY(1)=
          .
          .
          .
       DERY(N)=
C
C*****************************************************************************
C                    SECTION #4 : CALL DIFSUB
C*****************************************************************************
C
C      CALL DIFSUB TO SOLVE ODES FOR MODULE
C      DIFSUB MAY BE CALLED MORE THAN ONCE FROM A MODULE FOR EXAMPLE IF THE
C         MODULE CONTAINS A SET OF STIFF O.D.E.S AND ANOTHER SET OF NONSTIFF
C         O.D.E.S
C
C      ARGUMENTS :  N    - NUMBER OF ODES
C                   Y    - INDEPENDENT VARIABLE
C                   DERY - DERIVATIVES
```

```
C
      CALL DIFSUB(N,Y,DERY)
C
C     IF IDERY IS NOT ZERO,THE DERIVATIVES WILL BE RE-EVALUATED AND
C        RETURNED TO DIFSUB
C     ITRI MUST ALSO BE RESET IF IT IS 1
C
      IF(IDERY.NE.0) GO TO 1
C
C*****************************************************************************
C                    SECTION #5 : STREAM OUTPUT CALCULATION
C*****************************************************************************
C
C     CALCULATE STREAM OUTPUT (STORED IN S(1, , ))
C
      S(1, , )=
          .
          .
          .
      RETURN
      END
```

(2) calculate Jacobian matrix (if o.d.e.s are stiff) on corrector pass only

(3) evaluate derivatives

(4) CALL DIFSUB to solve o.d.e.s for module

(5) calculate stream output.

There are several variables used in the skeleton module which relate to the integration. These are summarized in Table 7.2.

It is not necessary to supply a Jacobian matrix for the nonstiff modules. Setting the variable ITER to zero inside the module will cause direct iteration of the corrector. ITER should be set to one for stiff modules so that Newton-Raphson iteration will be used and the Jacobian of course must be supplied. If the module is known to be always stiff or nonstiff, ITER can be specified inside the module; otherwise, an equipment parameter can be used to specify the option.

It is possible to have more than one set of o.d.e.s in a module. This will present no problem as long as each set is defined before the call to DIFSUB.

It may be necessary to use a low value of HMAX, the maximum permissible integration step size, to keep the system under control. If too large a step is used, say

Table 7.2: Summary Of Module Variables Relating To DIFSUB

VARIABLE	DESCRIPTION
A	values of subdiagonal elements of a tridiagonal Jacobian matrix are stored in A(2),...,A(N)
B	values of diagonal elements of a tridiagonal Jacobian matrix are stored in B(1),...,B(N)
C	values of superdiagonal elements of a tridiagonal Jacobian matrix are stored in C(1),...,C(N-1)
DERY	derivative values
IDERY	IDERY is tested immediately after call to DIFSUB IDERY=0 : module execution is continued IDERY=1 : derivatives are re-evaluated and returned to DIFSUB, ITR1 is reset to 1 if necessary
IROW	row numbers of nonzero Jacobian elements and zero diagonal elements in normal option for storing Jacobian
ITER	ITER is used only if ISTIFF=1 (stiff option) ITER=0 : direct iteration of corrector (nonstiff module) ITER=1 : Newton-Raphson iteration of corrector (stiff module)
ITR1	ITR1 specifies option for storing Jacobian ITR1=0 : normal option ITR1=1 : tridiagonal option if ITR1=1, it must be reset to 1 each time DIFSUB is called
JCOL	column numbers of nonzero Jacobian elements and zero diagonal elements in normal option for storing Jacobian
N	number of ordinary differential equations in module
NZERO	number of nonzero elements in Jacobian for normal option (zero diagonal elements are included)
XJACOB	values of nonzero Jacobian elements in normal option
Y	independent variable

near steady state, the truncation error will be acceptable, but the corrector may not converge and thus the step size will be reduced by a factor of four. This may happen repeatedly if HMAX is too large, for both stiff and nonstiff systems.

Problems may also occur if the variables being simulated have widely varying values. If an important variable has a value of the order 10^{-6}, then EPS, the error tolerance, must be set to about 10^{-9}. This can use a great deal of computer time for the other variables. It may be necessary to scale the mathematical model so that there are no important variables with extremely small value.

DYNSYS 2.0 was generally tested with optimization level one of the CYBER 73 FTN compiler. Using the RUN compiler (a nonoptimizing compiler), erroneous results were obtained on some examples probably because of excessive round-off from the unoptimized code. Thus the user should ensure that an optimizing compiler should be used. Both DIFSUB and TRGB-TRGB2 should have at least twelve figure accuracy in the computer word length. Excessive round-off may affect the results, if a smaller word size is used.

7.3.3 Jacobian Evaluation

If the module o.d.e.s are stiff (Section 2.6), the Jacobian matrix must be supplied in the module. There are two ways of accomplishing this: the normal option for a general Jacobian matrix and a special option for a tri-diagonal Jacobian. The Jacobian does not have to be extremely accurate as it is needed only for the convergence of the corrector; for example, if the Jacobian is diagonally dominant, it may suffice to use only the diagonal coefficients.

7.3.3.1 *Normal Option*

The row number, column number and value of each nonzero element are stored in arrays IROW, JCOL and XJACOB respectively. The elements may be stored in any order. For example the matrix:

$$\underset{\approx}{J} = \begin{bmatrix} 1.0 & -2.0 \\ 3.0 & 4.0 \end{bmatrix} \qquad (7.3.1)$$

would be stored as:

```
NZERO   =  4
IROW(1)  =  1
JCOL(1)  =  1
XJACOB(1)  =  1.0
IROW(2)  =  1
JCOL(2)  =  2
XJACOB(2)  =  2.0
IROW(3)  =  2
JCOL(3)  =  1
XJACOB(3)  =  3.0
IROW(4)  =  2
JCOL(4)  =  2
XJACOB(4)  =  4.0
```

NZERO is the number of nonzero elements in the matrix. Note also that a Jacobian element must be created for all diagonal elements whether or not they are nonzero.

Figure 8.7 depicts a simple example of a module using the normal option.

7.3.3.2. *Tridiagonal Option*

The subdiagonal, diagonal and superdiagonal element values are stored in arrays *A*, *B* and *C* respectively. *A(1)* and *C(N)* are not used. For example:

$$\underset{\sim}{J} = \begin{bmatrix} 1.0 & 2.0 & & & \\ 3.0 & 4.0 & 5.0 & & \\ & 6.0 & 7.0 & 8.0 & \\ & & 9.0 & 10.0 & 11.0 \\ & & & 12.0 & 13.0 \end{bmatrix} \qquad (7.3.2)$$

is stored as

$$\begin{array}{llll} & B(1) = 1.0 & C(1) = 2.0 & \\ A(2) = 3.0 & B(2) = 4.0 & C(2) = 5.0 & \\ A(3) = 6.0 & B(3) = 7.0 & C(3) = 8.0 & \\ A(4) = 9.0 & B(4) = 10.0 & C(4) = 11.0 & \\ A(5) = 12.0 & B(5) = 13.0 & & \end{array}$$

The variable ITR1 must be set equal to 1 each time DIFSUB is called.

Figure 8.14 depicts a simple example of a module using the tridiagonal option.

7.3.4 Error Messages

DYNSYS 2.0 prints several error messages related to the numerical integration. The program continues for messages 1-3, but execution stops for messages 4-8.

(1) TRUNCATION ERROR IS TOO LARGE

This message may occur in critical regions of the simulation where some variables may be changing rapidly. The step size is too large and will be adjusted automatically by the executive.

(2) THE CORRECTOR FAILED TO CONVERGE IN 3 ITERATIONS
DIFFERENTIAL EQUATIONS ___ TO ___

This message will be printed when the nonstiff option is used for a stiff system, or if the stiff option is used with direct iteration of the corrector for a stiff system. The stiff option should be used with Newton-Raphson iteration in either case. It may also occur if the problem is formulated incorrectly; i.e., there is an error in the differential equations or in the Jacobian, or the Jacobian is not accurate enough. The differential equations causing the problem are given. They are numbered in order of their occurence in the simulation.

(3) THE CORRECTOR CANNOT BE SOLVED BECAUSE PW(__) IS A PIVOT ELEMENT WHOSE VALUE IS BELOW XMIN DIFFERENTIAL EQUATIONS __ TO __ OPERATOR LIST WILL BE RE-CREATED

This message can appear in using the stiff option with Newton-Raphson iteration, i.e., the Bending-Hutchison linear equation solver. If it occurs more than once or

or twice, it may help to decrease the variable XMIN through the data set. The formulation of the differential equations and Jacobian should be checked.

(4) THE MAXIMUM ORDER SPECIFIED IS OUT OF RANGE

For the nonstiff option, the maximum order must be in the range 1-7. The range is 1-6 for the stiff option. The default value is 6 for both options, but the user may specify a value through the data set.

(5) CORRECTOR CONVERGENCE COULD NOT BE ACHIEVED FOR H.GT.HMIN DIFFERENTIAL EQUATIONS __ TO __

Message (2) will precede this one and the cause is the same, but this time the program will stop.

(6) THE STEP WAS TAKEN WITH H=HMIN BUT THE REQUESTED ERROR WAS NOT ACHIEVED

The user should try a larger value for the error tolerance or a smaller value of HMIN.

(7) THE REQUESTED ERROR IS SMALLER THAN CAN BE HANDLED FOR THIS PROBLEM

EPS has been specified equal to zero. As in (6), the user should try a larger tolerance.

(8) ERROR IN TRGB

This means that the Jacobian matrix has not been properly defined; i.e., there is a redundant or inconsistent equation. It may also occur if the arrays are dimensioned too small.

8. SIMULATION EXAMPLES

Four relatively simple examples were simulated with the new DYNSYS 2.0 executive:

(1) the level control of a simple stirred tank (nonstiff) with time delay

(2) a network of 15 stirred tank reactors, 2 o.d.e.s per reactor, some stiff and some nonstiff

(3) a tubular reactor with 222 stiff equations resulting from the discretization of partial differential equations

(4) a tubular reactor with 49 stiff o.d.e.s with tri-diagonal Jacobian matrix.

The examples are not intended to be rigorously realistic, but they do demonstrate the types of problems which may be handled. A more complex simulation is described in Chapter 9. No system of units is employed since the examples are fictitious. The reader may assume any set of consistent units. Each example contains a listing of the modules and data set. A description of a data set is in Appendix F.

8.1 Level Control Example

This example is based on one in the DYNSYS Manual (Bobrow *et al.*, 1970). Figure 8.1 is the process flow diagram for a simple mixing tank equipped with a level controller.

Liquid flows into a stirred tank at a fluctuating rate. The tank outlet stream is equipped with a valve which is manipulated by a level controller connected to the tank.

The response of the system to a simultaneous step change in inlet flow and temperature is to be simulated. The controller will attempt to keep the tank level at a certain set point by manipulating the flow rate through the outlet valve. Available in the DYNSYS library of modules are mathematical models for:

(1) a perfectly mixed tank (STIR1)

(2) a valve with a parabolic characteristic (VALV1)

(3) a proportional-integral controller (CONT1)

(4) a time delay (DELAY).

Figure 8.2 is the dynamic information flow diagram for the system. There is a time delay between the tank and the controller and another between the controller and

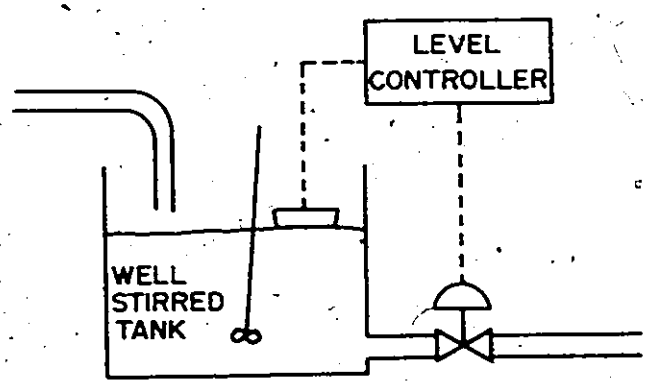


FIGURE 8.1 PROCESS FLOW DIAGRAM - LEVEL CONTROL EXAMPLE

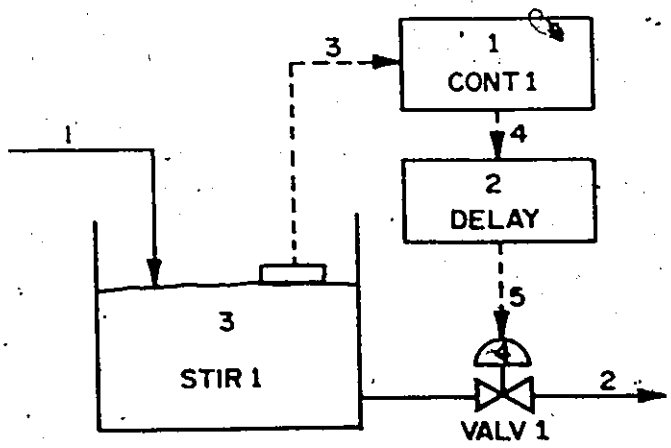


FIGURE 8.2 : DYNAMIC INFORMATION FLOW DIAGRAM - LEVEL CONTROL EXAMPLE

the valve but these can be represented together by one module.

STIR1 is the only differential module, i.e., the only one containing differential equations.

The unsteady state mass balance for each component i can be written:

$$\frac{dM_i}{dt} = F_{i,in} - F_{i,out} \qquad (8.1.1)$$

where $M_i \equiv$ mass of component i in the tank

$F_{i,in} \equiv$ inlet mass flow of component i

$F_{i,out} \equiv$ outlet mass flow of component i

in terms of the variables of DYNSYS

$$\frac{dM_i}{dt} = F_{in} * x_{i,in} - F_{out} * x_{i,out} \qquad (8.1.2)$$

where $x_i \equiv$ component i mass fraction

$F \equiv$ total mass flow.

This equation can be integrated to give the mass holdup of the component, M_i.

Summation over all components i of M_i

$$M = \sum_{i=1}^{NCOMP} M_i \quad (NCOMP \text{ - number of components}) \qquad (8.1.3)$$

gives the total mass holdup in the tank. The mass fractions of each component in the vessel, and hence in the outlet stream, may be obtained by division:

$$x_i = \frac{M_i}{M} \qquad (8.1.4)$$

similarly the heat balance for the vessel:

$$\frac{dH}{dt} = H_{in} - H_{out} \qquad (8.1.5)$$

becomes:

$$\frac{dH}{dt} = F_{in} * T_{in} * CP_{in} - F_{out} * T_{out} * CP_{out} \qquad (8.1.6)$$

and H is obtained by integration.

Assuming constant heat capacity, the new vessel temperature is:

$$T = \frac{H}{M * CP} \qquad (8.1.7)$$

The modules CONTL and VALVL are described in the DYNSYS Manual (Bobrow *et al.*, 1970).

The time delay module, DELAY, uses the "bucket brigade" approach to simulate a delayed variable. The past times and stream information are stored in vectors

and the stream output from the delay is interpolated from these values. The module is described more fully in Appendix G.

A listing of the modules (except DELAY) and the data set is shown in Figure 8.4. A listing of DELAY appears in Appendix G. A graph of the tank level versus time is depicted in Figure 8.3. After a little more than two time units, the tank level returns to its set point of 1000.

The old version of DYNSYS required 20.8 seconds of execution time to simulate this example, while DYNSYS 2.0 took 5.9 seconds. In both runs, HMAX, the maximum permissible step size was limited to 0.01. However, with DYNSYS 2.0, HMAX had to be limited to 0.01 to get accurate results, while the AMOS version could use a maximum step of 0.05 and still be accurate. Possibly the Gear algorithm does not handle forcing functions as well as the Adams-Moulton-Shell routine. The Gear algorithm might function better if the order was limited to one or two when forcing functions are present.

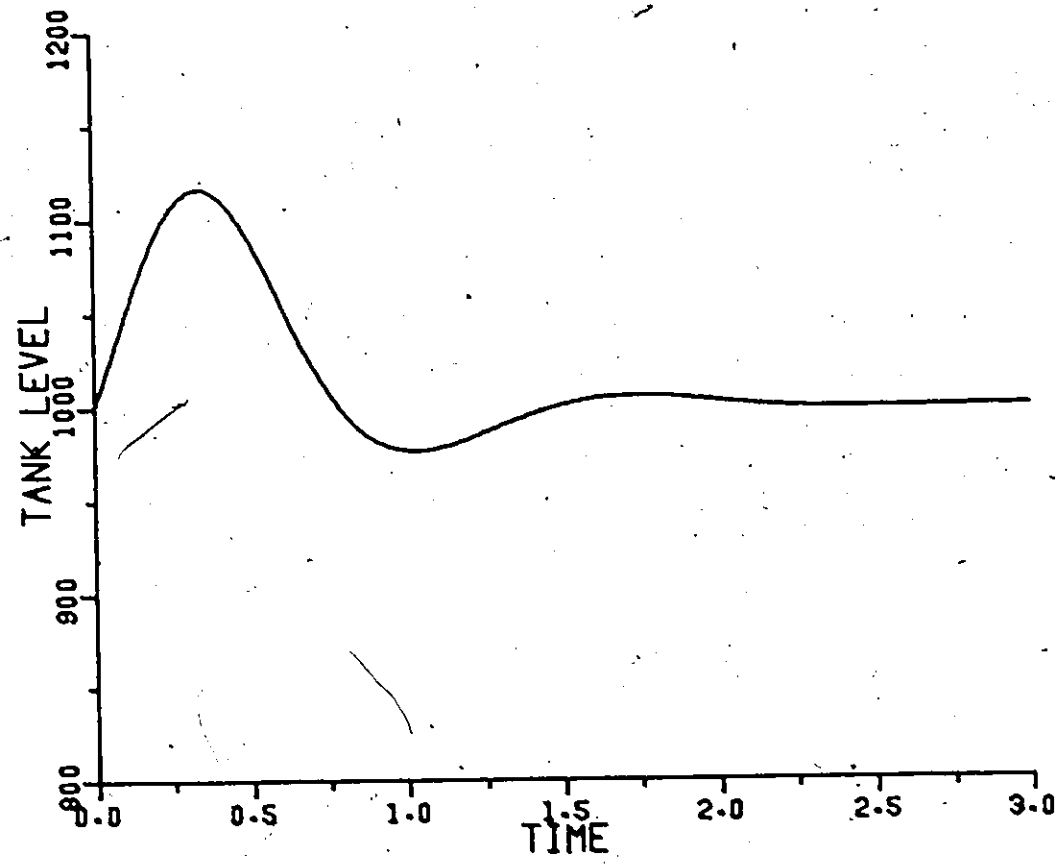


FIGURE 8.3: GRAPH OF TANK LEVEL VERSUS TIME FOR EXAMPLE#1

FIGURE 8.4: LISTING OF MODULES AND DATA SET FOR EXAMPLE #1
LEVEL CONTROL SYSTEM WITH TIME DELAY
LISTING OF TIME DELAY MODULE IS IN APPENDIX G

```
      SUBROUTINE TYPE1                                                          T01    1
C                                                                               T01    2
C     SUBROUTINE VALV1                                                          T01    3
C                                                                               T01    4
C     V-PORT (PARABOLIC) CONTROL VALVE                                          T01    5
C                                                                               T01    6
C     EQUIPMENT PARAMETERS                                                      T01    7
C                                                                               T01    8
C     1 - NOT USED                                                              T01    9
C     2 - VALVE CONSTANT                                                        T01   10
C     3 - ACTION (+=DIRECT,-=REVERSE)                                           T01   11
C                                                                               T01   12
      COMMON /UNIT/ IM,NMP                                                      T01   13
      COMMON /MAT/ MP(4,5),EP(4,5),S(2,5,7),EX(1)                               T01   14
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH             T01   15
                                                                                T01   16
C                                                                               T01   17
      MI=MP(IM,3)                                                               T01   18
      MO=IABS(MP(IM,4))                                                         T01   19
C     CHECK THAT THE VALVE SIGNAL IS IN THE RANGE 0-100                         T01   20
      V=S(1,MI,3)                                                               T01   21
      IF (V.LT.0.) V=0.                                                         T01   22
      IF (V.GT.100.) V=100.                                                     T01   23
C     ACTION                                                                    T01   24
      A=EP(IM,3)                                                                T01   25
      IF (A.LT.0.) S(1,MO,3)=EP(IM,2)*(100.-V)**2                               T01   26
      IF (A.GE.0.) S(1,MO,3)=EP(IM,2)*V**2                                      T01   27
      RETURN                                                                    T01   28
      END
```

```
      SUBROUTINE TYPE4                                                          T04    1
C                                                                               T04    2
C     SUBROUTINE CONT1                                                          T04    3
C                                                                               T04    4
C     PI OR ON/OFF CONTROLLER                                                   T04    5
C     SECOND OPTIONS ARE FOR ON/OFF CONTROLLER                                  T04    6
C     IF EQUIPMENT PARAMETERS 4 AND 5 ARE LT.0,CONTROLLER IS ON/OFF             T04    7
C                                                                               T04    8
C     EQUIPMENT PARAMETERS                                                      T04    9
C                                                                               T04   10
C     1 - CONTROLLED VARIABLE                                                   T04   11
C     2 - RANGE OR UPPER LIMIT                                                  T04   12
C     3 - SET POINT OR LOWER LIMIT                                              T04   13
C     4 - PROPORTIONAL GAIN OR LT. 0                                            T04   14
C     5 - INTEGRAL TIME OR LT. 0  ,IF.LT.-10.0,OFF WHEN .GT.SETP                T04   15
C                                                                               T04   16
      COMMON /UNIT/ IM,NMP                                                      T04   17
      COMMON /MAT/ MP(4,5),EP(4,5),S(2,5,7),EX(1)                               T04   18
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH             T04   19
C                                                                               T04   20
      P1=EP(IM,4)                                                               T04   21
      P2=EP(IM,5)                                                               T04   22
      IN=MP(IM,3)                                                               T04   23
      IOUT=IABS(MP(IM,4))                                                       T04   24
      K=EP(IM,1)                                                                T04   25
      IF (P1.LT.0.0.AND.P2.LE.0.0) GO TO 1                                      T04   26
C     MEASURED VARIABLE AT CURRENT TIME                                         T04   27
      SIG1=S(1,IN,K)                                                            T04   28
C     MEASURED VARIABLE AT LAST TIME                                            T04   29
      SIG2=S(2,IN,K)                                                            T04   30
C     SCALE THE ERROR                                                           T04   31
      SCALE=50.0/EP(IM,2)                                                       T04   32
C     PRESENT ERROR                                                             T04   33
      ERR=(SIG1-EP(IM,3))*SCALE                                                 T04   34
C     LAST ERROR                                                                T04   35
      OLDER=(SIG2-EP(IM,3))*SCALE                                               T04   36
C     OUTPUT SIGNAL                                                             T04   37
      S(1,IOUT,3)=P1*(ERR-OLDER+P2*(ERR+OLDER)*0.5*H)+S(2,IOUT,3)               T04   38
      IF (S(1,IOUT,3).GT.100.0) S(1,IOUT,3)=100.0                               T04   39
      IF (S(1,IOUT,3).LT.0.0) S(1,IOUT,3)=0.0                                   T04   40
      RETURN                                                                    T04   41
1     CONTINUE                                                                  T04   42
C     ON/OFF CONTROLLER                                                         T04   43
      IF (IG.EQ.1) RETURN                                                       T04   44
      T=S(1,IN,K)                                                               T04   45
      IF (P2.LT.-10.0) GO TO 2                                                  T04   46
      IF (T.GE.EP(IM,2)) T=50.0                                                 T04   47
      IF (T.LT.EP(IM,3)) T=0.0                                                  T04   48
      GO TO 3                                                                   T04   49
2     CONTINUE                                                                  T04   50
      IF (T.LE.EP(IM,2)) T=50.0                                                 T04   51
      IF (T.GT.EP(IM,2)) T=0.0                                                  T04   52
```

```
3     CONTINUE
      S(1,IOUT,3)=T
      RETURN
      END
```

```
T04  53
T04  54
T04  55
T04  56
```

```
      SUBROUTINE TYPE3                                                  T03   1
C                                                                       T03   2
C     STIR01  -  TYPE 3                                                 T03   3
C     ****************************************************************T03   4
C     GENERAL WELL MIXED, MULTIPLE INPUT/OUTPUT VESSEL                  T03   5
C     PRESSURE EFFECTS ARE CONSIDERED( ASSUMING IDEAL GAS               T03   6
C     BEHAVIOUR ) IF A VOLUME IS SPECIFIED                              T03   7
C     PARAMETERS.....                                                   T03   8
C     -1-    INITIAL HOLDUP                                             T03   9
C     -2-    VOLUME                                                     T03  10
C     -5-    1ST. OUTPUT STREAM, USED INTERNALLY                        T03  11
C     3 AND 4 ARE NOT SPECIFIED BY USER.                                T03  12
C     ****************************************************************T03  13
      COMMON /UNIT/ IM,NMP                                              T03  14
      COMMON /MAT/ MP(4,5),EP(4,5),S(2,5,7),EX(1)                       T03  15
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH     T03  16
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NQVMAX,ICONV,NOPPT,ISTIFF    T03  17
      COMMON /PTAB/ IGFLAG,PP(10,10)                                    T03  18
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO                             T03  19
      DIMENSION Y(10), CMPT(10)                                         T03  20
      REAL MW                                                           T03  21
C                                                                       T03  22
C     USE DIRECT ITERATION IF STIFF OPTION IS USED (ITER=0)             T03  23
C                                                                       T03  24
      ITER=0                                                            T03  25
      NC1=NCOMP+1                                                       T03  26
      VOL=EP(IM,2)                                                      T03  27
      IF (JSTART.NE.0.OR.IG.EQ.1) GO TO 4                               T03  28
C     FIND THE FIRST OUTPUT STREAM                                      T03  29
      DO 1 J=3,NMP                                                      T03  30
      IS=IABS(MP(IM,J))                                                 T03  31
      IF (ABS(S(1,IS,2)).GT.10.) GO TO 1                                T03  32
      IF (MP(IM,J).LT.0) GO TO 2                                        T03  33
1     CONTINUE                                                          T03  34
2     EP(IM,5)=J                                                        T03  35
      NOUT=EP(IM,5)                                                     T03  36
      IOUT=-MP(IM,NOUT)                                                 T03  37
C     CALCULATE INITIAL HOLDUP IF NOT SPECIFIED.                        T03  38
      IF (VOL.GT.0.0.AND.EP(IM,1).LE.0.0) EP(IM,1)=VOL*SG(1,IS)         T03  39
      IF (ABS(S(1,IS,2)).GT.10.) WRITE (6,15) MP(IM,1)                  T03  40
C     CALCULATE INITIAL CONDITIONS FOR VARIABLES TO BE INTEGRATED,      T03  41
C     I.E.,HEAT AND MASSES.                                             T03  42
C     HEAT                                                              T03  43
      Y(1)=EP(IM,1)*S(IG,IOUT,4)*CP(IG,IOUT)                            T03  44
C     MASSES                                                            T03  45
      DO 3 K=6,NC5                                                      T03  46
      KK=K-4                                                            T03  47
      Y(KK)=EP(IM,1)*S(IG,IOUT,K)                                       T03  48
3     CONTINUE                                                          T03  49
4     CONTINUE                                                          T03  50
      NOUT=EP(IM,5)                                                     T03  51
      IOUT=-MP(IM,NOUT)                                                 T03  52
```

```
C     CALCULATE DERIVATIVES                                                    T03  53
5     CONTINUE                                                                 T03  54
      DO 6 J=1,NC1                                                             T03  55
      CMPT(J)=0                                                                T03  56
6     CONTINUE                                                                 T03  57
      DO 9 J=3,NMP                                                             T03  58
      IS=IABS(MP(IM,J))                                                        T03  59
      IF (IS.EQ.0) GO TO 9                                                     T03  60
C     DO NOT INCLUDE CONTROLLER SIGNALS IN MASS AND HEAT BALANCES              T03  61
      IF (ABS(S(1,IS,2)).GT.10.) GO TO 9                                       T03  62
      SIGN=IS/MP(IM,J)                                                         T03  63
C     ALLOW HEAT INPUT STREAM WITH FLAG = 9                                    T03  64
C     S( , ,3) CONTAINS HEAT RATE IN BTU./MIN.                                 T03  65
      IF (ABS(S(1,IS,2)).LE.8.999) GO TO 7                                     T03  66
      CMPT(1)=CMPT(1)+S(1,IS,3)                                                T03  67
      GO TO 9                                                                  T03  68
7     CONTINUE                                                                 T03  69
C     HEAT                                                                     T03  70
      CMPT(1)=CMPT(1)+SIGN*S(IG,IS,4)*CP(IG,IS)*S(IG,IS,3)                     T03  71
C     MASS BALANCES                                                            T03  72
      DO 8 K=2,NC1                                                             T03  73
      KK=K+4                                                                   T03  74
      CMPT(K)=CMPT(K)+SIGN*S(IG,IS,KK)*S(IG,IS,3)                              T03  75
8     CONTINUE                                                                 T03  76
9     CONTINUE                                                                 T03  77
C     PREDICT/CORRECT NEW VALUES                                               T03  78
      CALL DIFSUB (NC1,Y,CMPT)                                                 T03  79
C     NORMALIZE MASS FRACTIONS                                                 T03  80
      SUM=0.0                                                                  T03  81
      DO 10 I=2,NC1                                                            T03  82
      SUM=SUM+Y(I)                                                             T03  83
10    CONTINUE                                                                 T03  84
      DO 11 K=6,NC5                                                            T03  85
      S(1,IOUT,K)=Y(K-4)/SUM                                                   T03  86
11    CONTINUE                                                                 T03  87
C     THE NEW TEMPERATURE                                                      T03  88
      S(1,IOUT,4)=Y(1)/(SUM*CP(1,IOUT))                                        T03  89
      IF (IDERY.NE.0) GO TO 5                                                  T03  90
C     INSERT THE NEW VALUES IN THE OUTPUT STREAMS                              T03  91
      DO 13 J=3,NMP                                                            T03  92
      IS=-MP(IM,J)                                                             T03  93
      IF (IS.LE.0) GO TO 13                                                    T03  94
      DO 12 K=4,NC5                                                            T03  95
      S(1,IS,K)=S(1,IOUT,K)                                                    T03  96
12    CONTINUE                                                                 T03  97
C     FOR CONTROLLER OUTPUT,HOLD-UP STORED IN PLACE OF FLOW                    T03  98
      IF (ABS(S(1,IS,2)).GT.10.) S(1,IS,3)=SUM                                 T03  99
13    CONTINUE                                                                 T03 100
      IF (VOL.LE.0.0) RETURN                                                   T03 101
C     PRESSURE                                                                 T03 102
      PRES=SUM*10.71*S(1,IOUT,4)/(VOL*MW(1,IOUT))                              T03 103
      DO 14 I=3,NMP                                                            T03 104
```

```
      J=IABS(MP(IM,I))                                                  T03 105
      IF (J.EQ.0) GO TO 14                                              T03 106
      S(1,J,5)=PRES                                                     T03 107
14    CONTINUE                                                          T03 108
      RETURN                                                            T03 109
C                                                                       T03 110
15    FORMAT (1H0,14H ERROR EQ. NO.,I3,22H ONLY OUTPUT IS SIGNAL)       T03 111
      END                                                               T03 112
```

```
**********************************************************************
      TEST OF SIMPLE LEVEL CONTROL SYSTEM, WITH STEP
      CHANGE IN INLET FLOW RATE AND TEMPERATURE,
      TIME DELAY INCLUDED.
**********************************************************************

BEGIN
IN/OUT       3.0
TOLERANCE    0.001
HMAX         0.01
TIME         3.0
LIBRARY       1.0
DELAY         9.0
PROCESS
CONT1        1.0
             3.0          -4.0
             3.           1500.          1000.          2.          3.
DELAY         2.0
             4.0          -5.0
             0.1          0.0            100.0          3.0         0.0
STIR1        3.0
             1.0          -2.0           -3.0
             1000.
VALV1        4.0
             5.0          -2.0
                          1.0            1.0
END
STREAMS      5.
EXPLICIT
             1.           1.             1000.          100.        14.7
             1.
             2.           1.             500.           50.         14.7
             1.
             3.           11.            1000.

             4.           11.            22.36068

             5.           11.            22.36068

END
PROPERTIES   -1.
END
      END
```

8.2 Stirred Tank Reactor Network

Figures 8.5 and 8.6 represent the process flow diagram and the dynamic information flow diagram respectively for a network of CSTRs. The first order reversible reaction:

$$A \underset{k_2}{\overset{k_1}{\rightleftharpoons}} B$$

occurs in each reactor.

The differential equations for the module are the mass balances for components A and B:

$$\frac{dy_1}{dt} = -(k_1 + \frac{1}{\tau})y_1 + k_2 y_2 + y_{1,0}/\tau$$

$$\frac{dy_2}{dt} = k_1 y_1 - (k_2 + \frac{1}{\tau})y_2 + y_{2,0}/\tau \qquad (8.2.1)$$

where y_1, y_2 ≡ mass fraction components A and B respectively

τ ≡ reactor time constant (volume/volumetric flow rate)

k_1, k_2 ≡ reaction rate constants

$y_{1,0}, y_{2,0}$ ≡ mass fraction of components A and B respectively in inlet stream

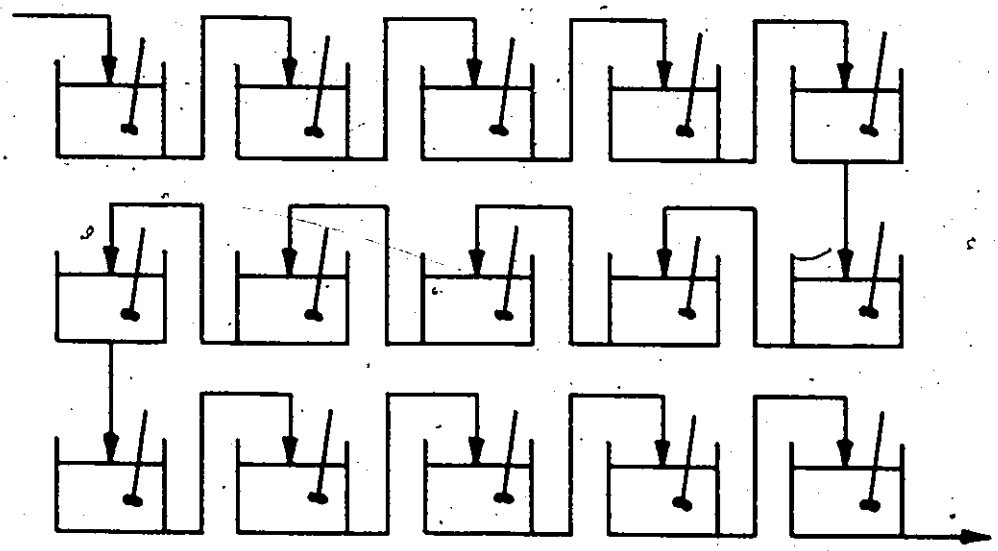


FIGURE 8.5 : PROCESS FLOW DIAGRAM – REACTOR NETWORK EXAMPLE

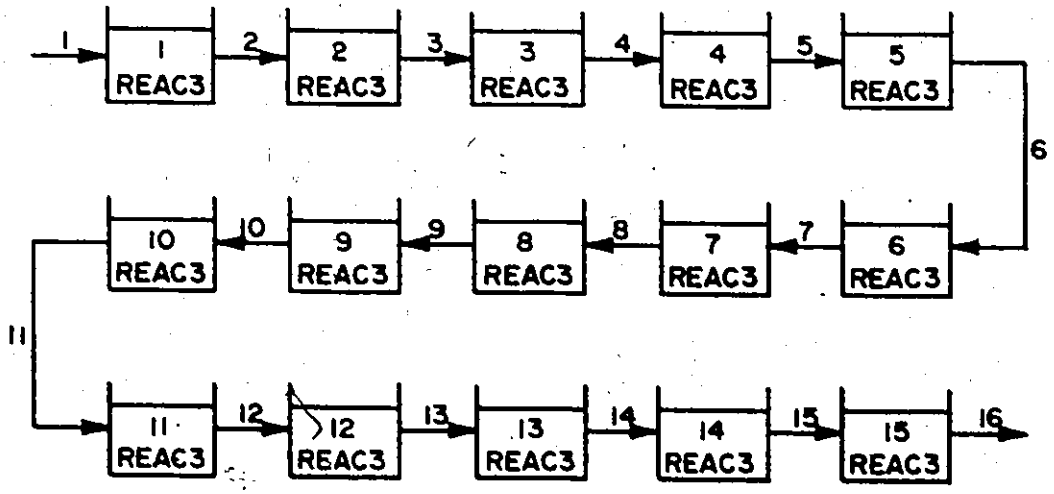


FIGURE 8.6 : DYNAMIC INFORMATION FLOW DIAGRAM – REACTOR NETWORK EXAMPLE

The Jacobian matrix of the system is :

$$\underset{\approx}{J} = \begin{bmatrix} -(k_1 + \frac{1}{\tau}) & + k_2 \\ k_1 & -(k_2 + \frac{1}{\tau}) \end{bmatrix} \qquad (8.2.2)$$

The eigenvalues are $-\frac{1}{\tau}$ and $-k_1 - k_2 - \frac{1}{\tau}$.

By choosing values of k_1, k_2 and τ we may create different stiffness ratios in each of the reactors. In practice the k's may vary because of different temperatures in each reactor. Table 8.1 shows the values which were used for the simulation. This creates a mixture of non-stiff and stiff modules. It was found experimentally that if the stiffness ratio is greater than approximately 3.5:1, Newton-Raphson iteration of the corrector should be used, but that below 3.5:1 the execution time is faster for direct iteration.

DYNSYS 2.0 required 3.92 seconds of execution time. A listing of the module and its data set are given in Figure 8.7. A graph of the outlet concentration of A versus time for each reactor is shown in Figure 8.8. The curves are ordered from top to bottom in order of increasing module number. As the module number increases, so does the stiffness ratio as seen in Table 8.1. It can be seen in Figure 8.8 that the higher the stiffness ratio, the faster the reactor reaches steady state.

Table 8.1: Simulation Parameters For Reactor Network Example

Module	Tau	K1/K2	K1	K2	Eig1	Eig2	Stiffness Ratio
1	1.0	1.1	.26190	.23810	-1.0	-1.5	1.5
2	1.0	1.2	.54545	.45455	-1.0	-2.0	2.0
3	1.0	1.3	.84783	.65217	-1.0	-2.5	2.5
4	1.0	1.4	1.16667	.83333	-1.0	-3.0	3.0
5	1.0	1.5	5.40000	3.60000	-1.0	-10.0	10.0
6	1.0	1.6	30.15385	18.84615	-1.0	-50.0	50.0
7	1.0	1.7	62.33333	36.66667	-1.0	-100.0	100.0
8	1.0	1.8	127.92857	71.07143	-1.0	-200.0	200.0
9	1.0	1.9	261.41379	137.58621	-1.0	-400.0	400.0
10	1.0	2.0	399.33333	199.66667	-1.0	-600.0	600.0
11	1.0	2.1	541.25806	257.74194	-1.0	-800.0	800.0
12	1.0	2.2	686.81250	312.18750	-1.0	-1000.0	1000.0
13	1.0	2.3	835.66667	363.33333	-1.0	-1200.0	1200.0
14	1.0	2.4	1058.11765	440.88235	-1.0	-1500.0	1500.0
15	1.0	2.5	1427.85714	571.14286	-1.0	-2000.0	2000.0

FIGURE 8.7: LISTING OF MODULE AND DATA SET FOR EXAMPLE #2
CSTR WITH FIRST ORDER REVERSIBLE REACTION (15 IN SERIES)

```
      SUBROUTINE TYPE10                                                  T10    1
                                                                         T10    2
C                                                                        T10    3
C     SUBROUTINE REAC1                                                   T10    4
C                                                                        T10    5
C     THIS MODULE REPRESENTS A CSTR WITH A FIRST ORDER REVERSIBLE        T10    6
C     REACTION, 1 INPUT STREAM, 1 OUTPUT STREAM, CONSTANT TEMPERATURE.   T10    7
C                                                                        T10    8
C     EQUIPMENT PARAMETERS                                               T10    9
C                                                                        T10   10
C     1 - VOL  - REACTOR VOLUME - FT**3                                  T10   11
C     2 - K1   - FORWARD REACTION RATE CONSTANT - MIN**-1                T10   12
C     3 - K2   - BACKWARD REACTION RATE CONSTANT - MIN**-1               T10   13
C     4 - ITER - ITER=0 - USE DIRECT ITERATION WITH STIFF OPTION         T10   14
C                ITER=1 - USE NEWTON-RAPHSON ITERATION WITH STIFF OPTION T10   15
C                FOR NON-STIFF OPTION ITER IS NOT USED                   T10   16
C                                                                        T10   17
      COMMON /MAT/ MP(15,5),EP(15,5),S(2,16,7),EX(1)                     T10   18
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      T10   19
      COMMON /PTAB/ IGFLAG,PP(10,10)                                     T10   20
      COMMON /UNIT/ IM                                                   T10   21
      COMMON /ROW/ IROW(4)                                               T10   22
      COMMON /COLUMN/ JCOL(4)                                            T10   23
      COMMON /JACOB/ XJACOB(4)                                           T10   24
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO                              T10   25
      DIMENSION Y(2), DERY(2)                                            T10   26
      REAL K1,K2                                                         T10   27
C                                                                        T10   28
C     CALCULATE MODULE PARAMETERS.                                       T10   29
C                                                                        T10   30
C     GET VOLUME OF REACTOR FROM EP - VOL                                T10   31
      VOL=EP(IM,1)                                                       T10   32
C     GET REACTION RATE CONSTANTS FROM EP - K1,K2                        T10   33
      K1=EP(IM,2)                                                        T10   34
      K2=EP(IM,3)                                                        T10   35
C     GET ITERATION OPTION FROM EP                                       T10   36
      ITER=EP(IM,4)                                                      T10   37
C     GET STREAM NUMBER OF INPUT STREAM FROM MP - IN                     T10   38
      IN=MP(IM,3)                                                        T10   39
C     GET STREAM NUMBER OF OUTPUT STREAM FROM MP - IOUT                  T10   40
      IOUT=IABS(MP(IM,4))                                                T10   41
C     GET DENSITY OF INPUT STREAM FROM PP - DENS                         T10   42
      DENS=PP(1,2)                                                       T10   43
C     CALCULATE REACTOR TIME CONSTANT - TAU                              T10   44
      TAU=VOL/(S(IG,IN,3)/DENS)                                          T10   45
C     GET INITIAL REACTOR CONCENTRATIONS FROM OUTPUT STREAM              T10   46
      Y(1)=S(IG,IOUT,6)                                                  T10   47
      Y(2)=S(IG,IOUT,7)                                                  T10   48
C
```

```
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS                        T10   49
C                                                                        T10   50
      IF (IG.EQ.2) GO TO 1                                               T10   51
C     NZERO IS NUMBER OF NON-ZERO ELEMENTS IN JACOBIAN                   T10   52
      NZERO=4                                                            T10   53
      IROW(1)=1                                                          T10   54
      JCOL(1)=1                                                          T10   55
      XJACOB(1)=-(K1+1.0/TAU)                                            T10   56
      IROW(2)=2                                                          T10   57
      JCOL(2)=1                                                          T10   58
      XJACOB(2)=K1                                                       T10   59
      IROW(3)=1                                                          T10   60
      JCOL(3)=2                                                          T10   61
      XJACOB(3)=K2                                                       T10   62
      IROW(4)=2                                                          T10   63
      JCOL(4)=2                                                          T10   64
      XJACOB(4)=-(K2+1.0/TAU)                                            T10   65
1     CONTINUE                                                           T10   66
C                                                                        T10   67
C     CALCULATE DERIVATIVES                                              T10   68
C                                                                        T10   69
C     MASS BALANCE FOR COMPONENT A (UNITS=MIN**-1)                       T10   70
C     RATE OF CHANGE OF MASS FRACTION OF COMPONENT A                     T10   71
C      = RATE OF INPUT OF A                                              T10   72
C       -RATE OF OUTPUT OF A                                             T10   73
C        +RATE AT WHICH B REACTS INTO A                                  T10   74
C        -RATE AT WHICH A REACTS INTO B                                  T10   75
      DERY(1)=-(K1+1.0/TAU)*Y(1)+K2*Y(2)+S(IG,IN,6)/TAU                  T10   76
C     MASS BALANCE FOR COMPONENT B (UNITS=MIN**-1)                       T10   77
C     RATE OF CHANGE OF MASS FRACTION OF COMPONENT B                     T10   78
C      = RATE OF INPUT OF B                                              T10   79
C        -RATE OF OUTPUT OF B                                            T10   80
C        +RATE AT WHICH A REACTS INTO B                                  T10   81
C        -RATE AT WHICH B REACTS INTO A                                  T10   82
      DERY(2)=K1*Y(1)-(K2+1.0/TAU)*Y(2)+S(IG,IN,7)/TAU                   T10   83
C                                                                        T10   84
C     CALL DIFSUB TO SOLVE ODES FOR MODULE                               T10   85
C                                                                        T10   86
      CALL DIFSUB (2,Y,DERY)                                             T10   87
      IF (IDERY.NE.0) GO TO 1                                            T10   88
C                                                                        T10   89
C     CALCULATE STREAM OUTPUT                                            T10   90
C                                                                        T10   91
C     NORMALIZE CONCENTRATIONS OF A AND B                                T10   92
      SUM=Y(1)+Y(2)                                                      T10   93
      Y(1)=Y(1)/SUM                                                      T10   94
      Y(2)=Y(2)/SUM                                                      T10   95
C     PUT MASS FRACTIONS INTO OUTPUT STREAMS                             T10   96
      S(1,IOUT,6)=Y(1)                                                   T10   97
      S(1,IOUT,7)=Y(2)                                                   T10   98
      RETURN                                                             T10   99
      END                                                                T10  100
```

```
**********************************************************************
        SIMULATION OF 15 CSTRS IN SERIES
**********************************************************************
BEGIN
HMAX        1.0
TIME        1.0
COMPS       2.0
LIBRARY     1.0
REAC1       10.0
PROCESS
REAC1       1.0
            1.0         -2.0
            10.0        0.26190     0.23810     0.0
REAC1       2.0
            2.0         -3.0
            10.0        0.54545     0.45455     0.0
REAC1       3.0
            3.0         -4.0
            10.0        0.84783     0.65217     0.0
REAC1       4.0
            4.0         -5.0
            10.0        1.16667     0.83333     0.0
REAC1       5.0
            5.0         -6.0
            10.0        5.4         3.6         0.0
REAC1       6.0
            6.0         -7.0
            10.0        30.154      18.846      1.0
REAC1       7.0
            7.0         -8.0
            10.0        62.333      36.667      1.0
REAC1       8.0
            8.0         -9.0
            10.0        127.929     71.071      1.0
REAC1       9.0
            9.0         -10.0
            10.0        261.41      137.59      1.0
REAC1       10.0
            10.0        -11.0
            10.0        399.33      199.67      1.0
REAC1       11.0
            11.0        -12.0
            10.0        541.26      257.74      1.0
REAC1       12.0
            12.0        -13.0
            10.0        686.81      312.19      1.0
REAC1       13.0
            13.0        -14.0
            10.0        835.67      363.33      1.0
REAC1       14.0
            14.0        -15.0
```

```
                10.0        1058.12     440.88      1.0
REAC1           15.0
                15.0        -16.0
                10.0        1427.86     571.14      1.0
END
STREAMS         16.0
EXPLICIT
                1.0         1.0         624.0       60.0        14.7
                1.0         0.0
                2.0         1.0         624.0       65.0        14.7
                1.0         0.0
                3.0         1.0         624.0       70.0        14.7
                1.0         0.0
                4.0         1.0         624.0       75.0        14.7
                1.0         0.0
                5.0         1.0         624.0       80.0        14.7
                1.0         0.0
                6.0         1.0         624.0       85.0        14.7
                1.0         0.0
                7.0         1.0         624.0       90.0        14.7
                1.0         0.0
                8.0         1.0         624.0       95.0        14.7
                1.0         0.0
                9.0         1.0         624.0       100.0       14.7
                1.0         0.0
                10.0        1.0         624.0       105.0       14.7
                1.0         0.0
                11.0        1.0         624.0       110.0       14.7
                1.0         0.0
                12.0        1.0         624.0       115.0       14.7
                1.0         0.0
                13.0        1.0         624.0       120.0       14.7
                1.0         0.0
                14.0        1.0         624.0       125.0       14.7
                1.0         0.0
                15.0        1.0         624.0       130.0       14.7
                1.0         0.0
                16.0        1.0         624.0       135.0       14.7
                1.0         0.0
END
PROPERTIES      -1.0
END
      END
```

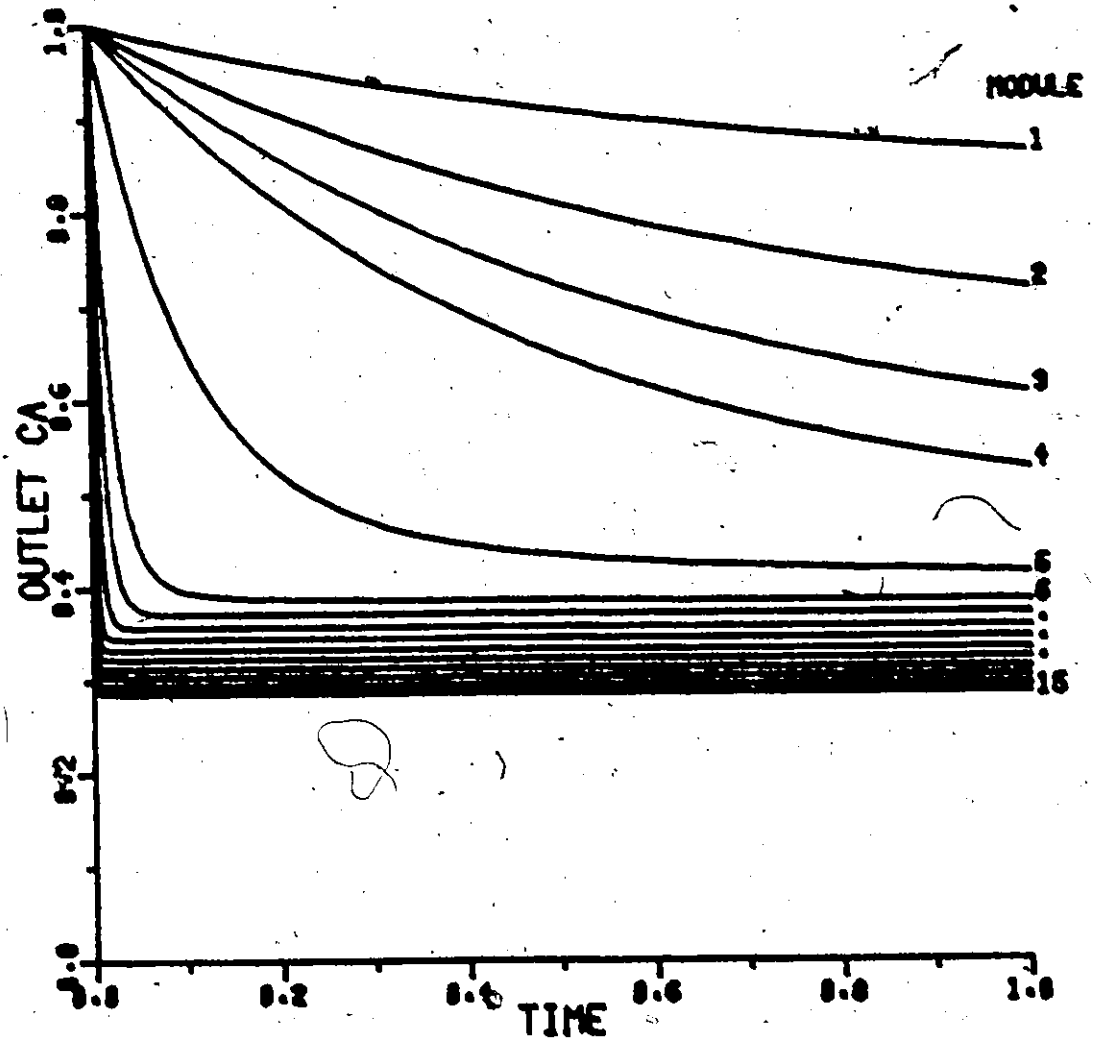


FIGURE 8.8: GRAPH OF OUTLET CONCENTRATION OF A VERSUS TIME FOR EXAMPLE#2

8.3 Tubular Reactor Simulation

An exothermal tubular chemical reactor based on Chapter 7 of the IMP Manual (Brandon, 1972) was simulated. A diagram of the reactor is shown in Figure 8.8. The following reaction occurs within the reactor:

$$2A \xrightarrow{k_1} B \xrightarrow{k_2} C \qquad (8.3.1)$$

The partial differential equations describing the process are:

$$\frac{\partial C_A}{\partial t} = -k_1 e^{\alpha_1 T} C_A^2 + D_1 \frac{\partial^2 C_A}{\partial Z^2} - V \frac{\partial C_A}{\partial Z}$$

$$\frac{\partial C_B}{\partial t} = -k_2 e^{\alpha_2 T} C_B + \frac{k_1}{2} e^{\alpha_1 T} C_A^2 + D_2 \frac{\partial^2 C_B}{\partial Z^2} - V \frac{\partial C_B}{\partial Z} \qquad (8.3.2)$$

$$\frac{\partial T}{\partial t} = k_1 e^{\alpha_1 T} C_A^2 H_1 + k_2 e^{\alpha_2 T} C_B H_2 + D_3 \frac{\partial^2 T}{\partial Z^2} - V \frac{\partial T}{\partial Z}$$

Reaction, diffusion and convection terms are included in the mathematical model. The following boundary and initial conditions apply:

$$C_A = 0 \text{ at } t = 0 \text{ for all } Z$$
$$C_B = 0 \text{ at } t = 0 \text{ for all } Z$$
$$T = 0 \text{ at } t = 0 \text{ for all } Z$$

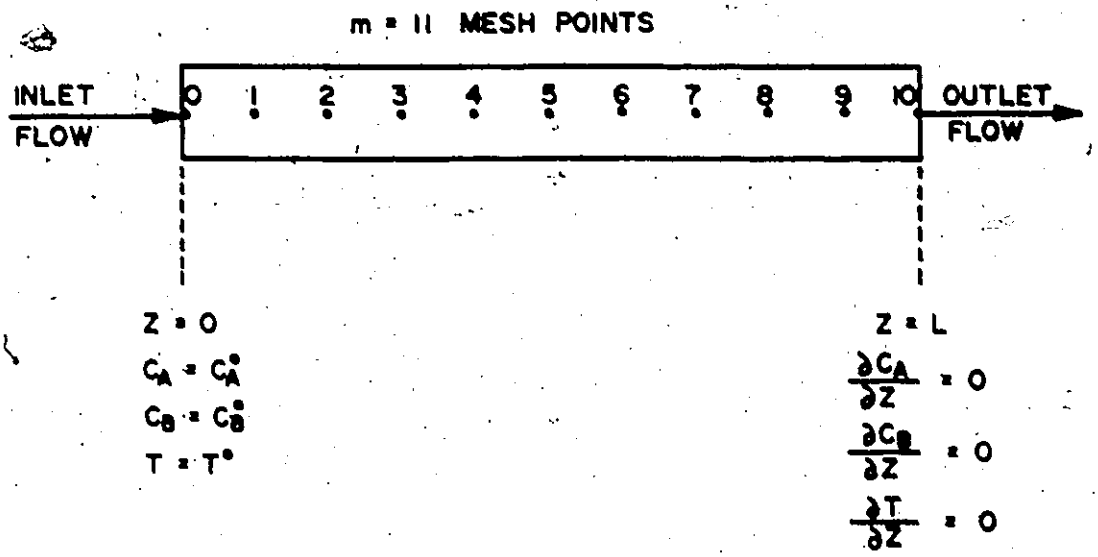


FIGURE 8.9: SCHEMATIC OF TUBULAR REACTOR

$$
\begin{aligned}
C_A &= C_A^{\,o} \text{ at } Z = 0 \text{ for all } t \\
C_B &= C_B^{\,o} \text{ at } Z = 0 \text{ for all } t \\
T &= T^o \text{ at } Z = 0 \text{ for all } t \\
\frac{\partial C_A}{\partial Z} &= 0 \text{ at } Z = L \text{ for all } t \\
\frac{\partial C_B}{\partial Z} &= 0 \text{ at } Z = L \text{ for all } t \\
\frac{\partial T}{\partial Z} &= 0 \text{ at } Z = L \text{ for all } t
\end{aligned}
\qquad (8.3.3)
$$

where

C_A ≡ concentration of component A

C_B ≡ concentration of component B

T ≡ temperature

t ≡ time

Z ≡ space coordinate along reactor axis

$C_A^{\,o}$ ≡ feed concentration of component A

$C_B^{\,o}$ ≡ feed concentration of component B

T^o ≡ inlet temperature

L ≡ reactor length

V ≡ fluid velocity

D_1 ≡ diffusivity of component A

D_2 ≡ diffusivity of component B

D_3 ≡ thermal diffusivity

k_1 ≡ reaction one rate constant

k_2 ≡ reaction two rate constant

α_1 ≡ reaction one activation constant

α_2 ≡ reaction two activation constant

H_1 ≡ heat of reaction one

H_2 ≡ heat of reaction two

The values of the parameters for this simulation are:

$$\begin{array}{lll} D_1 = 30.0 & H_1 = 1.0 & \alpha_2 = 0.07 \\ D_2 = 20.0 & H_2 = 50.0 & C_A^{\,o} = 10.0 \\ D_3 = 90.0 & k_1 = 1.5 & C_B^{\,o} = 0.0 \\ V = 100.0 & k_2 = 0.00002 & T^o = 100.0 \\ L = 100.0 & \alpha_1 = 0.01 & t_{max} = 5.0 \end{array} \qquad (8.3.4)$$

To use DYNSYS the set of partial differential equations must be broken down into a set of ordinary differential equations. To do this the length of the reactor is divided into M mesh points as shown in Figure 8.9. Each mesh point will have 3 o.d.e.s (except for the inlet and outlet):

Replacing $\frac{\partial X}{\partial Z}$ and $\frac{\partial^2 X}{\partial Z^2}$ by

$$\frac{\partial X}{\partial Z} = \frac{X|_Z - X|_{Z-\Delta Z}}{\Delta Z} \qquad (8.3.5)$$

$$\frac{\partial^2 X}{\partial Z^2} = \frac{X|_{Z+\Delta Z} - 2X|_Z + X|_{Z-\Delta Z}}{(\Delta Z)^2}$$

we obtain:

$$\frac{dC_{A,i}}{dt} = -k_1 e^{a_1 T_i} C_{A,i}^2 + D_1\left[\frac{C_{A,i+1} - 2C_{A,i} + C_{A,i-1}}{(\Delta Z)^2}\right] - V\left[\frac{C_{A,i+1} - C_{A,i}}{\Delta Z}\right]$$

$$\frac{dC_{B,i}}{dt} = -k_2 e^{a_2 T_i} C_{B,i} + \frac{k_1}{2} e^{a_1 T_i} C_{A,i}^2 + D_2\left[\frac{C_{B,i+1} - 2C_{B,i} + C_{B,i-1}}{(\Delta Z)^2}\right] - V\left[\frac{C_{2,i+1} - C_{2,i}}{\Delta Z}\right] \qquad (8.3.6)$$

$$\frac{dT_i}{dt} = k_1 e^{a_1 T_i} C_{A,i}^2 H_1 + k_2 e^{a_2 T} C_{B,i} H_2 + D_3\left[\frac{T_{i+1} - 2T_i + T_{i-1}}{(\Delta Z)^2}\right] - V\left[\frac{T_i - T_{i-1}}{\Delta Z}\right]$$

The equations are very stiff.

The equations at mesh point 1 are:

$$\frac{dy_1}{dt} = -k_1 e^{a_1 y_3} y_1^2 + D_1\left[\frac{y_4 - 2y_1 + y_{10}}{(\Delta Z)^2}\right] - V\left[\frac{y_4 - y_1}{\Delta Z}\right]$$

$$\frac{dy_2}{dt} = -k_2 e^{a_2 y_3} y_2 + \frac{k_1}{2} e^{a_1 y_3} y_1^2 + D_2\left[\frac{y_5 - 2y_2 + y_{20}}{(\Delta Z)^2}\right] - V\left[\frac{y_5 - y_2}{\Delta Z}\right] \qquad (8.3.7)$$

$$\frac{dy_3}{dt} = k_1 e^{a_1 y_3} y_1^2 H_1 + k_2 e^{a_2 y_3} y_2 H_2 + D_3\left[\frac{y_6 - 2y_3 + y_{30}}{(\Delta Z)^2}\right] - V\left[\frac{y_5 - y_2}{\Delta Z}\right]$$

The equations at mesh points 2, 3, ..., M-2 are:

$$\frac{dy_i}{dt} = -k_1 e^{a_1 y_{i+2}} y_i^2 + D_1\left[\frac{y_{i+3}-2y_i+y_{i-3}}{(\Delta Z)^2}\right] - V\left[\frac{y_{i+3}-y_i}{\Delta Z}\right]$$

$$\frac{dy_{i+1}}{dt} = -k_2 e^{a_2 y_{i+2}} y_{i+1} + \frac{k_1}{2} e^{a_1 y_{i+2}} y_i^2$$

$$+ D_2\left[\frac{y_{i+4}-2y_{i+2}+y_{i-2}}{(\Delta Z)^2}\right] - V\left[\frac{y_{i+4}-y_{i+1}}{\Delta Z}\right] \qquad (8.3.8)$$

$$\frac{dy_{i+2}}{dt} = -k_1 e^{a_1 y_{i+2}} y_i^2 H_1 + k_2 e^{a_2 y_{i+2}} y_{i+1} H_2$$

$$+ D_3\left[\frac{y_{i+5}-2y_{i+2}+y_{i-1}}{(\Delta Z)^2}\right] - V\left[\frac{y_{i+5}-y_{i+2}}{\Delta Z}\right]$$

The equations at the last mesh point are (i=3M-5):

$$\frac{dy_i}{dt} = -k_1 e^{a_1 y_{i+2}} y_i^2 + D_1\left[\frac{y_{i-3}-y_i}{(\Delta Z)^2}\right]$$

$$\frac{dy_{i+1}}{dt} = -k_2 e^{a_2 y_{i+2}} + \frac{k_1}{2} e^{a_1 y_{i+2}} y_i^2 + D_2\left[\frac{y_{i-2}-y_{i+1}}{(\Delta Z)^2}\right] \qquad (8.3.9)$$

$$\frac{dy_{i+2}}{dt} = k_1 e^{a_1 y_{i+2}} y_i^2 H_1 + k_2 e^{a_2 y_{i+2}} y_{i+1} H_2 + D_3\left[\frac{y_{i-1}-y_{i+2}}{(\Delta Z)^2}\right]$$

where $y_1, y_4, y_7, \ldots, y_{3M-5}$ refer to C_A

$y_2, y_5, y_8, \ldots, y_{3M-4}$ refer to C_B

$y_3, y_6, y_9, \ldots, y_{3M-3}$ refer to T

76 mesh points were taken giving a total of 222 o.d.e.s.

The execution time for the simulation was 102.5 seconds while an equation-oriented version of Gear's method took 114.2 seconds. The IMP package required 577.7 seconds.

A listing of the module and its data set are given in Figure 8.10.

Graphs of the outlet values of C_A, C_B and T versus time are depicted in Figures 8.11, 8.12 and 8.13 respectively. No change in the outlet values is seen until about 0.6 time units. Steady state is reached after about 1.5. C_A rises to a fairly sharp peak at about 1.0 and then falls to a steady state of about 0.226. The graphs of C_B and T are sigmoid in appearance.

FIGURE 8.10: LISTING OF MODULE AND DATA SET FOR EXAMPLE #3
TUBULAR REACTOR (222 STIFF ODES)

```
      SUBROUTINE TYPES5                                                  T05   1
C                                                                        T05   2
C     SUBROUTINE REAC3                                                   T05   3
C                                                                        T05   4
C     THIS MODULE REPRESENTS A TUBULAR REACTOR WITH                      T05   5
C     REACTION 2C1-C2-C3                                                 T05   6
C     FEED CONCENTRATIONS MUST BE FAIRLY CONSTANT OTHERWISE ODES         T05   7
C     MUST BE SOLVED FOR C3                                              T05   8
C                                                                        T05   9
C     EQUIPMENT PARAMETERS                                               T05  10
C                                                                        T05  11
C     1 - D1 - DIFFUSIVITY OF COMPONENT 1                                T05  12
C     2 - D2 - DIFFUSIVITY OF COMPONENT 2                                T05  13
C     3 - D3 - THERMAL DIFFUSIVITY                                       T05  14
C     4 - V  - FLUID VELOCITY                                            T05  15
C     5 - L  - LENGTH OF REACTOR                                         T05  16
C     6 - H1 - HEAT OF REACTION 1                                        T05  17
C     7 - H2 - HEAT OF REACTION 2                                        T05  18
C     8 - K1 - REACTION 1 RATE CONSTANT                                  T05  19
C     9 - K2 - REACTION 2 RATE CONSTANT                                  T05  20
C    10 - A1 - REACTION 1 ACTIVATION CONSTANT                            T05  21
C    11 - A2 - REACTION 2 ACTIVATION CONSTANT                            T05  22
C    12 - M  - NUMBER OF SECTIONS IN REACTOR                             T05  23
C                                                                        T05  24
      COMMON /MAT/ MP(1,5),EP(1,5),S(2,2,8),EX(7)                        T05  25
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      T05  26
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF     T05  27
      COMMON /UNIT/ IM,NMP                                               T05  28
      COMMON /ROW/ IROW(1030)                                            T05  29
      COMMON /COLUMN/ JCOL(1030)                                         T05  30
      COMMON /JACOB/ DFDY(1030)                                          T05  31
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO                              T05  32
      REAL K1,K2,K1K,K2K,L                                               T05  33
      DIMENSION Y(222), DERY(222)                                        T05  34
C                                                                        T05  35
C     CALCULATE MODULE PARAMETERS                                        T05  36
C                                                                        T05  37
C                                                                        T05  38
C     USE NEWTON-RAPHSON ITERATION WITH STIFF OPTION (ITER=1)            T05  39
C                                                                        T05  40
      ITER=1                                                             T05  41
      IF (JSTART.NE.0.OR.IG.EQ.1) GO TO 2                                T05  42
      D1=EP(IM,1)                                                        T05  43
      D2=EP(IM,2)                                                        T05  44
      D3=EP(IM,3)                                                        T05  45
      V=EP(IM,4)                                                         T05  46
      L=EP(IM,5)                                                         T05  47
      NEX=MP(IM,NMP+1)                                                   T05  48
```

```
      H1=EX(NEX)                                                        T05  49
      H2=EX(NEX+1)                                                      T05  50
      K1=EX(NEX+2)                                                      T05  51
      K2=EX(NEX+3)                                                      T05  52
      A1=EX(NEX+4)                                                      T05  53
      A2=EX(NEX+5)                                                      T05  54
      M=EX(NEX+6)                                                       T05  55
      DZ=L/FLOAT(M)                                                     T05  56
      DZ2=DZ*DZ                                                         T05  57
      D1DZ2=D1/DZ2                                                      T05  58
      D2DZ2=D2/DZ2                                                      T05  59
      D3DZ2=D3/DZ2                                                      T05  60
      VDZ=V/DZ                                                          T05  61
C                                                                       T05  62
C     CALCULATE INITIAL CONDITIONS                                      T05  63
C                                                                       T05  64
      N=3*M-3                                                           T05  65
      IN=MP(IM,3)                                                       T05  66
      IOUT=IABS(MP(IM,4))                                               T05  67
      IF=N-2                                                            T05  68
      DO 1 I=1,IF,3                                                     T05  69
      Y(I)=S(2,IOUT,3)*S(2,IOUT,6)                                      T05  70
      Y(I+1)=S(2,IOUT,3)*S(2,IOUT,7)                                    T05  71
      Y(I+2)=S(2,IOUT,4)                                                T05  72
1     CONTINUE                                                          T05  73
2     CONTINUE                                                          T05  74
C     NZERO IS NUMBER OF NONZERO ELEMENTS IN JACOBIAN                   T05  75
      NZERO=14*M-20                                                     T05  76
C                                                                       T05  77
C     CALCULATE FEED CONDITIONS                                         T05  78
C                                                                       T05  79
      C10=S(IG,IN,3)*S(IG,IN,6)                                         T05  80
      C20=S(IG,IN,3)*S(IG,IN,7)                                         T05  81
      T0=S(IG,IN,4)                                                     T05  82
C                                                                       T05  83
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS                       T05  84
C                                                                       T05  85
      IF (IG.EQ.2) GO TO 4                                              T05  86
C                                                                       T05  87
C     MESH POINT 1                                                      T05  88
C                                                                       T05  89
      K1K=K1*EXP(A1*Y(3))                                               T05  90
      K2K=K2*EXP(A2*Y(3))                                               T05  91
      IROW(1)=1                                                         T05  92
      JCOL(1)=1                                                         T05  93
      DFDY(1)=-2.0*K1K*Y(1)-2.0*D1DZ2-VDZ                               T05  94
      IROW(2)=1                                                         T05  95
      JCOL(2)=3                                                         T05  96
      DFDY(2)=-A1*K1K*Y(1)*Y(1)                                         T05  97
      IROW(3)=1                                                         T05  98
      JCOL(3)=4                                                         T05  99
      DFDY(3)=D1DZ2                                                     T05 100
```

```
      IROW(4)=2                                                          T05 101
      JCOL(4)=1                                                          T05 102
      DFDY(4)=K1K*Y(1)                                                   T05 103
      IROW(5)=2                                                          T05 104
      JCOL(5)=2                                                          T05 105
      DFDY(5)=-K2K-2.0*D2DZ2-VDZ                                         T05 106
      IROW(6)=2                                                          T05 107
      JCOL(6)=3                                                          T05 108
      DFDY(6)=-A2*K2K*Y(2)+0.5*A1*K1K*Y(1)*Y(1)                          T05 109
      IROW(7)=2                                                          T05 110
      JCOL(7)=5                                                          T05 111
      DFDY(7)=D2DZ2                                                      T05 112
      IROW(8)=3                                                          T05 113
      JCOL(8)=1                                                          T05 114
      DFDY(8)=2.0*K1K*H1*Y(1)                                            T05 115
      IROW(9)=3                                                          T05 116
      JCOL(9)=2                                                          T05 117
      DFDY(9)=K2K*H2                                                     T05 118
      IROW(10)=3                                                         T05 119
      JCOL(10)=3                                                         T05 120
      DFDY(10)=A1*K1K*H1*Y(1)*Y(1)+A2*K2K*H2*Y(2)-2.0*D3DZ2-VDZ          T05 121
      IROW(11)=3                                                         T05 122
      JCOL(11)=6                                                         T05 123
      DFDY(11)=D3DZ2                                                     T05 124
C                                                                        T05 125
C     MESH POINTS 2,3,....,M-2                                           T05 126
C                                                                        T05 127
      J=1                                                                T05 128
      IF=14*M-44                                                         T05 129
      DO 3 I=12,IF,14                                                    T05 130
      J=J+3                                                              T05 131
      K1K=K1*EXP(A1*Y(J+2))                                              T05 132
      K2K=K2*EXP(A2*Y(J+2))                                              T05 133
      IROW(I)=J                                                          T05 134
      JCOL(I)=J-3                                                        T05 135
      DFDY(I)=D1DZ2+VDZ                                                  T05 136
      IROW(I+1)=J                                                        T05 137
      JCOL(I+1)=J                                                        T05 138
      DFDY(I+1)=-2.0*K1K*Y(J)-2.0*D1DZ2-VDZ                              T05 139
      IROW(I+2)=J                                                        T05 140
      JCOL(I+2)=J+2                                                      T05 141
      DFDY(I+2)=-A1*K1K*Y(J)*Y(J)                                        T05 142
      IROW(I+3)=J                                                        T05 143
      JCOL(I+3)=J+3                                                      T05 144
      DFDY(I+3)=D1DZ2                                                    T05 145
      IROW(I+4)=J+1                                                      T05 146
      JCOL(I+4)=J-2                                                      T05 147
      DFDY(I+4)=D2DZ2+VDZ                                                T05 148
      IROW(I+5)=J+1                                                      T05 149
      JCOL(I+5)=J                                                        T05 150
      DFDY(I+5)=K1K*Y(J)                                                 T05 151
      IROW(I+6)=J+1                                                      T05 152
```

```
      JCOL(I+6)=J+1                                                    T05 153
      DFDY(I+6)=-K2K-2.0*D2DZ2-VDZ                                     T05 154
      IROW(I+7)=J+1                                                    T05 155
      JCOL(I+7)=J+2                                                    T05 156
      DFDY(I+7)=-A2*K2K*Y(J+1)+0.5*A1*K1K*Y(J)*Y(J)                    T05 157
      IROW(I+8)=J+1                                                    T05 158
      JCOL(I+8)=J+4                                                    T05 159
      DFDY(I+8)=D2DZ2                                                  T05 160
      IROW(I+9)=J+2                                                    T05 161
      JCOL(I+9)=J-1                                                    T05 162
      DFDY(I+9)=D3DZ2+VDZ                                              T05 163
      IROW(I+10)=J+2                                                   T05 164
      JCOL(I+10)=J                                                     T05 165
      DFDY(I+10)=2.0*K1K*H1*Y(J)                                       T05 166
      IROW(I+11)=J+2                                                   T05 167
      JCOL(I+11)=J+1                                                   T05 168
      DFDY(I+11)=K2K*H2                                                T05 169
      IROW(I+12)=J+2                                                   T05 170
      JCOL(I+12)=J+2                                                   T05 171
      DFDY(I+12)=A1*K1K*H1*Y(J)*Y(J)+A2*K2K*H2*Y(J+1)-2.0*D3DZ2-VDZ    T05 172
      IROW(I+13)=J+2                                                   T05 173
      JCOL(I+13)=J+5                                                   T05 174
      DFDY(I+13)=D3DZ2                                                 T05 175
3     CONTINUE                                                         T05 176
C                                                                      T05 177
C     MESH POINT M-1                                                   T05 178
C                                                                      T05 179
      I=14*M-30                                                        T05 180
      J=3*M-5                                                          T05 181
      K1K=K1*EXP(A1*Y(J+2))                                            T05 182
      K2K=K2*EXP(A2*Y(J+2))                                            T05 183
      IROW(I)=J                                                        T05 184
      JCOL(I)=J-3                                                      T05 185
      DFDY(I)=D1DZ2+VDZ                                                T05 186
      IROW(I+1)=J                                                      T05 187
      JCOL(I+1)=J                                                      T05 188
      DFDY(I+1)=-2.0*K1K*Y(J)-D1DZ2-VDZ                                T05 189
      IROW(I+2)=J                                                      T05 190
      JCOL(I+2)=J+2                                                    T05 191
      DFDY(I+2)=-A1*K1K*Y(J)*Y(J)                                      T05 192
      IROW(I+3)=J+1                                                    T05 193
      JCOL(I+3)=J-2                                                    T05 194
      DFDY(I+3)=D2DZ2+VDZ                                              T05 195
      IROW(I+4)=J+1                                                    T05 196
      JCOL(I+4)=J                                                      T05 197
      DFDY(I+4)=K1K*Y(J)                                               T05 198
      IROW(I+5)=J+1                                                    T05 199
      JCOL(I+5)=J+1                                                    T05 200
      DFDY(I+5)=-K2K-D2DZ2-VDZ                                         T05 201
      IROW(I+6)=J+1                                                    T05 202
      JCOL(I+6)=J+2                                                    T05 203
      DFDY(I+6)=-A2*K2K*Y(J+1)+0.5*A1*K1K*Y(J)*Y(J)                    T05 204
```

```
      IROW(I+7)=J+2                                                          T05 205
      JCOL(I+7)=J-1                                                          T05 206
      DFDY(I+7)=D3DZ2+VDZ                                                    T05 207
      IROW(I+8)=J+2                                                          T05 208
      JCOL(I+8)=J                                                            T05 209
      DFDY(I+8)=2.0*H1*K1K*Y(J)                                              T05 210
      IROW(I+9)=J+2                                                          T05 211
      JCOL(I+9)=J+1                                                          T05 212
      DFDY(I+9)=K2K*H2                                                       T05 213
      IROW(I+10)=J+2                                                         T05 214
      JCOL(I+10)=J+2                                                         T05 215
      DFDY(I+10)=A1*K1K*H1*Y(J)*Y(J)+A2*K2K*H2*Y(J+1)-D3DZ2-VDZ              T05 216
    4 CONTINUE                                                               T05 217
C                                                                            T05 218
C     CALCULATE DERIVATIVES                                                  T05 219
C                                                                            T05 220
C                                                                            T05 221
C     MESH POINT 1                                                           T05 222
C                                                                            T05 223
      K1K=K1*EXP(A1*Y(3))                                                    T05 224
      K2K=K2*EXP(A2*Y(3))                                                    T05 225
      DERY(1)=-K1K*Y(1)*Y(1)+D1DZ2*(Y(4)-2.0*Y(1)+C10)-VDZ*(Y(1)-C10)        T05 226
      DERY(2)=-K2K*Y(2)+0.5*K1K*Y(1)*Y(1)+D2DZ2*(Y(5)-2.0*Y(2)+C20)-VDZ*T05 227
                                                                           T05 228
     1(Y(2)-C20)                                                           T05 229
      DERY(3)=K1K*H1*Y(1)*Y(1)+K2K*H2*Y(2)+D3DZ2*(Y(6)-2.0*Y(3)+T0)-VDZ*T05 230
     1(Y(3)-T0)                                                            T05 231
C                                                                          T05 232
C     MESH POINTS 2,3,...,M-2                                              T05 233
C                                                                          T05 234
      IF=3*M-8                                                             T05 235
      DO 5 I=4,IF,3                                                        T05 236
      K1K=K1*EXP(A1*Y(I+2))                                                T05 237
      K2K=K2*EXP(A2*Y(I+2))                                                T05 238
      DERY(I)=-K1K*Y(I)*Y(I)+D1DZ2*(Y(I+3)-2.0*Y(I)+Y(I-3))-VDZ*(Y(I)-Y(T05 239
     1I-3))                                                                T05 240
      DERY(I+1)=-K2K*Y(I+1)+0.5*K1K*Y(I)*Y(I)+D2DZ2*(Y(I+4)-2.0*Y(I+1)+YT05 241
     1(I-2))-VDZ*(Y(I+1)-Y(I-2))                                           T05 242
      DERY(I+2)=K1K*H1*Y(I)*Y(I)+K2K*H2*Y(I+1)+D3DZ2*(Y(I+5)-2.0*Y(I+2)+T05 243
     1Y(I-1))-VDZ*(Y(I+2)-Y(I-1))                                          T05 244
    5 CONTINUE                                                             T05 245
C                                                                          T05 246
C     MESH POINT M-1                                                       T05 247
C                                                                          T05 248
      I=3*M-5                                                              T05 249
      K1K=K1*EXP(A1*Y(I+2))                                                T05 250
      K2K=K2*EXP(A2*Y(I+2))                                                T05 251
      DERY(I)=-K1K*Y(I)*Y(I)+D1DZ2*(Y(I-3)-Y(I))-VDZ*(Y(I)-Y(I-3))         T05 252
      DERY(I+1)=-K2K*Y(I+1)+0.5*K1K*Y(I)*Y(I)+D2DZ2*(Y(I-2)-Y(I+1))-VDZ*T05 253
     1(Y(I+1)-Y(I-2))                                                      T05 254
      DERY(I+2)=K1K*H1*Y(I)*Y(I)+K2K*H2*Y(I+1)+D3DZ2*(Y(I-1)-Y(I+2))-VDZT05 255
     1*(Y(I+2)-Y(I-1))                                                     T05 256
      CALL DIFSUB (N,Y,DERY)
```

```
      IF (IDERY.NE.0) GO TO 4                                          T05 257
C                                                                      T05 258
C     CALCULATE STREAM OUTPUT                                          T05 259
C                                                                      T05 260
      C30=S(1,IN,3)*S(1,IN,8)                                          T05 261
      C1=Y(N-2)                                                        T05 262
      C2=Y(N-1)                                                        T05 263
      C3=C30+0.5*(C10-C1)+C20-C2                                       T05 264
      SUM=C1+C2+C3                                                     T05 265
      S(1,IOUT,4)=Y(N)                                                 T05 266
      S(1,IOUT,6)=C1/SUM                                               T05 267
      S(1,IOUT,7)=C2/SUM                                               T05 268
      S(1,IOUT,8)=C3/SUM                                               T05 269
      RETURN                                                           T05 270
      END                                                              T05 271
```

```
**********************************************************************
                        TUBULAR REACTOR SIMULATION
**********************************************************************
BEGIN
TIME          5.0
HMAX          1.0
IN/OUT        2.0
COMPS         3.0
LIBRARY       1.0
REAC3         5.0
PROCESS
REAC3         1.0
              1.0          -2.0
              30.0         20.0          90.0          100.0         100.0
EXTRA         7.0
              1.0          50.0          1.5           0.00002       0.01
              0.07         75.0
END
STREAMS       2.0
EXPLICIT
              1.0          1.0           10.0          100.0         14.7
              1.0          0.0           0.0
              0.0          1.0           10.0          0.0           14.7
              0.0          0.0           1.0
END
PROPERTIES    -1.0
END
      END
```

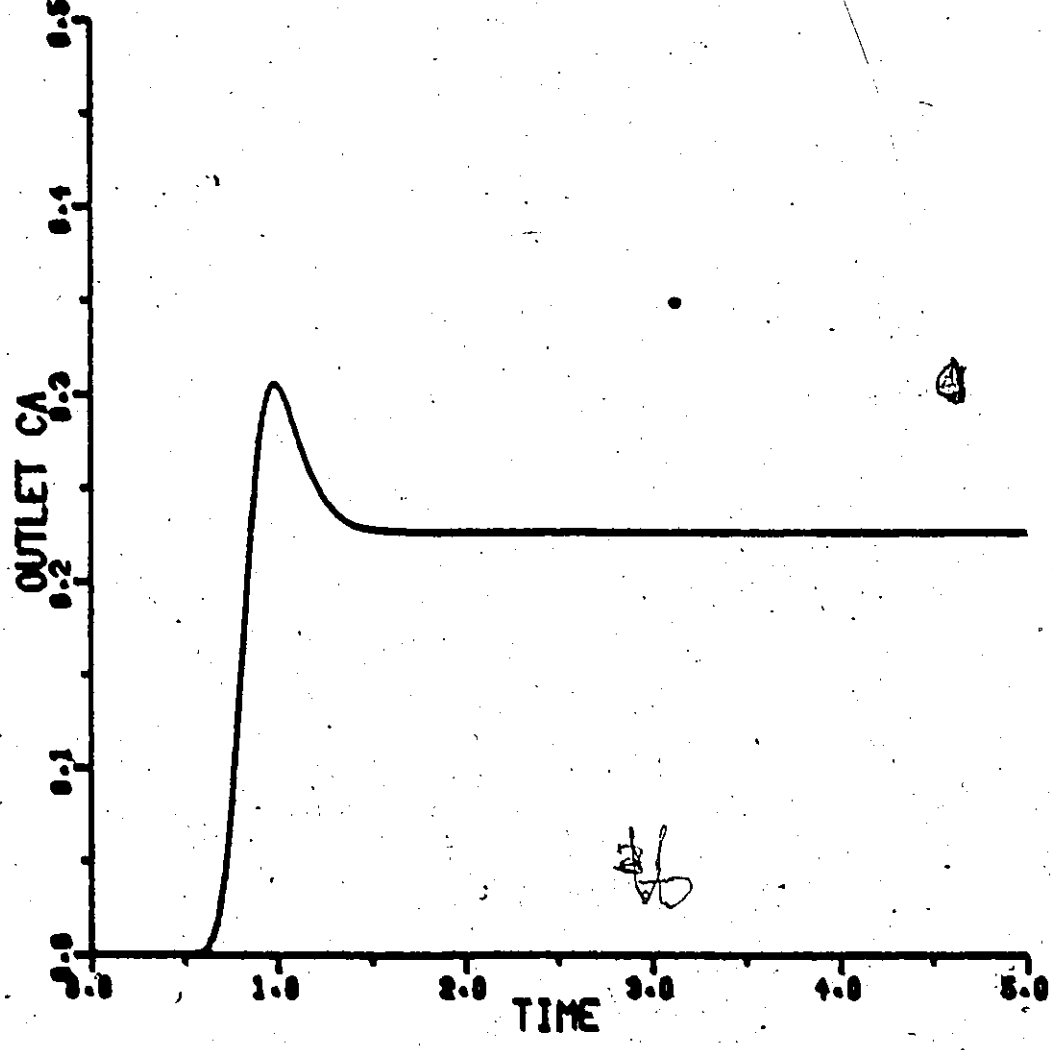


FIGURE 8.11: GRAPH OF OUTLET CONCENTRATION OF A VERSUS TIME FOR EXAMPLE 3

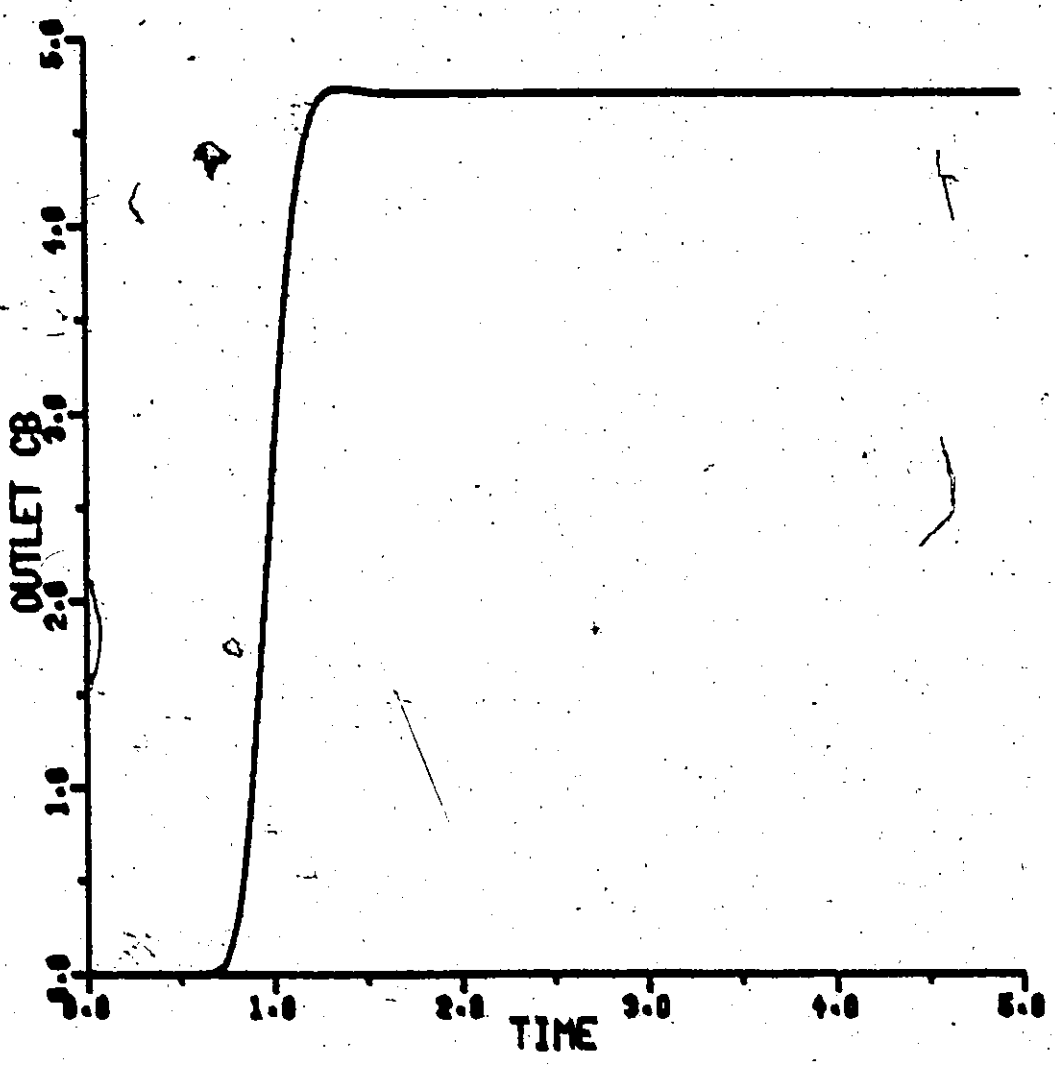


FIGURE 8.12: GRAPH OF OUTLET CONCENTRATION OF B VERSUS TIME FOR EXAMPLE 3

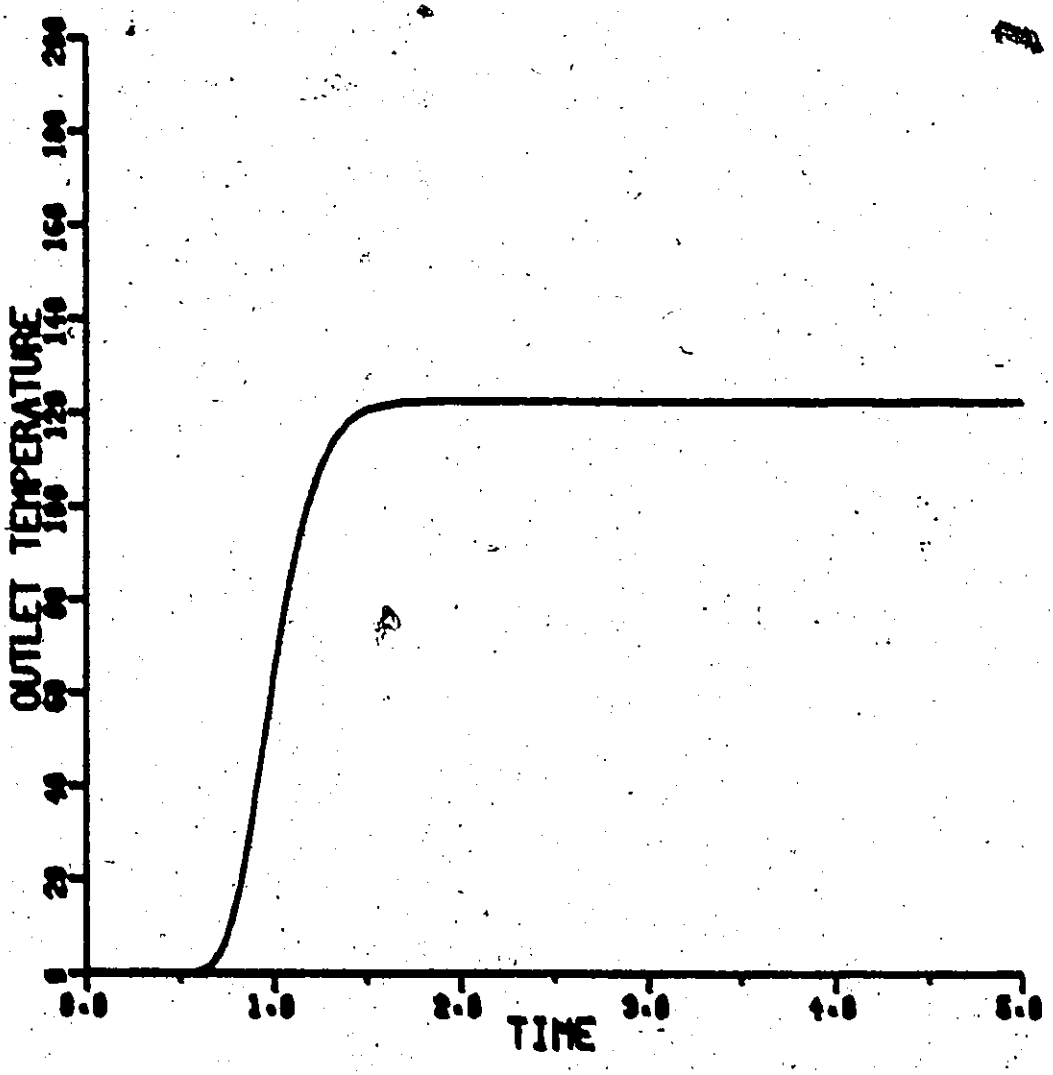


FIGURE 8.13: GRAPH OF OUTLET TEMPERATURE VERSUS TIME FOR EXAMPLE-3

8.4 Tubular Reactor Simulation With Tridiagonal Jacobian Matrix

An isothermal tubular reactor was simulated yielding a tridiagonal Jacobian matrix. The reaction $A \xrightarrow{k} B$ occurs.

The partial differential equation describing the process is:

$$\frac{\partial C_A}{\partial t} = -k\, C_A^{\,2} + D_{AB} \frac{\partial^2 C_A}{\partial Z^2} - V \frac{\partial C_A}{\partial Z} \qquad (8.4.1)$$

The following boundary and initial conditions apply:

$$\begin{aligned} C_A &= 0 \text{ at } t=0 \text{ for all } Z \\ C_A &= C_A^{\,o} \text{ at } Z=0 \text{ for all } t \\ \frac{\partial C_A}{\partial Z} &= 0 \text{ at } Z=L \text{ for all } t \end{aligned} \qquad (8.4.2)$$

where C_A ≡ concentration of component A

t ≡ time

Z ≡ space coordinate along reactor axis

$C_A^{\,o}$ ≡ feed concentration of component A

L ≡ reactor length

V ≡ fluid velocity

D_{AB} ≡ diffusivity of component A

k ≡ reaction rate constant

The parameter values used are:

$$\begin{aligned} D &= 30.0 \\ V &= 100.0 \\ L &= 100.0 \\ k &= 10.0 \\ C_A^{\circ} &= 10.0 \\ C_B^{\circ} &= 0.0 \\ t_{max} &= 5.0 \end{aligned} \qquad (8.4.3)$$

Dividing the reactor into M mesh points as in the previous example, each mesh point will have 1 o.d.e (except at the inlet and outlet)

$$\frac{dC_{A,i}}{dt} = -kC_{A,i}^2 + D_{AB}\left[\frac{C_{A,i+1}-2C_{A,i}+C_{A,i-1}}{(\Delta Z)^2}\right] - V\left[\frac{C_{A,i+1}-C_{A,i}}{\Delta Z}\right] \qquad (8.4.4)$$

The equations are moderately stiff.

The equation at mesh point 1 is:

$$\frac{dy_1}{dt} = -ky_1^2 + D_{AB}\left[\frac{y_2-2y_1+y_0}{(\Delta Z)^2}\right] - V\left[\frac{y_2-y_1}{\Delta Z}\right] \qquad (8.4.5)$$

The equations at mesh points 2,3,...M-2 are:

$$\frac{dy_i}{dt} = -ky_i^2 + D_{AB}\left[\frac{y_{i+1}-2y_i+y_{i-1}}{(\Delta Z)^2}\right] - V\left[\frac{y_{i+1}-y_i}{\Delta Z}\right] \qquad (8.4.6)$$

The equation at the last mesh point is:

$$\frac{dy_n}{dt} = -ky_n^2 + D_{AB}\left[\frac{y_{n-1}-y_n}{(\Delta Z)^2}\right] \qquad (8.4.7)$$

where y refers to C_A.

51 mesh points were taken giving a total of 49 o.d.e.s.

The execution time for the simulation was 4.04 seconds while an equation-oriented version of Gear's method took 3.92 seconds. An equation-oriented version using TRGB-TRGB2 required 6.78 seconds. Thus the tri-diagonal option does permit time savings. The IMP package required 46.1 seconds.

A list of the module and its data set are given in Figure 8.14. A graph of the outlet value of C_B versus time is shown in Figure 8.15. No change in the outlet concentration is seen until about 0.5 time units. Steady state is reached at about 1.2. The graph is sigmoid in appearance.

FIGURE 8.14: LISTING OF MODULE AND DATA SET FOR EXAMPLE #4
TUBULAR REACTOR WITH TRIDIAGONAL JACOBIAN MATRIX (49 ODES)

```
      SUBROUTINE TYPE6                                                    T06    1
C                                                                         T06    2
C     SUBROUTINE REAC4                                                    T06    3
C                                                                         T06    4
C     THIS MODULE REPRESENTS A TUBULAR REACTOR WITH REACTION C1=C2        T06    5
C     FEED CONCENTRATIONS MUST BE FAIRLY CONSTANT OTHERWISE               T06    6
C     ODES MUST BE SOLVED FOR C2                                          T06    7
C     THE JACOBIAN MATRIX IS TRIDIAGONAL                                  T06    8
C                                                                         T06    9
C     EQUIPMENT PARAMETERS                                                T06   10
C                                                                         T06   11
C     1 - D - DIFFUSIVITY OF COMPONENT 1                                  T06   12
C     2 - V - FLUID VELOCITY                                              T06   13
C     3 - L - LENGTH OF REACTOR                                           T06   14
C     4 - K - REACTION RATE CONSTANT                                      T06   15
C     5 - M - NUMBER OF SECTIONS IN REACTOR                               T06   16
C                                                                         T06   17
      COMMON /MAT/ MP(1,5),EP(1,5),S(2,2,8),EX(7)                         T06   18
      COMMON /CON/ IG,NCOMP,NC5,M,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH       T06   19
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NQVMAX,ICONV,NOPPT,ISTIFF      T06   20
      COMMON /UNIT/ IM,NMP                                                T06   21
      COMMON /SUBDI/ A(49)                                                T06   22
      COMMON /DIAG/ B(49)                                                 T06   23
      COMMON /SUPERD/ C(49)                                               T06   24
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO                               T06   25
      DIMENSION Y(49), DERY(49)                                           T06   26
      REAL K,L                                                            T06   27
C                                                                         T06   28
C     USE NEWTON-RAPHSON ITERATION FOR THIS MODULE                        T06   29
C                                                                         T06   30
      ITER=1                                                              T06   31
C                                                                         T06   32
C     CALCULATE MODULE PARAMETERS                                         T06   33
C                                                                         T06   34
      IF (JSTART.NE.0.OR.IG.EQ.1) GO TO 2                                 T06   35
      D=EP(IM,1)                                                          T06   36
      V=EP(IM,2)                                                          T06   37
      L=EP(IM,3)                                                          T06   38
      K=EP(IM,4)                                                          T06   39
      M=EP(IM,5)                                                          T06   40
      DZ=L/FLOAT(M)                                                       T06   41
      DZ2=DZ*DZ                                                           T06   42
      DDZ2=D/DZ2                                                          T06   43
      VDZ=V/DZ                                                            T06   44
      N=M-1                                                               T06   45
C                                                                         T06   46
C     CALCULATE INITIAL CONDITIONS                                        T06   47
C                                                                         T06   48
```

```
      N=M-1                                                              T06  49
      IN=MP(IM,3)                                                        T06  50
      IOUT=IABS(MP(IM,4))                                                T06  51
      DO 1 I=1,N                                                         T06  52
      Y(I)=S(2,IOUT,3)*S(2,IOUT,6)                                       T06  53
1     CONTINUE                                                           T06  54
2     CONTINUE                                                           T06  55
C                                                                        T06  56
C     CALCULATE FEED CONDITIONS                                          T06  57
C                                                                        T06  58
      C10=S(IG,IN,3)*S(IG,IN,6)                                          T06  59
C                                                                        T06  60
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS                        T06  61
C                                                                        T06  62
      IF (IG.EQ.2) GO TO 4                                               T06  63
C                                                                        T06  64
C     MESH POINT 1                                                       T06  65
C                                                                        T06  66
      B(1)=-2.0*K*Y(1)-2.0*DDZ2-VDZ                                      T06  67
      C(1)=DDZ2                                                          T06  68
C                                                                        T06  69
C     MESH POINTS 2,...,M-2                                              T06  70
C                                                                        T06  71
      M2=M-2                                                             T06  72
      DO 3 I=2,M2                                                        T06  73
      A(I)=DDZ2+VDZ                                                      T06  74
      B(I)=-2.0*K*Y(I)-2.0*DDZ2-VDZ                                      T06  75
      C(I)=DDZ2                                                          T06  76
3     CONTINUE                                                           T06  77
C                                                                        T06  78
C     MESH POINT M-1 = N                                                 T06  79
C                                                                        T06  80
      A(N)=DDZ2+VDZ                                                      T06  81
      B(N)=-2.0*K*Y(N)-DDZ2-VDZ                                          T06  82
4     CONTINUE                                                           T06  83
C                                                                        T06  84
C     USE TRIDIAGONAL OPTION FOR THIS MODULE (ITRI=1)                    T06  85
C                                                                        T06  86
      ITRI=1                                                             T06  87
C                                                                        T06  88
C     CALCULATE DERIVATIVES                                              T06  89
C                                                                        T06  90
C     MESH POINT 1                                                       T06  91
C                                                                        T06  92
      DERY(1)=-K*Y(1)*Y(1)+DDZ2*(Y(2)-2.0*Y(1)+C10)-VDZ*(Y(1)-C10)       T06  93
C                                                                        T06  94
C     MESH POINTS 2,...,M-2                                              T06  95
C                                                                        T06  96
      M2=M-2                                                             T06  97
      DO 5 I=2,M2                                                        T06  98
      DERY(I)=-K*Y(I)*Y(I)+DDZ2*(Y(I+1)-2.0*Y(I)+Y(I-1))-VDZ*(Y(I)-Y(I-1T06  99
     1))                                                                 T06 100
```

```
5     CONTINUE                                                                T06 101
C                                                                             T06 102
C     MESH POINT M=1 = N                                                      T06 103
C                                                                             T06 104
      DERY(N)=-K*Y(N)*Y(N)+DDZ2*(Y(N-1)-Y(N))-VDZ*(Y(N)-Y(N-1))               T06 105
      CALL DIFSUB (N,Y,DERY)                                                  T06 106
      IF (IDERY.NE.0) GO TO 4                                                 T06 107
C                                                                             T06 108
C     CALCULATE STREAM OUTPUT                                                 T06 109
C                                                                             T06 110
      C20=S(1,IN,3)*S(1,IN,7)                                                 T06 111
      C1=Y(N)                                                                 T06 112
      C2=C20+C10-C1                                                           T06 113
      SUM=C1+C2                                                               T06 114
      S(1,IOUT,6)=C1/SUM                                                      T06 115
      S(1,IOUT,7)=C2/SUM                                                      T06 116
      RETURN                                                                  T06 117
      END                                                                     T06 118
```

```
**********************************************************************
          TUBULAR REACTOR SIMULATION - TRIDIAGONAL JACOBIAN MATRIX
**********************************************************************
BEGIN
TIME          5.0
DELTAT        0.0001
IN/OUT        2.0
HMAX          1.0
COMPS         2.0
LIBRARY       1.0
REAC4         6.0
PROCESS
REAC4         1.0
              1.0          -2.0
              30.0         100.0        100.0        10.0         50.0
END
STREAMS       2.0
EXPLICIT
              1.0          1.0          10.0         100.0        14.7
              1.0          0.0
              2.0          1.0          10.0         100.0        14.7
              0.0          1.0
END
PROPERTIES    -1.0
END
      END
```

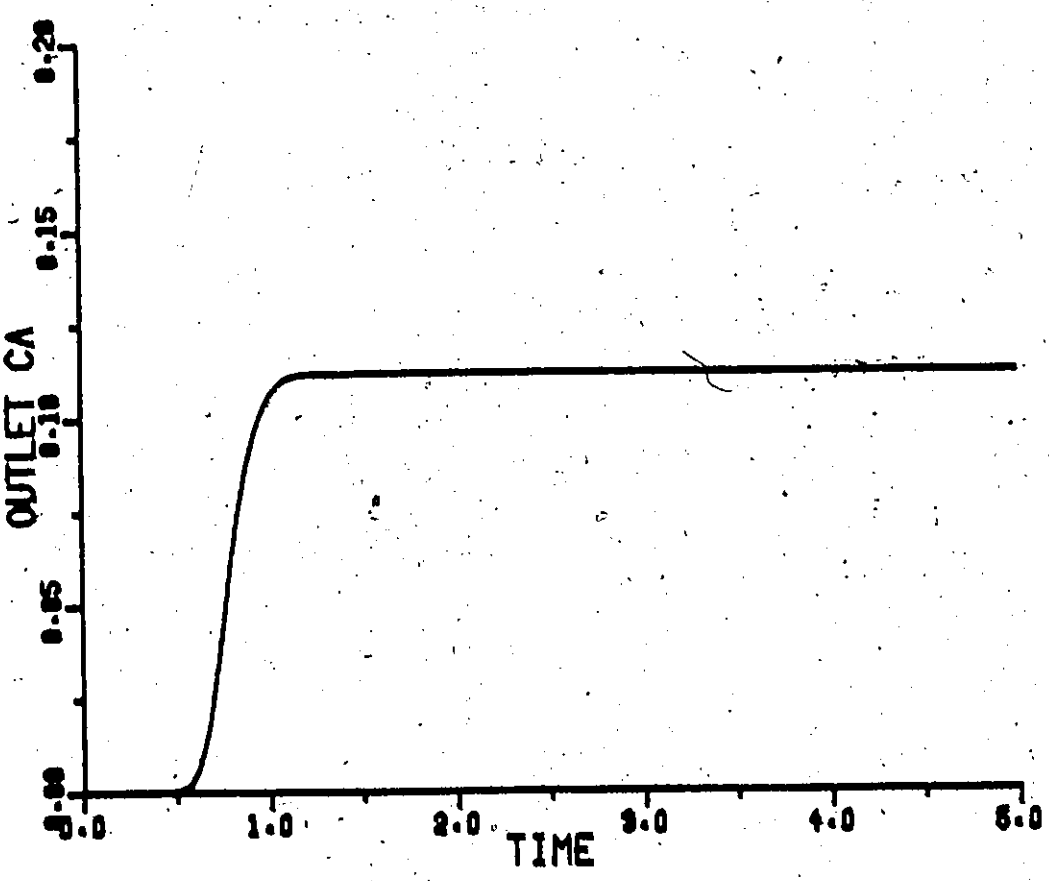


FIGURE 8.15: GRAPH OF OUTLET CONCENTRATION OF A VERSUS TIME FOR EXAMPLE#4

9. THE WILLIAMS-OTTO PLANT SIMULATION

The examples of the previous chapter were quite simple since they were only intended to illustrate different capabilities of the DYNSYS 2.0 package. In this chapter, we would like to show a simulation of a realistic chemical plant. The Williams-Otto plant (Williams and Otto, 1960; Williams, 1961) was chosen.

Around 1960, many computer control schemes had recently been proposed for use by the chemical and petroleum processing industries. There was no direct method of comparing the relative applicability of these schemes. For this reason, the Monsanto Chemical Company proposed this plant as a chemical processing model to those interested in computer control with the hope that it may serve as a basis for a direct comparison of the available computers and their relative capabilities. Every effort was made to have the model include as many as possible of the effects present in typical chemical manufacturing plants.

Williams assumed that a plant has already been designed and built for producing a chemical product *P*. However, considerable difficulty has been experienced with the stability of operation of the plant, and it is desired to investigate the feasibility of applying computer control

in an effort to achieve stable operation. At the same time, it is desired that the computer maintain an optimum balance of plant operating conditions to assure that the maximum return on investment is being obtained.

The plant consists of four major pieces of equipment, a stirred tank reactor, a heat exchanger, a decanter and a distillation column. Figure 9.1 depicts a simplified process flow diagram for the plant..

9.1 The Reactor

The reactor is a continuous stirred tank reactor in which the following reactions occur:

$$A + B \xrightarrow{k_1} C$$

$$B + C \xrightarrow{k_2} P + E$$

$$P + C \xrightarrow{k_3} G$$

The desired product of the plant is P, while G is a heavy oily waste material. Reactants A and B are fed to the reactor in pure form and there is a recycle stream containing A, B, C, E and P.

Steam is available to heat up the reactor and cooling water is available to keep the temperature under control, once the reactor has been heated up to about 180°F.

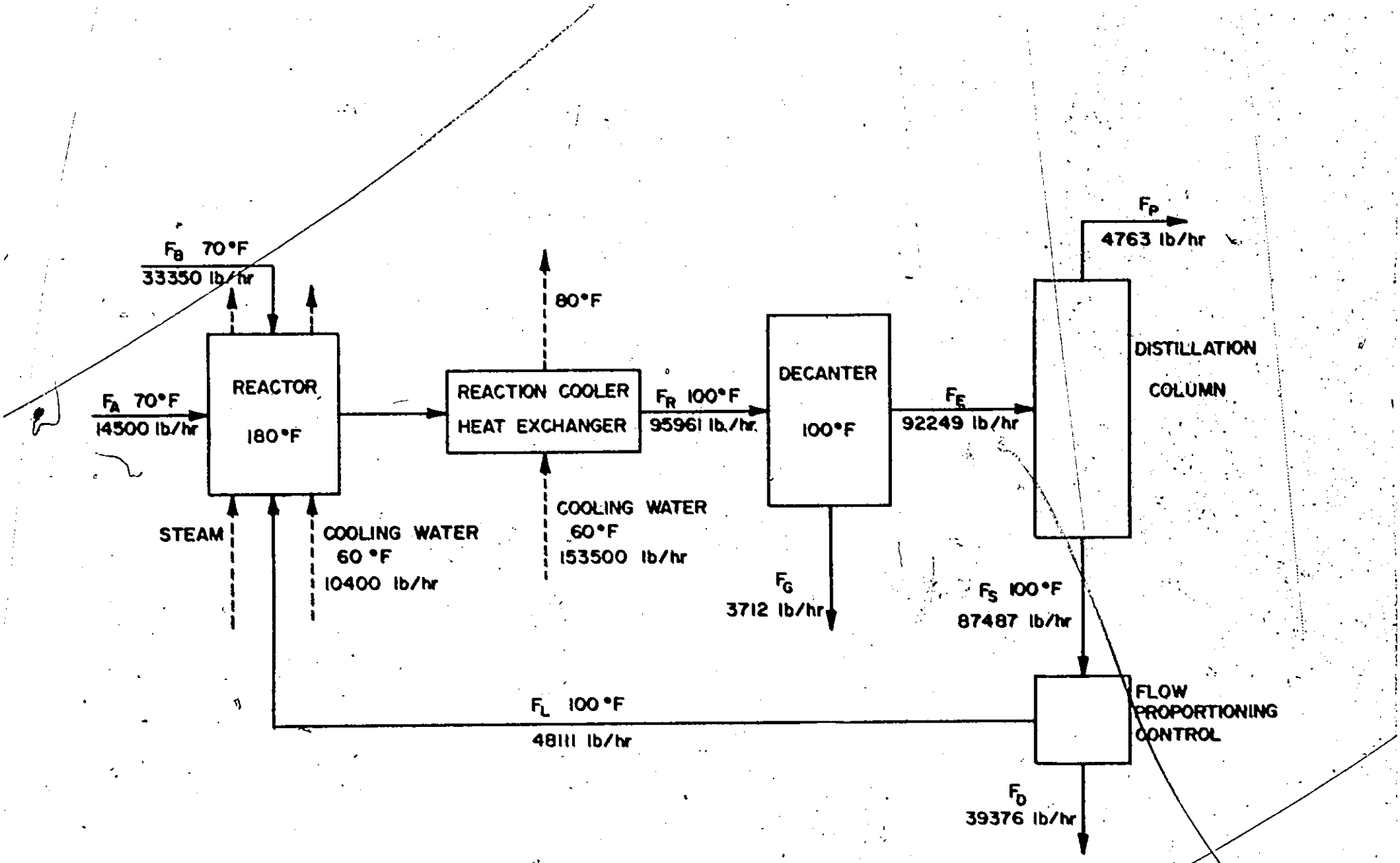


FIGURE 9.1 : SIMPLIFIED PROCESS FLOW DIAGRAM - WILLIAMS - OTTO PLANT

The following differential equations were used in the model:

mass balance on A:

$$\frac{dy_1}{dt} = \frac{1}{V_R}(F_A + F_L A_L - F_R y_1) - k_1 y_1 y_2 \qquad (9.1.1)$$

mass balance on B:

$$\frac{dy_2}{dt} = \frac{1}{V_R}(F_B + F_L B_L - F_R y_2) - k_1 y_1 y_2 - k_2 y_2 y_3 \qquad (9.1.2)$$

mass balance on C:

$$\frac{dy_3}{dt} = \frac{1}{V_R}(F_L C_L - F_R y_3) + 2.0\, k_1 y_1 y_2 - 2.0\, k_2 y_2 y_3 - k_3 y_3 y_6 \qquad (9.1.3)$$

mass balance on E:

$$\frac{dy_4}{dt} = \frac{1}{V_R}(F_L E_L - F_R y_4) + 2.0\, k_2 y_2 y_3 \qquad (9.1.4)$$

mass balance on G:

$$\frac{dy_5}{dt} = \frac{1}{V_R}(F_L G_L - F_R y_5) + 1.5\, y_3 y_6 \qquad (9.1.5)$$

mass balance on P:

$$\frac{dy_6}{dt} = \frac{1}{V_R}(F_L P_L - F_R y_6) + k_2 y_2 y_3 - 0.5 y_2 y_6 \qquad (9.1.6)$$

reactor heat balance:

$$\frac{dy_7}{dt} = \frac{1}{V_R C_{PR}}[-2.0 k_1 y_1 y_2 H_1 V_R - 3.0 k_2 y_2 y_3 H_2 V_R$$

$$-1.5 k_3 y_3 y_6 H_3 V_R - h_W a_W (y_7 - y_8)$$

$$+h_S a_S (T_S - y_7) - F_L C_{PR} (y_7 - T_L)$$

$$-F_A C_{PR} (y_7 - T_A) - F_B C_{PR} (y_7 - T_B)] \qquad (9.1.7)$$

cooling coil heat balance:

$$\frac{dy_8}{dt} = \frac{1}{V_W C_{PW}}[h_W a_W (y_7 - y_8) + F_W C_{PW} (T_{INC} - y_8)] \qquad (9.1.8)$$

total reactor mass balance:

$$\frac{dy_9}{dt} = F_A + F_B + F_L - F_R \qquad (9.1.9)$$

where

$$k_i = a_i \exp[-b_i/(y_7 + 459.69)] \qquad (9.1.10)$$

where

y_1-y_6 ≡ concentrations of A, B, C, E, G and P respectively in the reactor (mass fraction)

y_7 ≡ reactor temperature (°F)

y_8 ≡ cooling coil temperature (°F)

y_9, V_R ≡ reactor mass holdup (lb)

V_W ≡ cooling coil mass holdup (lb)

h_W, h_S ≡ overall heat transfer coefficient of cooling and steam coils respectively (BTU/hrft²°F)

a_W, a_S ≡ heat transfer area of cooling and steam coils respectively (ft²)

$A_L, B_L, C_L, E_L, G_L, P_L$ ≡ recycle stream concentrations of A, B, C, E, G and P respectively (mass fraction)

F_A, F_B, F_L, F_R, F_W ≡ flow rates of A feed stream, B feed stream, recycle stream, reactor exit stream and coolant respectively (lb/hr)

C_{PR}, C_{PW} ≡ heat capacities of reactor mixture (assumed same for all reactants) and coolant respectively (BTU/lb°F)

k_1, k_2, k_3 ≡ reaction rate coefficients (hr^{-1})

H_1, H_2, H_3 ≡ heats of reaction (BTU/lb)

T_{INC} ≡ inlet coolant temperature (°F)

T_S ≡ steam temperature (°F)

T_A, T_B, T_L ≡ temperatures of A feed stream, B feed stream and recycle stream respectively (°F)

a_i, b_i ≡ Arrhenius rate constants (hr^{-1}, °R)

The cooling coil is modelled as a perfectly stirred tank. A dynamic equation for the steam coil is not included since we assume a steam trap removes condensate instantly and the temperature of the coil is thus constant at the saturated steam temperature (assumed 250°F).

V_R is assumed constant for the first seven equations and is neglected in the Jacobian; however, a total mass balance is done over the whole reactor in equation (9.1.9) in order to provide a holdup signal to the controller. A signal is also sent to a controller which keeps the ratio F_A/F_B constant.

An information flow diagram for the reactor is shown in Figure 9.2. An on-off controller turns off the steam flow when the temperature reaches a certain value. A controller operates on the cooling water flow after the reactor is heated.

The reactor mass and heat balances are not stiff, but when the cooling coil heat balance is included, the model is stiff. Thus the module is stiff and a Jacobian matrix is supplied.

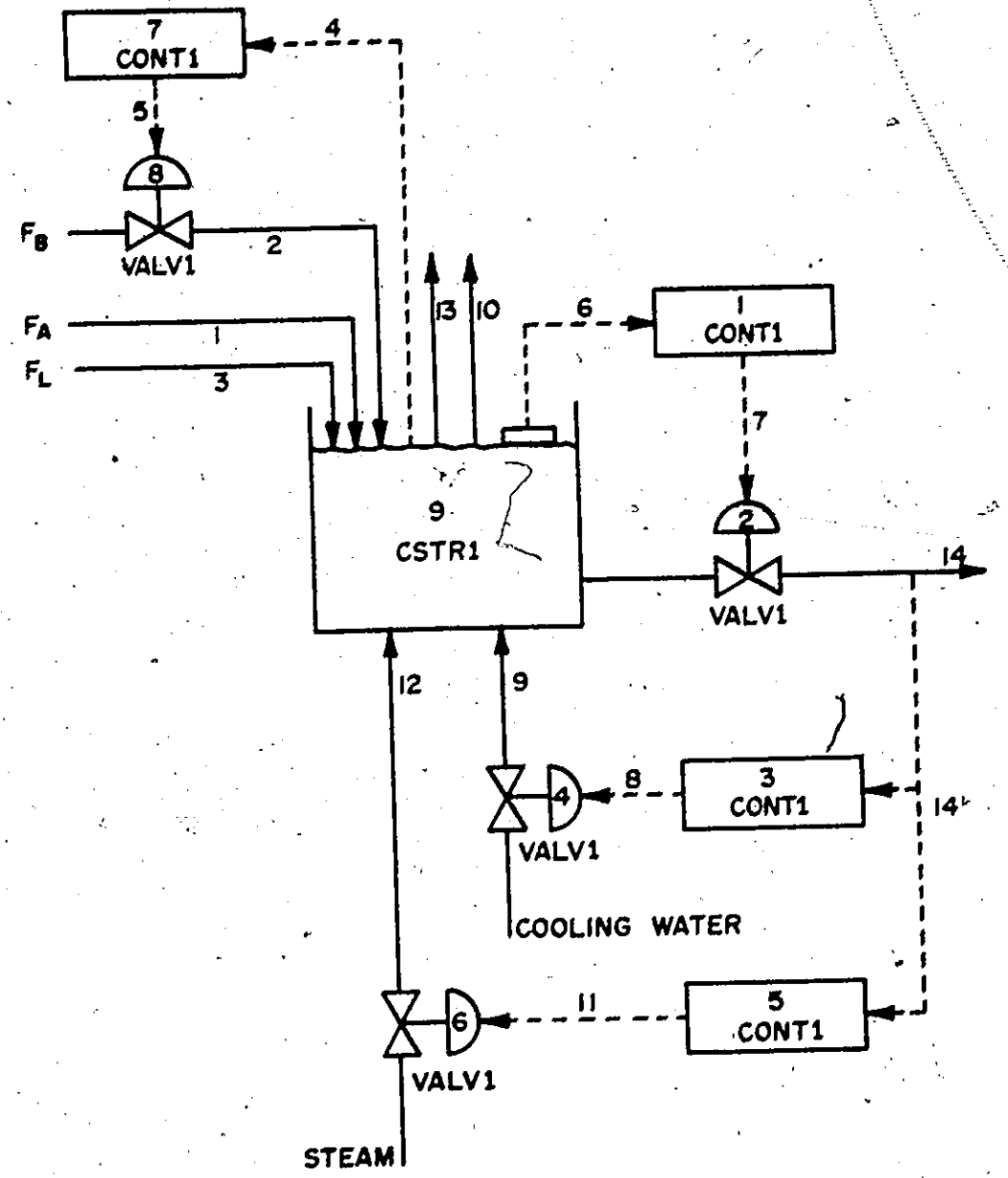


FIGURE 9.2 : REACTOR INFORMATION FLOW DIAGRAM

There are four equipment parameters:

(1) *VR* - initial holdup in reactor (lb)

(2) *VW* - holdup in cooling coil (lb)

(3) *HWAW* - product of overall cooling coil heat transfer coefficient and effective heat transfer area (BTU/°Fhr)

(4) *HSAS* - product of overall steam coil heat transfer coefficient and effective heat transfer area (BTU/°Fhr)

9.2 The Heat Exchanger

A heat exchanger follows the reactor in order to stop the reaction and prevent an overproduction of the waste material *G*. Both the tube and shell side of the exchanger are modelled as perfectly stirred tanks. It is assumed no reaction occurs in the heat exchanger.

The following differential equations were used in the model:

tube side heat balance:

$$\frac{dy_1}{dt} = \frac{1}{V_{HT}C_{PR}} [F_T C_{PR} (T_{INT} - y_1) + h_H a_H T_{LM}] \qquad (9.2.1)$$

where T_{LM} is the log mean temperature difference:

$$T_{LM} = \frac{(T_{INT} - y_2) - (y_1 - T_{INS})}{\ln\left(\frac{T_{INT} - y_2}{y_1 - T_{INS}}\right)} \qquad (9.2.2)$$

shell side heat balance (coolant):

$$\frac{dy_2}{dt} = \frac{1}{V_{HS}C_{PW}}\left[F_S C_{PW}(T_{INS} - y_2) + h_H a_H T_{LM}\right] \qquad (9.2.3)$$

component mass balances:

$$\frac{dy_{i+2}}{dt} = \frac{F_T}{V_{HT}}(C_i - y_{i+1}) \quad i=1,6 \qquad (9.2.4 - 9.2.9)$$

where

y_1 ≡ tube side temperature (°F)

y_2 ≡ shell side temperature (°F)

y_3-y_8 ≡ tube side concentrations of A, B, C, E, G and P respectively (mass fractions)

T_{INT}, T_{INS} ≡ inlet tube and shell side temperatures respectively (°F)

V_{HT}, V_{HS} ≡ tube and shell side mass holdups respectively (lb)

C_{PR}, C_{PW} ≡ tube and shell side heat capacities respectively (BTU/lb°F)

F_T, F_S ≡ tube and shell side flow rates respectively (lb/hr)

C_i ≡ inlet tube side concentrations of A, B, C, E, G and P respectively (mass fractions)

h_H ≡ overall heat transfer coefficient (BTU/°Fft²hr)

a_H ≡ area of heat transfer (ft²)

The module is nonstiff. An information flow diagram for the heat exchanger appears in Figure 9.3. The flow of the cooling water is controlled by the outlet temperature of the heat exchanger.

There are five equipment parameters:

(1) *VHT* - tube side mass holdup (lb)

(2) *VHS* - shell side (coolant) mass holdup (lb)

(3) *AH* - area of heat transfer (ft²)

(4) *HH* - overall heat transfer coefficient (BTU/°F ft²hr)

(5) *CPW* - coolant heat capacity (BTU/lb°F)

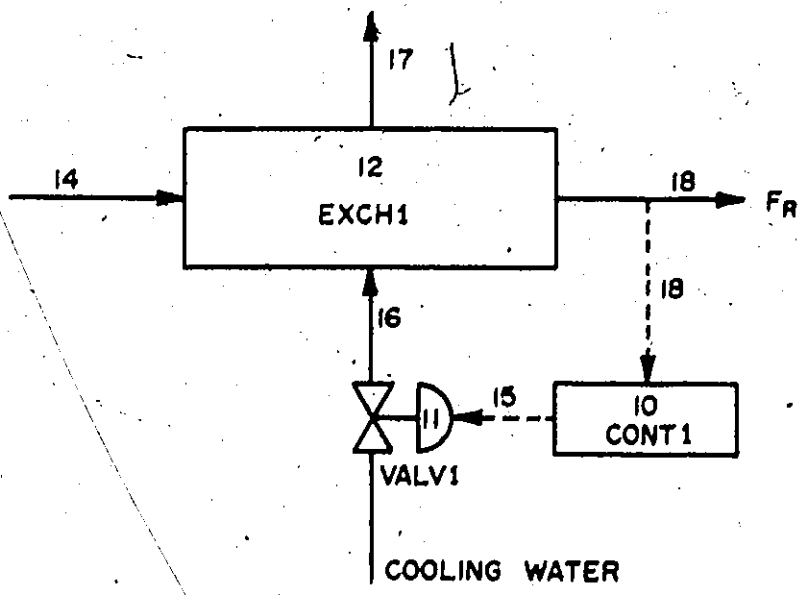


FIGURE 9.3 : HEAT EXCHANGER INFORMATION FLOW DIAGRAM

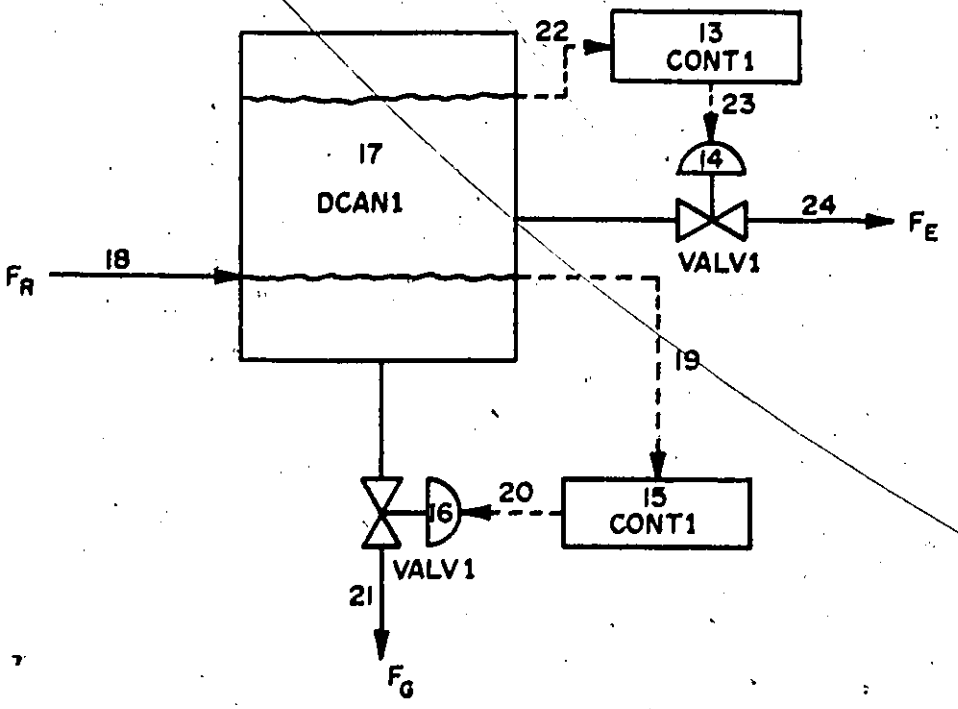


FIGURE 9.4 : DECANTER INFORMATION FLOW DIAGRAM

9.3 The Decanter

The heavy oil waste material, G, is insoluble at temperatures of less than 100°F and since it has a considerably higher specific gravity than the carrier stream in which it is suspended, it may be removed by settling. Thus a decanter follows the heat exchanger.

The top and bottom layers are both modelled as perfectly stirred tanks. The mass fraction of G in the reactant stream as a function of temperature is given. The decanter temperature determines the mass fraction of G entering the top layer.

The following differential equations were used in the model:

total mass balance:

$$\frac{dy_1}{dt} = F_{IN} - F_{OUTT} - F_{OUTB} \qquad (9.3.1)$$

total heat balance:

$$\frac{dy_2}{dt} = \frac{1}{y_1}\left[F_{IN}T_{IN} - (F_{OUTT} + F_{OUTB})\, y_2\right] \qquad (9.3.2)$$

component mass balances for the top layer:

$$\frac{dy_{i+2}}{dt} = F_{INT}y_{INT} - F_{OUTT}\ \frac{y_{i+2}}{\sum_{i=1}^{6} y_{i+2}} \qquad (9.3.3\text{-}9.3.8)$$

mass balance for bottom:

$$\frac{dy_2}{dt} = F_{INB} - F_{OUTB} \qquad (9.3.3)$$

where

y_1 ≡ total mass holdup (lb)

y_2 ≡ decanter temperature (°F)

y_{3-8} ≡ top layer holdups of A, B, C, E, G and P, respectively (lb/hr)

y_9 ≡ holdup of G in bottom layer (lb)

T_{IN} ≡ inlet temperature (°F)

F_{OUTT}, F_{OUTB} ≡ outlet flow rates from top and bottom layers respectively (lb/hr)

F_{INT}, F_{OUTB} ≡ inlet flow rates to top and bottom layers respectively (lb/hr)

F_{IN} ≡ total decanter inlet flow rate (lb/hr)

Figure 9.4 shows an information flow diagram for the decanter. The holdups in the top and bottom layers are controlled.

There are three equipment parameters:

(1) *HUPT* - initial holdup in top layer (lb)

(2) *HUPB* - initial holdup in bottom layer (lb)

(3) *ISEP* - number of component to be separated from entering stream

9.4 The Distillation Column

Williams and Otto proposed to model the column as a stirred tank. Rather expectedly, this did not give satisfactory results. Pulido (1975) has created DYNSYS 2.0 modules of a distillation column, condenser, reboiler and reboiler drum. These are discussed in detail in his thesis.

Assumptions used were no heat of mixing, constant holdup on each plate and equilibrium on each plate at all times. The column module contains mass and energy balances about each plate. Bubble point calculations are done. The condenser is either a total or a partial condenser. The reboiler module handles the heat transfer calculations, while the reboiler drum module does the mass and energy balances for the reboiler. The azeotrope in the column was neglected for simplicity, but it can be implemented once a correlation for the liquid activity coefficients is found, such as the Wilson equation.

The column, condenser drum, reboiler and reboiler drum modules are modelled as being stiff and Jacobian matrices are supplied. The condenser module is algebraic.

An information flow diagram for the column appears in Figure 9.5. Since the column uses molar rather than mass quantities, a module CONV1 converts the mass flow and mass fractions to molar flow and mole fractions respectively before entering the column and does the opposite for the

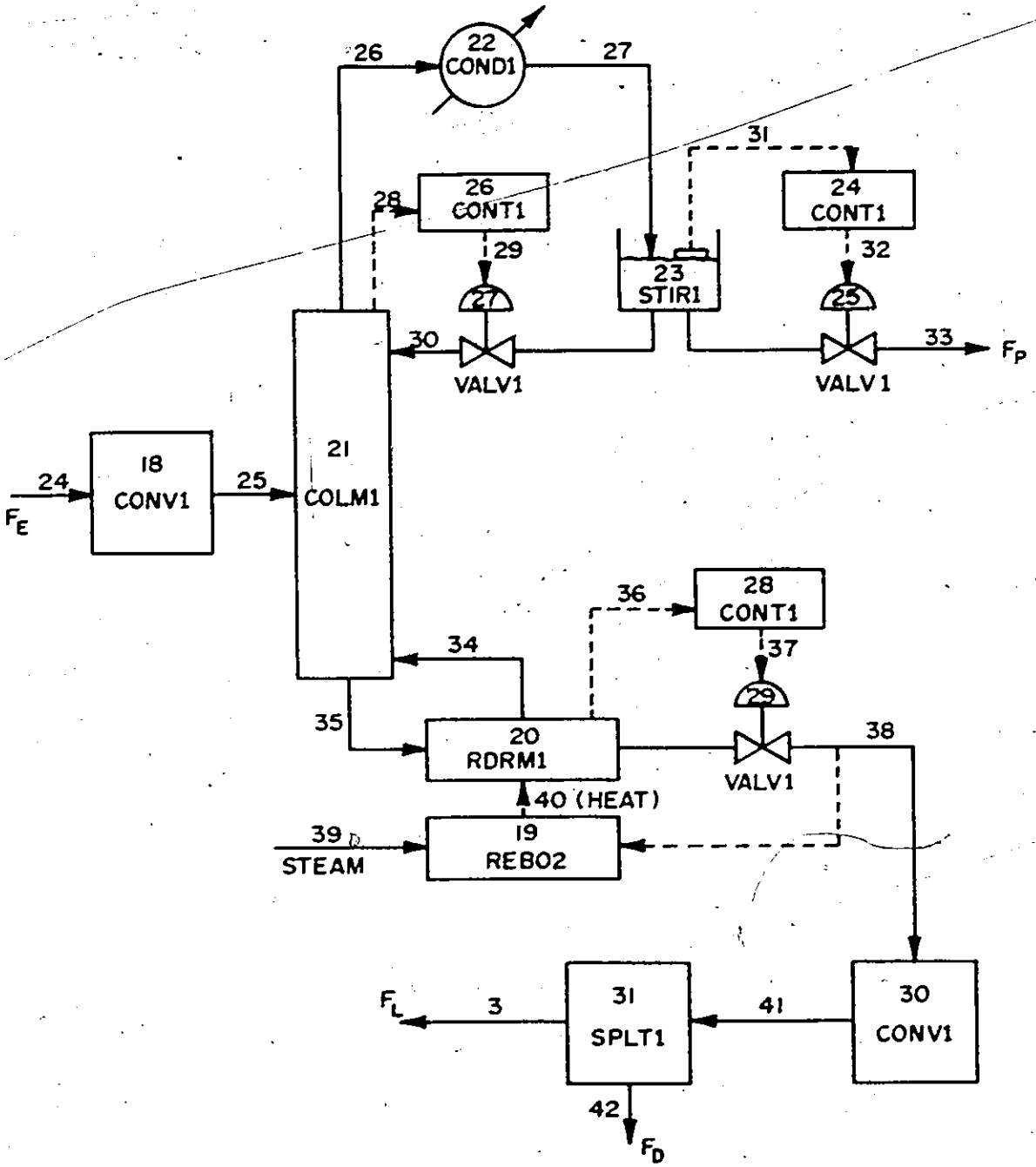


FIGURE 9.5: DISTILLATION COLUMN INFORMATION FLOW DIAGRAM

bottoms stream leaving the column.

A module SPLT1 divides the bottoms flow, according to a given ratio, into two streams, one of which is recycled back to the reactor.

9.5 Simulation Results

A listing of all the modules and a possible data set are contained in Appendix H. The modules form the basis of a library of modules for DYNSYS 2.0. In all, 31 modules and 42 streams were required.

The reactor was started up from 70°F assuming a constant recycle stream. Figure 9.6 shows a graph of the reactor concentrations versus time. Figure 9.7 shows a graph of the reactor and exit heat exchanger temperatures versus time. Steady state was reached after about 0.7 hours.

The control scheme for the reactor kept the temperature at 180°F after some slight oscillation. The steam flow is cut off at 175°F and a proportional controller with a high gain controls the cooling water flow so that the temperature does not rise above 180°F. This is similar to the control scheme recommended by Williams. The steady state cooling water flow rate at 180°F with no steam flow was calculated to be 5115 lb/hr. The reactor concentrations at steady state were the same as given by Williams.

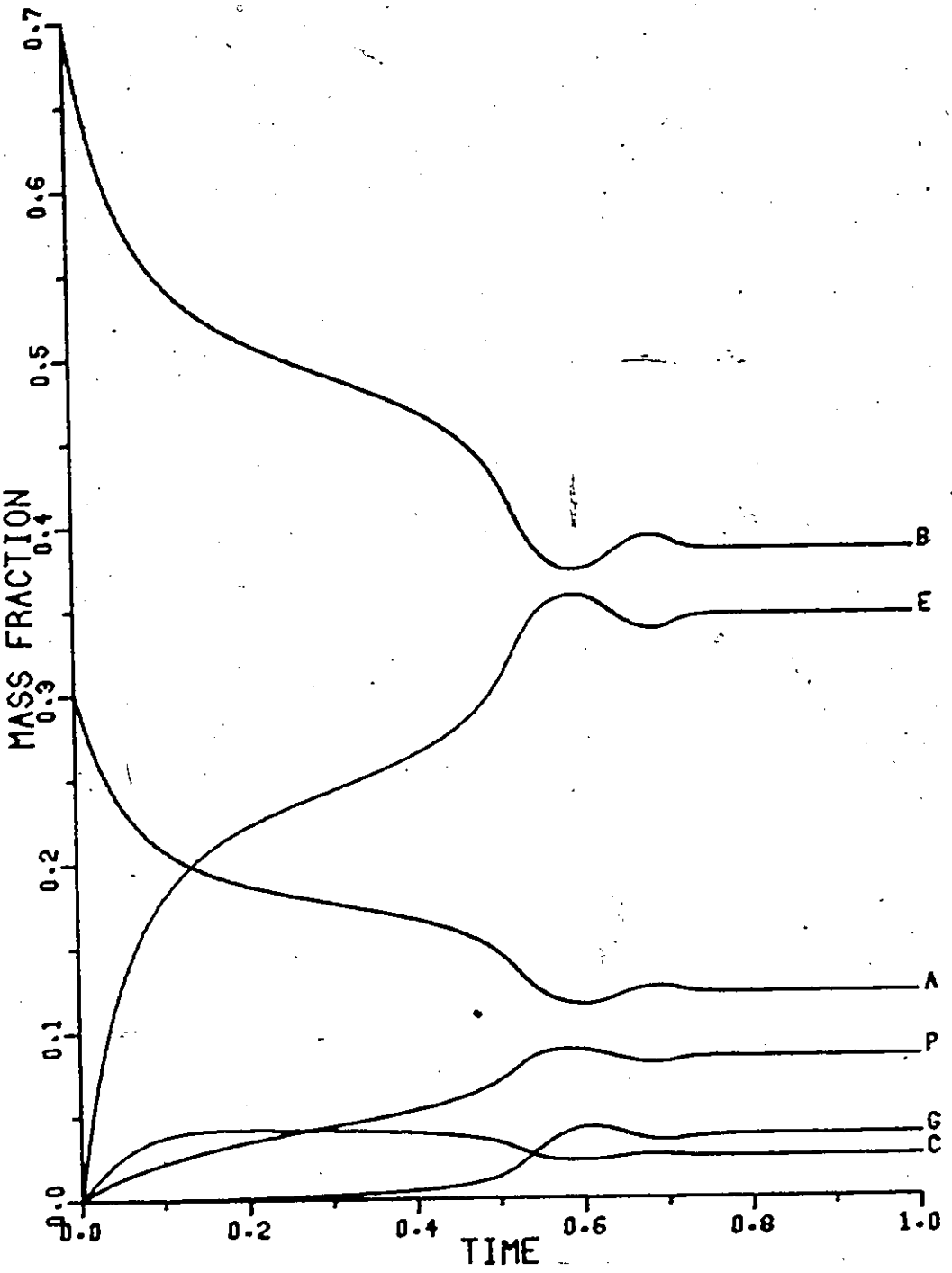


FIGURE 9.6: GRAPH OF REACTOR CONCENTRATIONS OF A.B.C.E.G AND P VERSUS TIME

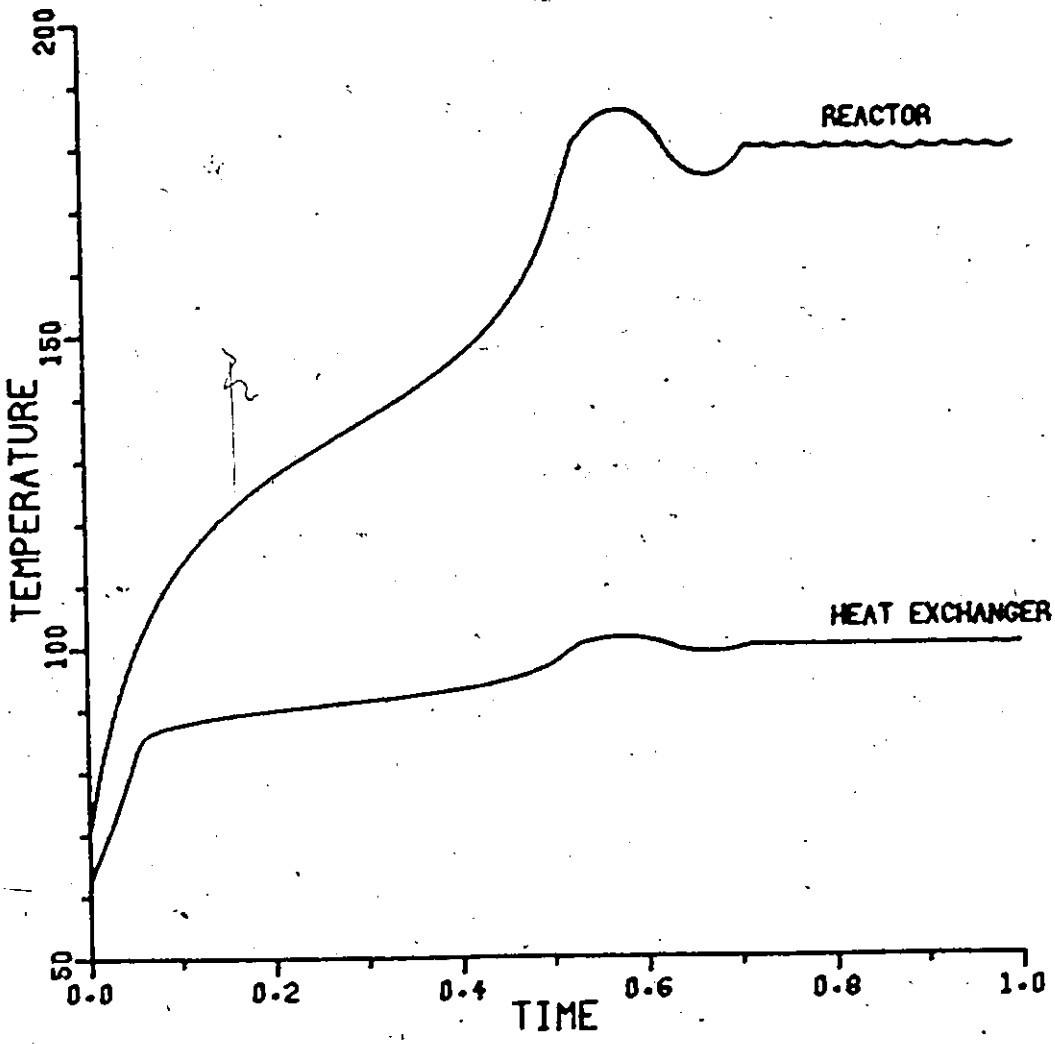


FIGURE 9.7: GRAPH OF REACTOR AND EXIT HEAT EXCHANGER TEMPERATURES VERSUS TIME

A value of 5000.0 was used for $h_s a_s$. The value of 10000.0 given by Williams heats the reactor too fast and the temperature becomes out of control.

The heat exchanger outlet temperature was below 102°F at all times. Since the reactor temperature control works well and the heat exchanger cools the reactor outlet stream efficiently, no trouble was encountered in the decanter. The decanter temperature never rose above 100°F and all of the G was settled out.

The distillation column results were handled by J. Pulido and a discussion of these may appear in his thesis (Pulido, 1975).

DYNSYS 2.0 performed the simulation quite well and is very convenient for simulating a modular plant such as this one. However, during the simulation, corrector convergence problems occurred often for the heat exchanger even with a supplied Jacobian. Possibly the model will have to be refined.

The Williams-Otto plant offers many possibilities for control system studies. More work may be done of this in the future. DYNSYS 2.0 is a very convenient tool for evaluating different control policies. Pulido (1975) has used DYNSYS 2.0 to evaluate various distillation column control schemes.

10. CONCLUSION

10.1 Summary

The numerical solution of ordinary differential equations is a fundamental aspect of dynamic simulation. The original version of DYNSYS used a third order Adams-Moulton-Shell routine; however, this is very inefficient for stiff systems where there is a wide variation of time constants. In chemical engineering, stiff o.d.e.s occur widely in reaction kinetics and to some extent in multi-stage systems. Conventional numerical techniques are restricted by stability to using a very small step size resulting in large computer times. There have been many new numerical techniques published in the recent literature directed at the efficient numerical solution of systems of stiff o.d.e.s. A literature survey of these has been made.

Most stiff techniques are implicit and require a technique such as Newton-Raphson iteration to converge. Each Newton-Raphson iteration involves the solution of a system of linear algebraic equations (usually sparse) equal in size to the number of o.d.e.s. For a large stiff system, this requires considerable computer time. Various

recently developed sparse linear equation solvers have been evaluated and that of Bending and Hutchison appears to be the most appropriate.

A comparison of several integration techniques has been done with a number of nonstiff and stiff test examples. Gear's method appears to be the most efficient.

Gear's integration algorithm in conjunction with the Bending-Hutchison linear equation solver has been implemented into DYNSYS version 2.0. An option for stiff systems with tridiagonal Jacobian matrix is also included. Several simulation examples have been described including the simulation of the Williams-Otto plant. The package is a very powerful one, capable of the simultaneous numerical solution of hundreds of stiff o.d.e.s.

The modular approach has been chosen as the framework of our study, but certainly the results apply equally well to the equation-oriented approach. An equation-oriented program containing Gear's method and the Bending-Hutchison linear equation solver has been created and it is useful for simulating nonmodular systems.

An interactive version of the executive, where the output can be displayed graphically on a CRT screen, has also been developed.

10.2 Future Work

A version of the executive which keeps the operator lists and possibly other variables in auxiliary storage can easily be created. Segments of the operator list can be stored or read in, thus allowing much larger systems to be simulated. Thousands of o.d.e.s could then be solved, but the program would not be as fast because of the extra time required to store and access the operator lists. It would also make the program more machine dependent.

A tridiagonal option has been included, but other options might well be included, similar to those in the IMP package, such as options for banded matrices in general rather than just tridiagonal, or iterative options for problems with diagonally dominant Jacobians.

This thesis has concentrated on the numerical solution of stiff systems, particularly large ones. Another area which can be studied is the convergence of recycle in dynamic simulation (Koenig, 1972).

With the present package, it is reasonably difficult for the novice to learn how to write modules. Ways of simplifying the module structure for user convenience should be studied.

A library of modules could be developed to make future programming easier. Pulido (1975) and Millares (1975) are presently using DYNSYS 2.0 to simulate a commercial

process. The modules from their work and from the Williams-Otto plant simulation would form the basis of such a library. Additional modules could be added corresponding to the types of equipments used in chemical plants.

The package should be applied to other industrial processes.

A physical properties package could be added to the system.

A version of DYNSYS which can use CRT graphical input in the form of an information flow diagram is being developed.

The applicability of the pseudo state approach is a possible area of research. We pointed out in Chapter 3 that it is a dangerous method to apply; however the circumstances under which it can be applied and the amount of error incurred are questions of interest in solving stiff systems.

All of the test examples in Appendix D are of the relaxation type; i.e. the fast components become essentially zero for a very short time. In a control system simulation with disturbances, the fast modes will be constantly re-introduced into the system. Whether Gear's method is the best numerical technique under such conditions should be investigated.

The study of numerical techniques for solving stiff o.d.e.s and for solving sparse linear algebraic equations is a currently active area of research and more efficient techniques than those used in DYNSYS 2.0 will undoubtedly be developed. Gear himself has a new, more efficient version of DIFSUB which will be available around the end of 1974 (Gear 1974). DYNSYS 2.0 will be modified to take this into account and the evolution of the package will include the incorporation of more efficient techniques.

APPENDIX A: GEAR'S METHOD

C.W. Gear (Gear, 1971a,b,c; 1972) has created a subroutine called DIFSUB for solving ordinary differential equations. It uses a variable-order, variable-step, linear, predictor-corrector algorithm.

There are options for nonstiff and stiff equations.

A.1 Nonstiff Option

The nonstiff option uses an Adams-Bashforth predictor and an Adams-Moulton corrector:

PREDICTOR EQN: (Adams-Bashforth)
$$y_{n+1} = y_n + h \sum_{i=1}^{k} \beta_i \dot{y}_{n+1-i} \qquad (A.1)$$

CORRECTOR EQN: (Adams-Moulton)
$$y_{n+1} = y_n + h \sum_{i=0}^{k} \beta_i^* \dot{y}_{n+1-i} \qquad (A.2)$$

The coefficients β_i and β_i^* may be found in Henrici (1962). The order may vary from 1-7.

A.2 Stiff Option

The stiff option uses a predictor and corrector based on the backward or numerical differentiation methods (Henrici, 1962):

PREDICTOR EQN.: $$y_{n+1} = h\,\eta_1 y_n + \sum_{i=1}^{k} \alpha_i y_{n+1-i} \qquad (A.3)$$

CORRECTOR EQN.: $$y_{n+1} = h\,\eta_o^* \dot{y}_{n+1} + \sum_{i=1}^{k} \alpha_i^* y_{n+1-i} \qquad (A.4)$$

The corrector coefficient values are given in Gear (1971c). From 1st to 6th order is available.

The corrector equation has the property of stiff stability (Gear, 1969a; Figure A.1) for up to 6th order; i.e., it is accurate and stable near the origin and stable to the far left of the origin in the $h\lambda$ plane.

Accuracy is not important far to the left of the origin, since the components here are very small, but absolute stability is required. Around the origin, accuracy is essential for which we need relative or absolute stability. Gear's method is relatively stable in the rectangle shown. Elsewhere there are no requirements, since to the right of $\mathrm{Re}(h\lambda) = \alpha$ and above and below $\mathrm{Im}(h\lambda) = \pm\theta$, the function is varying too rapidly to be represented by points spaced a distance h, but is not decaying so fast that we can ignore such terms. For order 1-6, $D > -6.1$, $\theta < 0.5$ and $\alpha \approx 0$ (Gear 1969a).

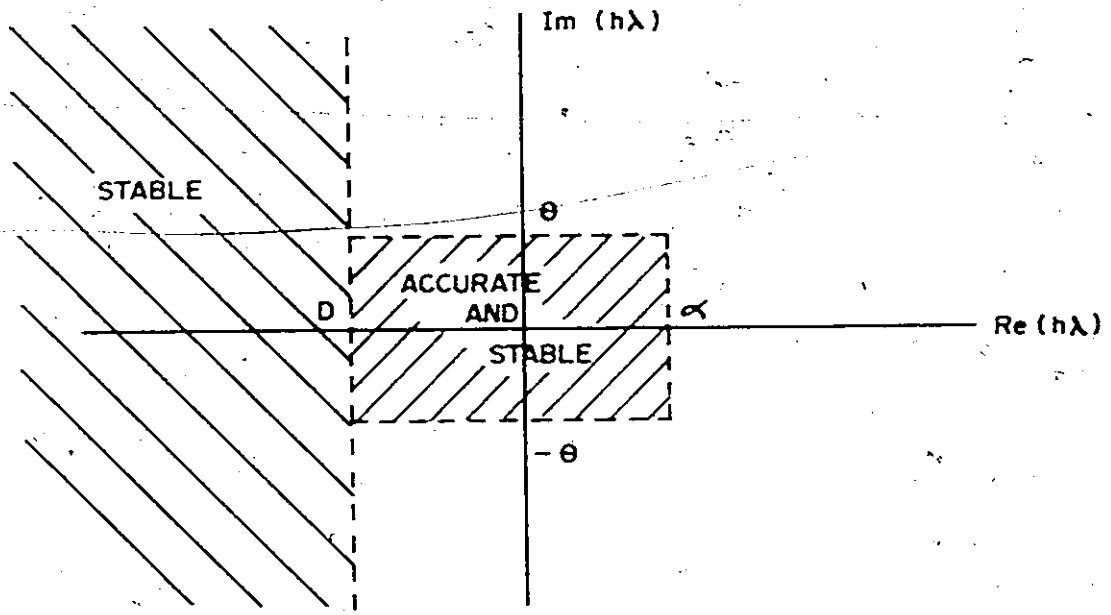


FIGURE A.1 : STIFF STABILITY

The Jacobian matrix is needed for the stiff option, as up to three Newton-Raphson iterations of the corrector are performed. The Jacobian may be user-supplied or evaluated numerically. It is re-evaluated only when there is a change in order or the corrector failed to converge.

Much of the algorithm is the same for the nonstiff and stiff option except that different coefficients are used.

A.3 General Features

Rather than storing back values of the independent variable and the first derivatives, the algorithm uses a variation of the Nordsieck vector (Nordsieck, 1962):

$$\underset{\sim}{a}_n = [y_n, h\dot{y}_n, \frac{h^2\ddot{y}_n}{2!}, \cdots, \frac{h^k y_n^{(k)}}{k!}] \qquad (A.5)$$

Gear expresses the predictor-corrector in the form:

PREDICTOR EQN.: $$\underset{\sim}{a}_{n+1,(0)} = \underset{\approx}{A}\, \underset{\sim}{a}_n \qquad (A.6)$$

CORRECTOR EQN.: $$\underset{\sim}{a}_{n+1,(m+1)} = \underset{\sim}{a}_{n+1,(m)} + l\, \underset{\approx}{M}\, \underset{\sim}{F}(\underset{\sim}{a}_{n+1,(m)}) \qquad (A.7)$$

where $\underset{\approx}{A}$ is the Pascal triangle matrix:

$$A_{ij} = \binom{j}{i} \qquad (A.8)$$

$$= 0 \text{ if } i > j$$

$\underset{\sim}{E}(a)$ is the amount by which the differential equation is not satisfied locally by a:

$$\underset{\sim}{E}(a) = h\,\underset{\sim}{f}(t,a) - \underset{\sim}{a}_1 = h\,\underset{\sim}{f}(t,y) - h\,\underset{\sim}{\dot{y}} \qquad (A.9)$$

For direct iteration as in the nonstiff option

$$\underset{\approx}{M} = \underset{\approx}{I} \quad \text{(identity matrix)} \qquad (A.10)$$

For Newton-Raphson iteration as in the stiff option

$$\underset{\approx}{M} = \left[\underset{\approx}{I} - h\,\ell_0 \frac{\partial \underset{\sim}{f}}{\partial \underset{\sim}{y}} \right] \qquad (A.11)$$

where $\frac{\partial \underset{\sim}{f}}{\partial \underset{\sim}{y}}$ is the Jacobian matrix

The coefficients $\underset{\sim}{\ell}$ are given in Gear (1971c). $\underset{\sim}{\ell}$ is chosen to achieve stability and accuracy.

The same order is used on the predictor and corrector steps. The local truncation error is given by $C_{k+1} h^{k+1} y^{(k+1)} + O(h^{k+2})$ for a kth order step. The C_{k+1} values are given in Gear (1971c). The last component of the Nordsieck vector is $h^k y_n^{(k)}/k!$ and under suitable smoothness conditions, its backward difference yields an estimate of $h^{k+1} y^{(k+1)}/k!$. The error is controlled by keeping the Euclidean norm of the relative local truncation errors below a specified value.

The method is self-starting, using the first order formula to begin the integration. The order is variable

and is chosen to maximize the step size with some heuristic controls. The algorithm begins with first order and increases quickly to about fourth order where it remains for the bulk of the integration. Then as steady state is approached, the order decreases back to first order.

The nomenclature was first introduced in Gear (1967). Ratliff (1968) does some computations with the method. Gear (1971d) has extended his program to handle large sparse sets of simultaneous differential and algebraic equations; however, the program is in machine language.

Several workers have recommended Gear's method for stiff systems. Nikolai (1973) tells of his experience with it. Walters (1972) provides a detailed discussion of the method. A version of DIFSUB is available in the IMSL library (IMSL, 1973).

APPENDIX B: IMP

D.M. Brandon (Brandon, 1972, 1973, 1974a), now with Control Data Corporation, has developed IMP (Implicit Solution Software System). IMP is a software system for the direct or iterative solution of large differential and/or algebraic systems. The package is equation-oriented and contains the author's own A-stable numerical integration technique and sparse matrix routines for solving the corrector equation. IMP is offered as a standard CDC software product through their Application Services Division. For non-CDC computers, the package is available from Brandon himself.

The IMP tests in this thesis were done with an object copy of version 1.1 compiled under FTN version 4.0, optimization level 2, Scope 3.4 operating system.

The IMP user must write his own user subroutine defining the system to be simulated and choosing from the many options available. The problem must be expressed in the form:

$$\underset{\sim}{\dot{y}} = \underset{\approx}{A}\,\underset{\sim}{y} + \underset{\sim}{B} \qquad (B.0)$$

where A is the Jacobian matrix

and B is the augmented constant vector.

Both A and B may be functions of time or y. The left hand side is set to zero for algebraic equations. Partial differential equations are reduced to ordinary differential equations by discretization of one or more dependent variables.

B.1 Brandon's Integration Technique

Brandon's integration procedure (Brandon, 1974f) is a single-step method:

$$y_{n+1} = y_n + h[(1 - \omega)\dot{y}_{n+1} + \omega\dot{y}_n] \qquad (B.1)$$

For the equation:

$$\dot{y} = \lambda y \qquad (B.2)$$

$$y_{n+1} = y_n e^{h\lambda} \qquad (B.3)$$

Substitute (B.2) into (B.1)

$$y_{n+1} = y_n + h[(1 - \omega)\lambda y_{n+1} + \omega\lambda y_n] \qquad (B.4)$$

$$y_{n+1}[1 - h\lambda(1 - \omega)] = y_n[1 + h\lambda\omega] \qquad (B.5)$$

$$\frac{y_{n+1}}{y_n} = \frac{1 + h\lambda w}{1 - h\lambda(1 - w)} = e^{h\lambda} \qquad (B.6)$$

Solving for w:

$$w = -\frac{1}{h\lambda} - \frac{1}{e^{-h\lambda} - 1} \qquad (B.7)$$

Liniger and Willoughby (1970) introduced the concept of exponential fitting in that w can be chosen to make the integration exact for a given value of λ.

For more than one equation, different values of w may be used:

$$w_i = -\frac{1}{z_i} - \frac{1}{e^{-z_i} - 1} \qquad (B.8)$$

There will not be an eigenvalue corresponding to each equation except for the system:

$$\dot{y}_i = \lambda y_i \qquad i=1,2,\ldots,N \qquad (B.9)$$

Brandon chooses z to approximate the local gradient:

$$z_i = \frac{\partial \dot{y}_i}{\partial y_i} = h \sum_{j=1}^{N} a_{ij} \left(\frac{\dot{y}_j}{\dot{y}_i} \right) \qquad (B.10)$$

For the single equation $\dot{y} = \lambda y$, this reduces to exponential fitting. Thus Brandon has extended the concept of exponential fitting to systems of equations.

Note that no predictor equation is used. The first value for the corrector iteration is taken as y_n.

The method is A-stable for $0 \leq w \leq 0.5$.

There is a very conservative local truncation error estimate. Expanding y_{n+1} in a Taylor Series:

$$y_{n+1} = y_n + h\dot{y}_n + \frac{h^2}{2!}\ddot{y}_n + \frac{h^3}{3!}\dddot{y}_n + \ldots \qquad (B.11)$$

$$w\dot{y}_{n+1} = wy_n + hw\dot{y}_n + \frac{h^2}{2!}w\ddot{y}_n + \frac{h^3}{3!}w\dddot{y}_n + \ldots \qquad (B.12)$$

$$(1-w)y_{n+1} = (1-w)y_n + h(1-w)\dot{y}_n + \frac{h^2}{2!}(1-w)\ddot{y}_n + \frac{h^3}{3!}(1-w)\dddot{y}_n + \ldots \qquad (B.13)$$

Adding (B.12) and (B.13)

$$y_{n+1} = y_n + hw\dot{y}_n + h(1-w)\dot{y}_{n+1} - h(1-w)\dot{y}_{n+1} + h(1-w)\dot{y}_n + \frac{h^2}{2!}\ddot{y}_n + \frac{h^3}{3!}\dddot{y}_n + \ldots \qquad (B.14)$$

The error term is:

$$E = -h(1-w)\dot{y}_{n+1} + h(1-w)\dot{y}_n + \frac{h^2}{2!}\ddot{y}_n + \frac{h^3}{3!}\dddot{y}_n + \ldots \qquad (B.15)$$

Expanding $\dot{y}_{n+1}$ in a Taylor Series:

$$\dot{y}_{n+1} = \dot{y}_n + h\,\ddot{y}_n + \frac{h^2}{2!}\dddot{y}_n + \ldots \qquad (B.16)$$

Substitute (B.16) into (B.15)

$$E = h^2\left(\omega - \frac{1}{2}\right)\ddot{y}_n + h^3\left(\frac{\omega}{2} - \frac{1}{3}\right)\dddot{y}_n \qquad (B.17)$$

Replace $\dddot{y}_n$ by $\dfrac{\ddot{y}_{n+1} - \ddot{y}_n}{h}$

$$E = \frac{h^2}{12}\left[(6\omega - 2)\ddot{y}_n + (6\omega - 4)\ddot{y}_{n+1}\right] \qquad (B.18)$$

Brandon adjusts this term empirically to:

$$E = \left|\frac{h^2 E_R}{6(y_n + y_{n+1})}\right|\left(\frac{1}{Y_i}\right) + \left|\frac{h(\dot{y}_{n+1} - \dot{y}_n)}{15(y_n + y_{n+1})}\right| \qquad (B.19)$$

where

$$E_R = \left[(1 + d_i)\ddot{y}_{n+1} - (1 - d_i)\ddot{y}_n\right] \qquad (B.20)$$

$$\ddot{y}_i = \sum_{j=1}^{N} a_{ij}\, y_j \qquad (B.21)$$

$$d_i = 3\left(1 - \frac{2\omega_i}{h}\right) \qquad (B.22)$$

$$Y_i = 45\,\frac{|a_{ii}|}{\sum_{j=1}^{N} |a_{ij}|} + 5 \qquad (B.23)$$

b is the augmented constant vector

z is restricted as:

$$-170 \leq z_i \leq -0.015 \qquad (B.24)$$

This is the error term for IMP version 1.1; however, Brandon has used in later versions a new error option (Brandon, 1974d). This consists of taking a double step and comparing the solution with two single steps. This approach has been used with Runge-Kutta methods (Carnahan, Luther and Wilkes, 1969).

B.2 Sparse Linear Equation Solution

Version 1.1 offers many options for solving sparse linear equations. These can occur in solving the corrector or the user can solve his own set of linear equations.

There are two iterative options: Gauss-Seidel double sweep and gradient iteration. Direct elimination using the Crout approach is available. In elimination, the user has options for ordering the system matrix, handling the fill-in and pivoting. Ordering is important, since the order in which the variables are eliminated affects the number of nonzero elements generated (fill-in). However, in normal applications problems, the natural order of the equations would be such that the bandwidth would be minimized.

The storage scheme used is cumulative indexing (section 5.1.3) and quasilinearization (section 4.5) is used in the corrector solution.

Future versions of IMP will include the following additions (Brandon 1974b):

(1) the conjugate gradient method

(2) special elimination routines for banded matrices

(3) special Gauss-Seidel routine for banded matrices

(4) special elimination routine for symmetric and banded symmetric matrices

(5) special conjugate gradient method for symmetric matrices

(6) full matrix elimination for smaller full matrix problems.

APPENDIX C: SHANNON'S METHOD

Shannon (1971) has developed a technique for the numerical integration of stiff, sensitive and multivalued equations.

For a single equation, the method is very simple.

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

If $f(t,y)$ is less than 1, t is the dependent variable or the variable of integration. However, if $f(t,y)$ is or becomes greater than one, y is made the variable of integration.

i.e.

$$\frac{dt}{dy} = \frac{1}{f(t,y)} \qquad (C.2)$$

This may be extended to systems of equations as follows.

For the system:

$$\frac{dy_1}{dy_n} = f_1(y_1,\ldots,y_n)$$

$$\frac{dy_2}{dy_n} = f_2(y_1,\ldots,y_n) \qquad (C.3)$$

$$\vdots \qquad \vdots$$

$$\frac{dy_n}{dy_n} = 1$$

where y_n is time, the normal variable of integration, if we form the matrix of all possible derivatives,

$$\underset{\sim}{D} = \begin{bmatrix} \frac{dy_1}{dy_1} & \frac{dy_1}{dy_2} & \cdots & \frac{dy_1}{dy_n} \\ \frac{dy_2}{dy_1} & \frac{dy_2}{dy_2} & \cdots & \frac{dy_2}{dy_n} \\ \vdots & \vdots & & \vdots \\ \frac{dy_n}{dy_1} & \frac{dy_n}{dy_2} & \cdots & \frac{dy_n}{dy_n} \end{bmatrix} \qquad (C.4)$$

It can be shown that there is always at least one column of $\underset{\sim}{D}$ where the values of all derivatives are less than or equal to one in absolute value. If this is the kth column, then y_k should be taken as the variable of integration. If the integration proceeds with y_k as the variable of integration and the largest element dy_i/dy_k becomes greater than 1, then y_i will become the independent variable. The derivatives in column i will then be less than or equal to 1.

Thus the variable of integration is changed as the integration proceeds, so that all the derivatives with respect to that variable are less than or equal to 1. Any method of integration may be used.

Let us try this method on SYSTEM I (Appendix D).

the original system is:

$$\frac{dy_1}{dt} = -500.5y_1 + 499.5y_2 \qquad y_1(0) = 0$$

$$\frac{dy_2}{dt} = 499.5y_1 - 500.5y_2 \qquad y_2(0) = 2 \qquad (C.5)$$

with analytical solution:

$$y_1 = e^{-t} - e^{-1000t}$$

$$y_2 = e^{-t} + e^{-1000t} \qquad (C.6)$$

making time y_3, the system becomes:

$$\frac{dy_1}{dy_3} = -500.5y_1 + 499.5y_2 \qquad y_1(0) = 0$$

$$\frac{dy_2}{dy_3} = 499.5y_1 - 500.5y_2 \qquad y_2(0) = 2 \qquad (C.7)$$

$$\frac{dy_3}{dy_3} = 1 \qquad y_3(0) = 0$$

the matrix of derivatives is:

$$\underset{\approx}{D} = \begin{bmatrix} \frac{dy_1}{dy_1} & \frac{dy_1}{dy_2} & \frac{dy_1}{dy_3} \\ \frac{dy_2}{dy_1} & \frac{dy_2}{dy_2} & \frac{dy_2}{dy_3} \\ \frac{dy_3}{dy_1} & \frac{dy_3}{dy_2} & \frac{dy_3}{dy_3} \end{bmatrix} \qquad (C.8)$$

$$= \begin{bmatrix} 1 & \frac{-500.\delta y_1 + 499.\delta y_2}{499.\delta y_1 - 500.\delta y_2} & -500.\delta y_1 + 499.\delta y_2 \\ \frac{499.\delta y_1 - 500.\delta y_2}{-500.\delta y_1 + 499.\delta y_2} & 1 & 499.\delta y_1 - 500.\delta y_2 \\ \frac{1}{-500.\delta y_1 + 499.\delta y_2} & \frac{1}{499.\delta y_1 - 500.\delta y_2} & 1 \end{bmatrix} \quad (3.9)$$

With y_3 as the variable of integration, the Jacobian matrix is:

$$\underset{\sim}{J}(y_3) = \begin{bmatrix} -500.\delta & 499.\delta & 1 \\ 499.\delta & -500.\delta & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad (3.10)$$

The eigenvalues are -1, -1000 and 0.

At time 0, $y_1 = 0$, $y_2 = 0$

$$\underset{\sim}{J}(0) = \begin{bmatrix} 1 & -\frac{999}{1001} & 999 \\ -\frac{1001}{999} & 1 & -1001 \\ \frac{1}{999} & -\frac{1}{1001} & 1 \end{bmatrix} \quad (3.11)$$

Shannon's method say to make y_2 the independent variable.

If we make y_2 the independent variable

$$\frac{dy_1}{dy_2} = \frac{-500.5y_1 + 499.5y_2}{499.5y_1 - 500.5y_2}$$

$$\frac{dy_2}{dy_2} = 1 \qquad (C.12)$$

$$\frac{dy_3}{dy_2} = \frac{1}{499.5y_1 - 500.5y_2}$$

The Jacobian matrix is:

$$\underline{J}(y_2) = \begin{bmatrix} \frac{1000y_2}{(499.5y_1 - 500.5y_2)^2} & \frac{-1000y_1}{(499.5y_1-500.5y_2)^2} & 0 \\ 0 & 0 & 0 \\ \frac{-499.5}{(499.5y_1-500.5y_2)^2} & \frac{500.5}{(499.5y_1-500.5y_2)^2} & 0 \end{bmatrix} \qquad (C.13)$$

The eigenvalues are $\frac{1000y_2}{(499.5y_1-500.5y_2)^2}$, 0, 0.

at time 0: $y_1=0$, $y_2=2$ $\quad \lambda = \frac{2000}{(1001)^2} \approx 2 \times 10^{-3}$, 0, 0 $\qquad (C.14)$

as $t \to 0.01$ $\quad y_1 \to y_2 = y \to 0$ $\quad \lambda = \frac{1000}{y} \to \infty$, 0, 0 $\qquad (C.15)$

To see what happens when y_1 is the variable of integration

$$\frac{dy_1}{dy_1} = 1$$

$$\frac{dy_2}{dy_1} = \frac{499.5y_1 - 500.5y_2}{-500.5y_1 + 499.5y_2} \qquad (C.16)$$

$$\frac{dy_3}{dy_1} = \frac{1}{-500.5y_1 + 499.5y_2}$$

The Jacobian matrix $J(y_1)$ is:

$$J(y_1) = \begin{bmatrix} 0 & 0 & 0 \\ \frac{-1000y_2}{(-500.5y_1+499.5y_2)^2} & \frac{1000y_1}{(-500.5y_1+499.5y_2)^2} & 0 \\ \frac{500.5}{(-500.5y_1+499.5y_2)^2} & \frac{-499.5}{(-500.5y_1+499.5y_2)^2} & 0 \end{bmatrix} \qquad (C.17)$$

The eigenvalues are $\frac{1000y_1}{(-500.5y_1+499.5y_2)^2}$, 0, 0.

At time 0: $y_1=0$, $y_2=2$ $\quad \lambda = 0, 0, 0$ $\qquad (C.18)$

as $t \to 0.01$ $\quad y_1 \to y_2 = y \to 0$ $\quad \lambda = \frac{1000}{y} \to \infty, 0, 0$ $\qquad (C.19)$

If y_2 is made the independent variable, the eigenvalues of the Jacobian at time 0 are 0, 0, 2×10^{-3}. However a very short time later (~0.01), the largest eigenvalue approaches infinity.

Similarly if y_1 is made the variable of integration, the largest eigenvalue also approaches infinity.

As $t \to 0.01$, $e^{-1000t} \to 0$, and $y_1' \to y_2' = y_3' \to 0$.

The matrix of derivatives becomes:

$$\underset{\approx}{D} \quad \begin{bmatrix} 1 & 1 & -y \\ 1 & 1 & -y \\ -\frac{1}{y} & -\frac{1}{y} & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ -\infty & -\infty & 1 \end{bmatrix} \qquad (0.99)$$

As t approaches 0.01 approximately, y_3 (time) should be made the variable of integration.

Thus for a very short period of time, the change of independent variable from y_3 to y_2 would speed up the integration; however, after that period we are back where we started with y_3 (time), again the independent variable.

Thus Shannon's method seems to be of limited effectiveness for handling stiff systems.

APPENDIX D: TEST EXAMPLES

Seven stiff examples were tested:

SYSTEM I: Moderately Stiff Linear Real Eigenvalue Example

II: Very Stiff Linear Real Eigenvalue Example

III: Linear Complex Eigenvalue Example

IV: Krogh's Example

V: Chemistry Example

VI: Polymer Example

VII: Stable Polymer Example

and four nonstiff examples:

SYSTEM VIII: Linear Nonstiff Complex Eigenvalue Example

IX: Krogh's Nonstiff Example

X: Nonlinear Reaction Example

XI: Gas Absorber Example

D.1 SYSTEM I: Moderately Stiff Linear Real Eigenvalue Example

We use the 2-dimensional linear system:

$$\dot{\underline{y}} = \underline{\underline{A}}\,\underline{y} = \begin{bmatrix} -a & b \\ b & -a \end{bmatrix} \underline{y} \qquad \underline{y}(0) = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \tag{D.1.1}$$

with analytical solution:

$$\begin{aligned} y_1 &= e^{-(a-b)t} - e^{-(a+b)t} \\ y_2 &= e^{-(a-b)t} + e^{-(a+b)t} \end{aligned} \tag{D.1.2}$$

We choose $a = 500.5$, $b = 499.5$ to yield the moderately stiff example:

$$\begin{aligned} \dot{y}_1 &= -500.5\, y_1 + 499.5\, y_2 \qquad y_1(0) = 0 \\ \dot{y}_2 &= 499.5\, y_1 - 500.5\, y_2 \qquad y_2(0) = 2 \end{aligned} \tag{D.1.3}$$

with analytical solution:

$$\begin{aligned} y_1 &= e^{-t} - e^{-1000t} \\ y_2 &= e^{-t} + e^{-1000t} \end{aligned} \tag{D.1.4}$$

The range of integration is [0,1]

The eigenvalues are -1 and -1000.

D.2 SYSTEM II: Very Stiff Linear Real Eigenvalue Example

As in SYSTEM I we choose $a = 500000.5$, $b = 499999.5$ to yield the very stiff example:

$$\begin{aligned} \dot{y}_1 &= -500000.5y_1 + 499999.5y_2 \quad y_1(0) = 0 \\ \dot{y}_2 &= 499999.5y_1 - 500000.5y_2 \quad y_2(0) = 2 \end{aligned} \tag{D.2.1}$$

with analytical solution:

$$\begin{aligned} y_1 &= e^{-t} - e^{-1000000t} \\ y_2 &= e^{-t} + e^{-1000000t} \end{aligned} \tag{D.2.2}$$

The range of integration is [0,1]

The eigenvalues are -1 and -1000000.

The stiffness ratio of 10^6 is as large as has been found in practice.

D.3 SYSTEM III: Linear Complex Eigenvalue Example

The system:

$$\begin{aligned} \dot{y}_1 &= -Ay_1 + By_2 + (A - B - 1)e^{-t} \quad y_1(0)=2 \\ \dot{y}_2 &= -By_2 - Ay_2 + (A + B - 1)e^{-t} \quad y_2(0)=2 \\ \dot{y}_3 &= -Cy_3 + Dy_4 + (C - D - 1)e^{-t} \quad y_3(0)=2 \\ \dot{y}_4 &= -Dy_3 - Cy_4 + (C + D - 1)e^{-t} \quad y_4(0)=2 \end{aligned} \tag{D.3.1}$$

is used with analytical solution:

$$\begin{aligned} y_1 &= e^{-At}(\cos Bt + \sin Bt) + e^{-t} \\ y_2 &= e^{-At}(\cos Bt - \sin Bt) + e^{-t} \\ y_3 &= e^{-Ct}(\cos Dt + \sin Dt) + e^{-t} \\ y_4 &= e^{-Ct}(\cos Dt - \sin Dt) + e^{-t} \end{aligned} \qquad (D.3.2)$$

We choose $A = 1000$, $B = C = D = 1$ to yield the moderately stiff example:

$$\begin{aligned} \dot{y}_1 &= -1000y_1 + y_2 + 999e^{-t} & y_1(0) &= 2 \\ \dot{y}_2 &= -y_1 - 1000y_2 + 1000e^{-t} & y_2(0) &= 2 \\ \dot{y}_3 &= -y_3 + y_4 - e^{-t} & y_3(0) &= 2 \\ \dot{y}_4 &= -y_3 - y_4 + e^{-t} & y_4(0) &= 2 \end{aligned} \qquad (D.3.3)$$

with analytical solution:

$$\begin{aligned} y_1 &= e^{-1000t}(\cos t + \sin t) + e^{-t} \\ y_2 &= e^{-1000t}(\cos t - \sin t) + e^{-t} \\ y_3 &= e^{-t}(\cos t + \sin t) + e^{-t} \\ y_4 &= e^{-t}(\cos t - \sin t) + e^{-t} \end{aligned} \qquad (D.3.4)$$

The range of integration is [0,1]

The eigenvalues are $-1000 \pm i$, $-1 \pm i$.

D.4 SYSTEM IV: Krogh's Example

A nonlinear example suggested by Krogh was taken from Gear (1971c). The system:

$$\dot{z}_i = -\beta_i z_i + z_i^2 \qquad i=1,2,3,4 \quad z_i(0) = -1 \tag{D.4.1}$$

has analytical solution:

$$z_i = \frac{\beta_i}{1 + c_i e^{\beta_i t}} \qquad c_i = -(1 + \beta_i) \tag{D.4.2}$$

using the transformation $\underset{\sim}{y} = \underset{\approx}{u}\ \underset{\sim}{z}$ (D.4.3)

where

$$\underset{\approx}{u} = 1/2 \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix} \tag{D.4.4}$$

note that $\underset{\sim}{z} = \underset{\approx}{u}\ \underset{\sim}{y}$ since $\underset{\approx}{u} = \underset{\approx}{u}^{-1}$

we obtain the system:

$$\dot{\underset{\sim}{y}} = -\underset{\approx}{u}\ \underset{\approx}{\beta}\ \underset{\approx}{u}\ \underset{\sim}{y} + \underset{\approx}{u}\ \underset{\sim}{z}^2 \tag{D.4.5}$$

where

$$\underset{\approx}{\beta} = \begin{bmatrix} \beta_1 & & & \\ & \beta_2 & & \\ & & \beta_3 & \\ & & & \beta_4 \end{bmatrix} \tag{D.4.6}$$

Using $\beta_1 = 1000$, $\beta_2 = 800$, $\beta_3 = -10$ and $\beta_4 = 0.001$ we obtain:

$$\dot{\underline{y}} = \frac{1}{4}\begin{bmatrix} -1790.001 & 1809.999 & 189.999 & 210.001 \\ 1809.999 & -1790.001 & -210.001 & -189.999 \\ 189.999 & -210.001 & -1790.001 & -1809.999 \\ 210.001 & -189.999 & -1809.999 & -1790.001 \end{bmatrix} \underline{y}$$

$$+ \frac{1}{4}\begin{bmatrix} y_1 + y_2 + y_3 + y_4 + 2y_1y_2 + 2y_1y_3 + 2y_1y_4 - 2y_2y_3 - 2y_2y_4 - 2y_3y_4 \\ " \quad " \quad " \quad " + 2 " - 2 " - 2 " + 2 " + 2 " - 2 " \\ " \quad " \quad " \quad " - 2 " + 2 " - 2 " + 2 " - 2 " + 2 " \\ " \quad " \quad " \quad " - 2 " - 2 " + 2 " - 2 " + 2 " + 2 " \end{bmatrix} \tag{D.4.7}$$

$$\underline{y}(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

The range of integration is [0,5]

The eigenvalues are time-variant. Initially they are $2x_i - \beta_i$, i.e., -1002, -802, 8 and -2.001. As $t \to \infty$ the eigenvalues approach β_i (if $\beta_i \leq 0$) and 0 (if $\beta_i > 0$).

D.5 SYSTEM V: Chemistry Example

An example which arose in a chemistry problem is taken from Gear (1969a):

$$\begin{aligned}
\dot{y}_1 &= -0.013\, y_1 - 1000\, y_1 y_3 & y_1(0) &= 1 \\
\dot{y}_2 &= -2500\, y_2 y_3 & y_2(0) &= 1 \qquad (D.5.1) \\
\dot{y}_3 &= -0.013\, y_1 - 1000 y_1 y_3 & & \\
&\quad -2500\, y_2 y_3 & y_3(0) &= 0
\end{aligned}$$

The range of integration is [0,50]

The eigenvalues are real and initially are about -3500, 0.017 and 0.

D.6 SYSTEM VI: Polymer Example

A large stiff system of 33 equations representing a polymerization reaction was taken from Hull *et al.* (1972b):

$$\begin{aligned}
\dot{y}_1 &= k_1 y_{12}^2 - k_2 y_1 y_{23} + \frac{2D}{h^2}\left[y_2 - (1+\frac{0.01h}{D})y_1\right] \\
\dot{y}_i &= k_1 y_{11+i}^2 - k_2 y_i y_{22+i} + \frac{D}{h^2}(y_{i-1} - 2y_i + y_{i+1}) \quad i=2,3,\ldots,10 \\
\dot{y}_{11} &= k_1 y_{22}^2 - k_2 y_{11} y_{33} + \frac{2D}{h^2}(y_{10} - y_{11}) \qquad (D.6.1) \\
\dot{y}_i &= 2k_1 y_i^2 + 2k_2 y_{i-11} y_{i+11} + k_3 y_{i+11} \quad i=12,13,\ldots,22 \\
\dot{y}_i &= 2k_1 y_{i-11}^2 - 2k_3 y_{i-22} y_i - 2k_3 y_i \quad i=23,24,\ldots,33
\end{aligned}$$

where

$$k_1 = 0.70$$
$$k_2 = 1.35$$
$$k_3 = 3 \times 10^{-6} \qquad (D.6.2)$$
$$D = 1.15 \times 10^{-4}$$
$$h = 0.00127$$

initial conditions $y_i(0) = 0 \quad i = 1,\ldots,11$

$= 9.0 \quad i = 12,13,\ldots,22$

$= 0.0 \quad i = 23,24,\ldots,33$

The range of integration was [0,100]

This example represents a physically unstable system in that some of the variables (y_i, $i = 23,24,\ldots,33$) become larger and larger as integration proceeds. Mathematically this means that the eigenvalues corresponding to these variables are positive.

D.7 SYSTEM VII: Stable Polymer Example

The polymer example in 2.2.5 is an unstable system. If k_4 is changed to k_1 in equations for $i = 12$-22, the system becomes a stable fictitious one.

D.8 SYSTEM VIII: Linear Nonstiff Complex Eigenvalue Example

The system:

$$\begin{aligned}
\dot{y}_1 &= -Ay_1 + By_2 && y_1(0) = 1 \\
\dot{y}_2 &= -By_1 - Ay_2 && y_2(0) = 1 \\
\dot{y}_3 &= -Cy_3 + Dy_4 && y_3(0) = 1 \\
\dot{y}_4 &= -Dy_3 - Cy_4 && y_4(0) = 1
\end{aligned} \tag{D.8.1}$$

is used with analytical solution:

$$\begin{aligned}
y_1 &= e^{-At}(\cos Bt + \sin Bt) \\
y_2 &= e^{-At}(\cos Bt - \sin Bt) \\
y_3 &= e^{-Ct}(\cos Dt + \sin Dt) \\
y_4 &= e^{-Ct}(\cos Dt - \sin Dt)
\end{aligned} \tag{D.8.2}$$

We choose $A = 0.5$, $B = 0.25$, $C = 0.25$, $D = 0.5$ to yield:

$$\begin{aligned}
\dot{y}_1 &= -0.5y_1 + 0.25y_2 && y_1(0)=1 \\
\dot{y}_2 &= -0.25y_1 - 0.5y_2 && y_2(0)=1 \\
\dot{y}_3 &= -0.25y_3 + 0.5y_4 && y_3(0)=1 \\
\dot{y}_4 &= -0.5y_3 - 0.25y_4 && y_4(0)=1
\end{aligned} \tag{D.8.3}$$

with analytical solution:

$$y_1 = e^{-0.5t}(\cos 0.25t + \sin 0.25t)$$

$$y_2 = e^{-0.5t}(\cos 0.25t - \sin 0.25t)$$

$$y_3 = e^{-0.25t}(\cos 0.5t + \sin 0.5t) \qquad (D.8.4)$$

$$y_4 = e^{-0.25t}(\cos 0.5t - \sin 0.5t)$$

The range of integration is [0,5]

The eigenvalues are $-0.5 \pm 0.25i$, $-0.25 \pm 0.5i$.

D.9 SYSTEM IX: Krogh's Nonstiff Example

The equations of SYSTEM IV: Krogh's example are used but with $\beta_1 = 0.2$, $\beta_2 = 0.2$, $\beta_3 = 0.3$, $\beta_4 = 0.4$ to obtain:

$$\underline{y} = \frac{1}{4} \begin{bmatrix} -1.0 & -0.4 & -0.2 & 0.0 \\ -0.4 & -1.0 & 0.0 & 0.2 \\ -0.2 & 0.0 & -1.0 & 0.4 \\ 0.0 & 0.2 & 0.4 & -1.0 \end{bmatrix} \underline{y}$$

$$+ \frac{1}{4} \begin{bmatrix} y_1^2 + y_2^2 + y_3^2 + y_4^2 + 2y_1y_2 + 2y_1y_3 + 2y_1y_4 - 2y_2y_3 - 2y_2y_4 - 2y_3y_4 \\ " \quad " \quad " \quad " + 2 " - 2 " - 2 " + 2 " + 2 " - 2 " \\ " \quad " \quad " \quad " - 2 " + 2 " - 2 " + 2 " - 2 " + 2 " \\ " \quad " \quad " \quad " - 2 " - 2 " + 2 " - 2 " + 2 " + 2 " \end{bmatrix} \qquad (D.9.1)$$

$$y(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

The range of integration is [0,10]

Here the step size is constrained only by accuracy not stability. The initial eigenvalues are -2.1, -2.2, -2.3 and -2.4. As $x \to \infty$, all the eigenvalues approach zero.

D.10 SYSTEM X: Nonlinear Reaction Example

A system representing a nonlinear reaction was taken from Hull *et al.* (1972b):

$$\begin{aligned} \dot{y}_1 &= -y_1 & y_1(0) &= 1 \\ \dot{y}_2 &= y_1 - y_2^2 & y_2(0) &= 0 \\ \dot{y}_3 &= y_2^2 & y_3(0) &= 0 \end{aligned} \qquad (D.10.1)$$

The integration range was [0,10].

D.11 SYSTEM XI: Gas Absorber Example

A nonlinear system representing the dynamics of a gas absorber was taken from Lapidus and Seinfeld (1971):

$$\dot{y}_1 = \{-[40.8+66.7(M_1+0.08y_1)]y_1 + 66.7(M_2+0.08y_2)y_2\}/z_1+40.8v_1/z_1$$

$$\dot{y}_i = \{40.8y_{i-1}-[40.8+66.7(M_i+0.08y_i)]y_i + 66.7(M_{i+1}+0.08y_{i+1})y_{i+1}\}/z_i \quad i=2,3,4,5 \qquad (D.11.1)$$

$$\dot{y}_6 = \{40.8y_5-[40.8+66.7(M_6+0.08y_6)y_6\}/z_6 + 66.7(M_7+0.08v_2)v_2/z_6$$

where

$$v_1 = v_2 = 0$$

$$z_i = (M_i + 0.16y_i) + 75 \quad i=1,2,\ldots,6 \qquad (D.11.2)$$

with initial conditions:

$$y_0 = \begin{bmatrix} -0.03424992 \\ -0.06199031 \\ -0.08368619 \\ -0.10042889 \\ -0.11306320 \\ -0.12243691 \end{bmatrix} \quad M = \begin{bmatrix} 0.73576500 \\ 0.74675687 \\ 0.75929635 \\ 0.76774008 \\ 0.77446637 \\ 0.77971110 \\ 0.78363672 \end{bmatrix} \qquad (D.11.3)$$

The range of integration was [0,50].

APPENDIX E: COMPUTER TIMINGS

In Chapters 5 and 6, various numerical techniques for solving linear algebraic equations and ordinary differential equations were compared by their computer execution times for solving a set of test examples. Only the time to actually solve the problem was measured; time for input-output or problem set-up was not included.

All of the runs were made on the CDC Cyber 73 computer at the University of Western Ontario. The Fortran Extended (FTN), version 4.2 compiler was used with optimization level 1 and Scope 3.4.2 operating system. All of the IMP tests, however, used an object deck supplied by Control Data from FTN version 4.0, optimization level 2 and Scope 3.4 operating system. Some of the longer non-IMP runs were made with both optimization levels 1 and 2, but the difference in execution time was very slight (about 1%).

A rigorous comparison of numerical techniques for solving o.d.e.s should do more than just measure execution time. Hull *et al.* (1972a) also measure the number of derivative evaluations, the time required to evaluate derivatives and the remaining execution time called the overhead. Krogh (1973) discusses in depth, the aspects of

numerically testing a subroutine for the numerical solutions of o.d.e.s.

Execution time itself is not a perfect criteria for comparison as one program may be faster than another on one computer but slower with a different machine. This arises because of different relative computer times for the various arithmetic operations: addition, multiplication and division. For example on the CYBER 73 or CDC 6400, the times for multiplication and division are identical, whereas on other computers, division can be three or four times slower. For the CYBER 73, the ratio of clock pulses for addition, multiplication and division is 11:57:57 while it is 4:5:20 on the Cyber 76 (Brandon, 1974e). For the CYBER 73, this means that an algorithm requiring many divisions rather than multiplications will not appear any slower than another algorithm using few divisions and many multiplications. For example, the IMP algorithm uses fewer divisions than the TRGB algorithm, but this is not reflected in the CYBER 73 execution times. Ideally one should not measure execution time, but the numbers of the different arithmetic operations; these could then be converted into execution times for any desired computer. However, this is very inconvenient.

Another problem is the reproducibility of execution times particularly those below 0.5 seconds. Any program

taking less than 0.5 seconds was run several times to get an average. One program took 0.000, 0.019 and 0.038 on different runs. This corresponds to 0, 1 or 2 pulses of the clock almost on a random basis. If it was convenient, the portion of the program being timed was put in a DO loop and repeated several times to get a larger time which could then be divided by the number of repetitions.

If many other jobs are running in the computer at the same time, it is possible for a program to be taken from core and placed in a waiting queue by another job of higher priority. This may have a very slight effect on the execution time. Ideally the execution times should be measured with no other jobs in the system. This is not very practical; but as much as possible, the times were measured during periods of low usage.

APPENDIX F: THE DYNSYS DATA SET

The data required for a simulation is inputted to the DYNSYS program through the data set. This is described fully in the DYNSYS Manual (Bobrow, Johnson and Ponton, 1971). We will give here, a brief description of a data set using Example #1 in Chapter 8 as an illustration.

Figure 8.2 contains the dynamic information flow diagram for the example and the data set is at the end of Figure 8.4.

There are four basic sections in a data set: the simulation data, the equipment data, the stream data and the physical properties data. All data is stored in 12 column fields. The first 12 columns contain only alphanumeric data and the numeric data goes into 5 F12 fields starting in column 13.

F.1 Simulation Data

The simulation data begins with the data word "BEGIN". All cards before this are comments and are ignored by DYNSYS.

For example #1

IN/OUT	3.0	sets the maximum number of streams entering or leaving any module to three (default = 5)
HMAX	0.01	sets the maximum integration set size to 0.01 (default = 0.05)
TIME	3.0	sets the time of simulation to 3.0 (default = 10.0)
LIBRARY	1.0	means that one new module will be defined immediately after this card. Most common modules such as CONT1, STIR1 and VALV1 are already in a library of module names stored inside DYNSYS.
DELAY	9.0	DELAY is the new module being defined. It will be accessed as SUBROUTINE TYPE9

Other possible simulation data words are:

COMPS	X	sets the number of components to X (default = 1)
DELTAT	X	sets the initial integration step size to X (default = 0.00001)
HMIN	X	sets the minimum integration step size to X (default = 0.000001)
TOLERANCE	X	sets the error tolerance for the integration to X (default = 0.001)
ORDER	X	sets the maximum permissible order of integration to X (default = 6)
NONSTIFF		causes the nonstiff option to be used (default = stiff option)
MINPIVOT	X	sets the minimum permissible pivot value for TRGB-TRGB2 to X (default = 0.000001)
PRINTING	X	causes output to be printed every X time steps (default = 1)
LINEPLOT		causes output to be printed as a lineplot (in this case additional data must be given after physical properties data)
GRAPH		causes output to be printed as graph or graphs on CALCOMP plotter as well as Lineplot. LINEPLOT must also be used.

FUNCTION	X	causes X vectors of stream information to be read in as an input function. Stream data follows immediately.

The data word "PROCESS" signifies the beginning of the equipment data and thus the end of the simulation data.

F.2 Equipment Data

The data word "PROCESS" signifies the beginning of the equipment data. Here data for the various modules is written in their order of execution.

CONT1	1.0	the first module to be executed is CONT1 , referred to as module #1 in the dynamic information flow diagram (Figure 8.2)

After this, the entering and leaving stream numbers are given. A positive number indicates an input stream while a negative number indicates an output stream.

3.0	-4.0	stream 3.0 enters CONT1 and stream 4 leaves it. This can be verified in Figure 8.2.

The following card gives the equipment parameters for the module. This is the data which describes the individual units.

3. 1500. 1000. 2. 3.

For example, the set point for the controller is the third parameter 1000. Up to five parameters may be given. If more are required, the data word "EXTRA" X must follow and the X extra parameters appear on the following data cards. The data set in Figure 8.10 gives an example of this. We continue in this way for all modules. The data word "END" signifies the end of the equipment data.

F.3 Stream Data

The material flows between equipment are represented by material streams. Information about each stream is stored in the following vector.

position 1 - stream number
2 - flag
3 - total mass flow
4 - temperature
5 - pressure
6 - mass fraction of component 1
7 - mass fraction of component 2,
etc.

The stream flag serves the dual purpose of identifying the stream type (flags with absolute value < 10 are material flows, those > 10 are information flows) and, if negative, suppresses printing of the stream vector in the output.

The user may choose his own system of units as long as he is consistent throughout. Usually English units are employed.

STREAMS 5. means that initial information data will be given for five streams.

EXPLICIT means that the data will be given explicitly.

The following cards give the data. For the first stream

```
1.    1.    1000.    100.    14.7
1.
```

the stream number is 1. The stream is a material flow (the flag is less than 10). The flow rate, temperature and pressure are 1000., 100. and 14.7 respectively. There is only one component so its mass fraction is 1.0.

Data is given for five streams. The data word "END" signifies the end of the stream data.

F.4 Physical Properties Data

PROPERTIES -1. means that the physical properties of water are assumed.

The user may provide his own data if desired. See the DYNSYS Manual for details.

The data word "END" signifies the end of the physical properties data and in this case the end of the data set.

APPENDIX G: TIME DELAY MODULE

A module for approximating a fixed or variable time delay is available. Time delays may occur in pipelines between equipments or in instruments such as values and controllers.

Instrument lag is fixed whereas a pipeline delay is usually variable depending on the flow rate of the fluid passing through it. The variable delay is equal to the volume of the delaying apparatus divided by the flow rate through it.

The bucket brigade approach is used to simulate the stream delay. The past times and stream information are stored in vectors and the stream output from the delay is interpolated from these values. For a lengthy delay, the storage space is prohibitive, but later versions of the module may remedy this.

The delayed streams are stored in SX(I,J,K)
where I represents the vector of past times
J represents different delays
and K represents the value of the stream vector.

Not all of the stream is necessarily delayed; a fraction of it may be bypassed. Nor is the entire stream vector necessarily delayed; the user may choose the first variable to be delayed and only this variable and all others after it will be delayed. The delay may also simulate a pump by setting the inlet flow rate equal to the output flow rate.

There are five equipment parameters:

(1) TLAG - if the time delay is fixed, TLAG is the value of the delay. If the delay is variable, TLAG is the negative of the volume of the delaying apparatus or pipeline.

(2) BYP - fraction of stream not delayed, i.e., bypassed.

(3) NV - number of storage spaces used in the delay vector. An error message is given if NV is not large enough.

(4) N1 - first variable to be delayed
if N1 = 3, the entire stream is delayed
if N1 = 6, only the mass fractions are delayed.

(5) FLAG - if FLAG is greater than zero, the module simulates a pump; if FLAG = 0, the delay is normal.

A listing of the time delay module is given in Figure G.1.

FIGURE G.1: LISTING OF TIME DELAY MODULE

```
      SUBROUTINE TYPE9                                                    T09    1
C                                                                         T09    2
C     THIS MODULE REPRESENTS A FIXED OR VARIABLE TIME DELAY               T09    3
C                                                                         T09    4
C     EQUIPMENT PARAMETERS                                                T09    5
C     1 - TLAG - LENGTH OF TIME DELAY,IF TIME DELAY IS FIXED              T09    6
C              - NEGATIVE OF VOLUME OF DELAYING EQUIPMENT OR PIPELINE,    T09    7
C                IF TIME DELAY IS VARIABLE                                T09    8
C     2 - BYP  - FRACTION OF STREAM NOT DELAYED,I.E.,BYPASSED             T09    9
C     3 - NV   - NUMBER OF STORAGE SPACES USED IN DELAY VECTOR            T09   10
C     4 - N1   - FIRST VARIABLE TO BE DELAYED 3-TOTAL FLOW,4-TEMP.,ETC    T09   11
C     5 - FLAG - FLAG.GT.0:OUTPUT FLOW CONTROL (PUMP)                     T09   12
C                FLAG.EQ.0:NORMAL DELAY                                   T09   13
C                                                                         T09   14
      COMMON /MAT/ MP(4,5),EP(4,5),S(2,5,7),EX(1)                         T09   15
      COMMON /UNIT/ IM,NMP                                                T09   16
      COMMON /CON/ IG,NCOMP,NC4,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH       T09   17
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF      T09   18
      COMMON /LAG/ ND,NSW                                                 T09   19
      DIMENSION MC(10), SX(100,1,8)                                       T09   20
      INTEGER OUT                                                         T09   21
      DATA MC/10*1/                                                       T09   22
C                                                                         T09   23
C     SX IS THE DELAY MATRIX                                              T09   24
C        1ST VECTOR - DELAY AT DIFFERENT VALUES OF TIME                   T09   25
C        2ND VECTOR - DIFFERENT DELAYS                                    T09   26
C        3RD VECTOR - VALUES OF STREAM VECTOR                             T09   27
C                                                                         T09   28
      IN=MP(IM,3)                                                         T09   29
      OUT=-MP(IM,4)                                                       T09   30
C                                                                         T09   31
C     FIXED TIME DELAY                                                    T09   32
C                                                                         T09   33
      TLAG=EP(IM,1)                                                       T09   34
C                                                                         T09   35
C     VARIABLE TIME DELAY                                                 T09   36
C                                                                         T09   37
      IF (TLAG.LT.0.0) TLAG=ABS(TLAG)/S(2,IN,1)                           T09   38
      BYP=EP(IM,2)                                                        T09   39
      NV=EP(IM,3)                                                         T09   40
      N1=EP(IM,4)                                                         T09   41
C                                                                         T09   42
C     FLAG=1,OUTPUT FLOW CONTROL ,0 , NORMAL                              T09   43
      IF (EP(IM,5).GT.0.0) S(1,IN,2)=S(1,OUT,2)                           T09   44
C                                                                         T09   45
C     FOR FIRST PREDICTOR STEP, SET ALL INITIAL INFORMATION REQUIRED      T09   46
C     BY SX-MATRIX, FOR ALL FOLLOWING PREDICTOR STEPS SKIP THE DELAY      T09   47
C                                                                         T09   48
      IF (IG.LT.2) GO TO 3                                                T09   49
```

```
      IF (JSTART.NE.0) RETURN                                          T09  50
      ND=ND+1                                                          T09  51
      SX(1,ND,1)=0.0                                                   T09  52
C                                                                      T09  53
C     SET INPUT STREAM VALUES IN 1ST VECTOR OF SX-MATRIX               T09  54
C                                                                      T09  55
      DO 2 K=N1,NC4                                                    T09  56
      SX(1,ND,K)=S(2,IN,K)                                             T09  57
C                                                                      T09  58
C     SET THE REST OF VALUES IN STORES TO EXIT VALUE                   T09  59
C                                                                      T09  60
      DO 1 I=2,NV                                                      T09  61
      SX(I,ND,K)=S(1,OUT,K)                                            T09  62
C                                                                      T09  63
C     FILL TIME STORES INITIALLY WITH -1.0 VALUES                      T09  64
C                                                                      T09  65
      SX(I,ND,1)=-1.                                                   T09  66
1     CONTINUE                                                         T09  67
2     CONTINUE                                                         T09  68
      RETURN                                                           T09  69
C                                                                      T09  70
3     ND=ND+1                                                          T09  71
      MC(ND)=MC(ND)+1                                                  T09  72
C                                                                      T09  73
C     IF REPEAT THEN PICKED UP VALUES OF TIME STORED IN SX(,,1)        T09  74
C                                                                      T09  75
      IF (NSW.EQ.0) GO TO 5                                            T09  76
      MC(ND)=MC(ND)-1                                                  T09  77
      DO 4 I=2,NV                                                      T09  78
      SX(I,ND,1)=SX(I,ND,8)                                            T09  79
4     CONTINUE                                                         T09  80
C                                                                      T09  81
C     NORMAL OPERATION,STORE A COPY OF TIME VECTOR FOR FUTURE REPEAT    T09  82
C     THEN INCREMENT TIME VALUES AS TIME PROGRESS                      T09  83
C                                                                      T09  84
5     MD=MC(ND)                                                        T09  85
      IF (MD.GT.NV) MD=NV                                              T09  86
      DO 6 I=2,NV                                                      T09  87
      SX(I,ND,8)=SX(I,ND,1)                                            T09  88
6     CONTINUE                                                         T09  89
7     IF (MD.EQ.2) GO TO 8                                             T09  90
      SX(MD,ND,1)=SX(MD-1,ND,1)+H                                      T09  91
      MD=MD-1                                                          T09  92
      GO TO 7                                                          T09  93
8     SX(2,ND,1)=H                                                     T09  94
C                                                                      T09  95
C     SHIFT VALUES IN THE STORES RIGHT STARTING WITH RIGHT-MOST        T09  96
C                                                                      T09  97
      IF (NSW.EQ.1) GO TO 11                                           T09  98
      NV1=NV-1                                                         T09  99
      DO 10 K=N1,NC4                                                   T09 100
      DO 9 I=1,NV1                                                     T09 101
```

```
      II=NV-I+1                                                          T09 102
      L=II-1                                                             T09 103
      SX(II,ND,K)=SX(L,ND,K)                                             T09 104
 9    CONTINUE                                                           T09 105
10    CONTINUE                                                           T09 106
C                                                                        T09 107
C     TRANSFER VALUE OF INLET TO 1ST STORAGE OF DELAY VECTOR             T09 108
C                                                                        T09 109
11    DO 12 K=N1,NC4                                                     T09 110
      SX(1,ND,K)=S(I,IN,K)                                               T09 111
12    CONTINUE                                                           T09 112
C                                                                        T09 113
C     COMPARE STORED TIME VALUES WITH TIME LAG                           T09 114
C                                                                        T09 115
      DO 13 I=1,NV1                                                      T09 116
      IF (SX(I,ND,1)-TLAG) 13,14,17                                      T09 117
13    CONTINUE                                                           T09 118
C                                                                        T09 119
C     IF NONE IS GREATER THAN TLAG,EXIT THE LAST VALUE                   T09 120
C     IF TIME VALUE EQUALS T AG,EXIT CORRESPONDING VALUES                T09 121
C                                                                        T09 122
      I=NV                                                               T09 123
      IF (SX(NV,ND,1).EQ.-1.0) GO TO 14                                  T09 124
      PRINT 19                                                           T09 125
      STOP                                                               T09 126
14    DO 15 K=N1,NC4                                                     T09 127
      S(1,OUT,K)=SX(I,ND,K)*(1.-BYP)+S(1,IN,K)*BYP                       T09 128
15    CONTINUE                                                           T09 129
      RETURN                                                             T09 130
C                                                                        T09 131
C     IF SX(I).GT.TLAG,CHECK IF SX(I-1).GT.TLAG                          T09 132
C     IF SO SET THE TIME AT SX(I) TO -1.                                 T09 133
C                                                                        T09 134
16    SX(I,ND,1)=-1.                                                     T09 135
      I=I-1                                                              T09 136
17    L=I-1                                                              T09 137
C                                                                        T09 138
C     KEEP TESTING UNTIL ONLY ONE IS LEFT                                T09 139
C                                                                        T09 140
      IF (SX(L,ND,1).GE.TLAG) GO TO 16                                   T09 141
C                                                                        T09 142
C     INTERPOLATE FOR EXIT VALUE                                         T09 143
C                                                                        T09 144
      A=TLAG-SX(L,ND,1)                                                  T09 145
      B=SX(I,ND,1)-SX(L,ND,1)                                            T09 146
      DO 18 K=N1,NC4                                                     T09 147
      U=SX(L,ND,K)+(A/B)*(SX(I,ND,K)-SX(L,ND,K))                         T09 148
      S(1,OUT,K)=(1.-BYP)*U+BYP*S(1,IN,K)                                T09 149
18    CONTINUE                                                           T09 150
      RETURN                                                             T09 151
C                                                                        T09 152
19    FORMAT (20H ERROR IN TIME DELAY,/,23H NV MUST BE MADE LARGER)      T09 153
```

END

T09 154-

APPENDIX H: WILLIAMS-OTTO PLANT LISTINGS

Figure H.1 gives a listing of all the modules and the data set used in the Williams-Otto plant simulation. The distillation column modules are not included. The version of the executive used allowed for 10 equipment parameters.

FIGURE H.11 LISTING OF MODULES AND DATA SET FOR WILLIAMS-OTTO PLANT

```
      SUBROUTINE TYPE10
C
C     SUBROUTINE CSTR1
C
C     THIS MODULE REPRESENTS THE CONTINUOUS STIRRED TANK REACTOR
C        IN THE WILLIAMS-OTTO PLANT
C     MODULE IS STIFF
C
C     EQUIPMENT PARAMETERS
C
C     1 - VR   - INITIAL HOLDUP IN REACTOR (LB)
C     2 - VW   - HOLDUP IN COOLING COIL (LB)
C     3 - HWAW - OVERALL COOLING COIL HEAT TRANSFER COEFFICIENT
C                * EFFECTIVE HEAT TRANSFER AREA (BTU/(DEGREE F*HR))
C     4 - HSAS - OVERALL STEAM COIL HEAT TRANSFER COEFFICIENT
C                * EFFECTIVE HEAT TRANSFER AREA (BTU/(DEGREE F*HR))
C     5 - XLHC - LATENT HEAT OF CONDENSATION OF ENTERING STEAM (BTU/LB)
C
C     STREAMS
C
C     1 - INA   - A FEED STREAM
C     2 - INB   - B FEED STREAM
C     3 - INL   - L RECYCLE STREAM
C     4 - INC   - COOLING COIL INPUT STREAM
C     5 - INS   - STEAM COIL INPUT STREAM
C     6 - IOUT  - PRODUCT STREAM
C     7 - IOUTC - COOLING COIL OUTPUT STREAM
C     8 - IOUTS - STEAM COIL OUTPUT STREAM
C     9 - ISIGL - LEVEL CONTROL SIGNAL
C    10 - ISIGB - B FLOW CONTROL SIGNAL (RATIO FA/FB)
C
C     NOMENCLATURE
C
C     FA   - A FEED STREAM FLOW RATE (LB/HR)
C     FB   - B FEED STREAM FLOW RATE (LB/HR)
C     FL   - L RECYCLE STREAM FLOW RATE (LB/HR)
C     FR   - PRODUCT STREAM FLOW RATE (LB/HR)
C     FW   - COOLING COIL FLOW RATE (LB/HR)
C     TA   - A FEED STREAM TEMPERATURE (DEGREES F)
C     TB   - B FEED STREAM TEMPERATURE (DEGREES F)
C     TL   - L RECYCLE STREAM TEMPERATURE (DEGREES F)
C     TINC - INLET COOLING COIL TEMPERATURE (DEGREES F)
C     TS   - STEAM COIL TEMPERATURE (DEGREES F)
C     SFLOW- STEAM FLOW RATE (LB/HR)
C     CPR  - HEAT CAPACITY OF REACTOR MIXTURE (BTU/(LB*DEGREE F))
C            (ASSUMED SAME FOR ALL COMPONENTS)
C     CPW  - HEAT CAPACITY OF COOLANT (BTU/(LB*DEGREE F))
C
```

```
      COMMON/MAT/MP(35,13),EP(35,10),S(2,45,11),EX(1)
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF
      COMMON /PTAB/ IGFLAG,PP(10,20)
      COMMON /UNIT/ IM,NMP
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO
      COMMON/ROW/IROW(1000)
      COMMON/COLUMN/JCOL(1000)
      COMMON/JACOB/XJACOB(1000)
      COMMON /RETN/ K1,K2,K3,H1,H2,H3,B1,B2,B3
      DIMENSION IROWS(33),JCOLS(33)
      DIMENSION Y(9),DERY(9)
      REAL K1,K2,K3
      DATA IROWS/3*1,4*2,5*3,4*4,4*5,4*6,6*7,2*8,9/
      DATA JCOLS/1,2,7,1,2,3,7,1,2,3,6,7,2,3,4,7,3,5,6,7,2,3,6,7,1,2,3,
     16,7,8,7,8,9/
      DATA CPR,CPW/0.4,1.0/
C
C     CALCULATE MODULE PARAMETERS
C
      IF(JSTART.NE.0.OR.IG.EQ.1) GO TO 2
      VR=EP(IM,1)
      VW=EP(IM,2)
      HWAW=EP(IM,3)
      HSAS=EP(IM,4)
      XLHC=EP(IM,5)
      INA=MP(IM,3)
      INB=MP(IM,4)
      INL=MP(IM,5)
      INC=MP(IM,6)
      INS=MP(IM,7)
      IOUT=IABS(MP(IM,8))
      IOUTC=IABS(MP(IM,9))
      IOUTS=IABS(MP(IM,10))
      ISIGL=IABS(MP(IM,11))
      ISIGB=IABS(MP(IM,12))
      DO 31 I=1,6
   31 Y(I)=S(2,IOUT,I+5)
      Y(7)=S(2,IOUT,4)
      Y(8)=S(2,IOUTC,4)
      Y(9)=VR
    2 CONTINUE
C     USE NEWTON-RAPHSON ITERATION WITH STIFF OPTION
      ITER=1
      FA=S(IG,INA,3)
      FB=S(IG,INB,3)
      FL=S(IG,INL,3)
      FR=S(IG,IOUT,3)
      FW=S(IG,INC,3)
      TA=S(IG,INA,4)
      TB=S(IG,INB,4)
      TL=S(IG,INL,4)
```

```
      TINC=S(IG,INC,4)
C
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS
C
      IF(IG.EQ.2) GO TO 1
      NZERO=33
      DO 81 I=1,33
      IROW(I)=IROWS(I)
   81 JCOL(I)=JCOLS(I)
C
C     CALL REACT TO GET REACTION RATE COEFFICIENTS AND
C        HEATS OF REACTION
      TEMP=Y(7)+459.69
      CALL REACT(TEMP)
      T1=K1*Y(1)*Y(2)*B1/(TEMP**2)
      T2=K2*Y(2)*Y(3)*B2/(TEMP**2)
      T3=K3*Y(3)*Y(6)*B3/(TEMP**2)
      XJACOB(1)=-FR/VR-K1*Y(2)
      XJACOB(2)=-K1*Y(1)
      XJACOB(3)=-T1
      XJACOB(4)=-K1*Y(2)
      XJACOB(5)=-FR/VR-K1*Y(1)-K2*Y(3)
      XJACOB(6)=-K2*Y(2)
      XJACOB(7)=-T1-T2
      XJACOB(8)=2.0*K1*Y(2)
      XJACOB(9)=2.0*K1*Y(1)-2.0*K2*Y(3)
      XJACOB(10)=-FR/VR-2.0*K2*Y(2)-K3*Y(6)
      XJACOB(11)=-K3*Y(3)
      XJACOB(12)=2.0*T1-2.0*T2-T3
      XJACOB(13)=2.0*K2*Y(3)
      XJACOB(14)=2.0*K2*Y(2)
      XJACOB(15)=-FR/VR
      XJACOB(16)=2.0*T2
      XJACOB(17)=1.5*K3*Y(6)
      XJACOB(18)=-FR/VR
      XJACOB(19)=1.5*K3*Y(3)
      XJACOB(20)=1.5*T3
      XJACOB(21)=K2*Y(3)
      XJACOB(22)=K2*Y(2)-0.5*K3*Y(6)
      XJACOB(23)=-0.5*K3*Y(3)-FR/VR
      XJACOB(24)=T2-0.5*T3
      XJACOB(25)=-2.0*K1*Y(2)*H1/CPR
      XJACOB(26)=-(2.0*K1*Y(1)*H1+3.0*K2*Y(3)*H2)/CPR
      XJACOB(27)=-(3.0*K2*Y(2)*H2+1.5*K3*Y(6)*H3)/CPR
      XJACOB(28)=-1.5*K3*Y(3)*H3/CPR
      XJACOB(29)=-(2.0*T1*H1+3.0*T2*H2+1.5*T3*H3)/CPR
     1-(HWAW+HSAS)/(VR*CPR)-(FL+FA+FB)/VR
      XJACOB(30)=HWAW/(VR*CPR)
      XJACOB(31)=HWAW/(VW*CPW)
      XJACOB(32)=-(HWAW+FW*CPW)/(VW*CPW)
      XJACOB(33)=0.0
      GO TO 12
```

```
    1 CONTINUE
C
C     CALCULATE DERIVATIVES
C
C     CALL REACT TO GET REACTION RATE COEFFICIENTS AND
C        HEATS OF REACTION
      TEMP=Y(7)+459.69
      CALL REACT(TEMP)
   12 CONTINUE
      S1=K1*Y(1)*Y(2)
      S2=K2*Y(2)*Y(3)
      S3=K3*Y(3)*Y(6)
      TS=S(IG,INS,4)
C     IF STEAM FLOW IS ZERO,THERE IS NO HEAT TRANSFER
      IF(S(IG,INS,3).LT.0.001) TS=Y(7)
C     A MASS BALANCE
      DERY(1)=1.0/VR*(FA+FL*S(IG,INL,6)-FR*Y(1))-S1
C     B MASS BALANCE
      DERY(2)=1.0/VR*(FB+FL*S(IG,INL,7)-FR*Y(2))-S1-S2
C     C MASS BALANCE
      DERY(3)=1.0/VR*(FL*S(IG,INL,8)-FR*Y(3))+2.0*S1-2.0*S2-S3
C     E MASS BALANCE
      DERY(4)=1.0/VR*(FL*S(IG,INL,9)-FR*Y(4))+2.0*S2
C     G MASS BALANCE
      DERY(5)=1.0/VR*(FL*S(IG,INL,10)-FR*Y(5))+1.5*S3
C     P MASS BALANCE
      DERY(6)=1.0/VR*(FL*S(IG,INL,11)-FR*Y(6))+S2-0.5*S3
C     TOTAL HEAT BALANCE
      DERY(7)=1.0/(VR*CPR)*(-2.0*S1*H1*VR-3.0*S2*H2*VR-1.5*S3*H3*VR
     1-HWAW*(Y(7)-Y(8))+HSAS*(TS-Y(7))-FL*CPR*(Y(7)-TL)-FA*CPR*(Y(7)-TA)
     2-FB*CPR*(Y(7)-TB))
C     HEAT BALANCE FOR COOLING COIL
      DERY(8)=1.0/(WW*CPW)*(HWAW*(Y(7)-Y(8))+FW*CPW*(TINC-Y(8)))
C     TOTAL MASS BALANCE
      DERY(9)=FA+FB+FL-FR
      CALL DIFSUB(9,Y,DERY)
      IF(IDERY.NE.0) GO TO 1
C
C     CALCULATE STREAM OUTPUT
C
C     PRODUCT STREAM
C     NORMALIZE MASS FRACTIONS
      SUM=0.0
      DO 55 I=1,6
   55 SUM=SUM+Y(I)
      DO 6 I=1,6
      Y(I)=Y(I)/SUM
    6 S(1,IOUT,I+5)=Y(I)
      S(1,IOUT,4)=Y(7)
C     COOLING COIL
      S(1,IOUTC,3)=S(1,INC,3)
      S(1,IOUTC,4)=Y(8)
```

```
C       STEAM COIL
        IF(S(IG,INS,3).LT.0.001) GO TO 56
        SFLOW=HSAS/XLHC*(S(IG,INS,4)-Y(7))
        S(1,INS,3)=SFLOW
   56   S(1,IOUTS,3)=S(1,INS,3)
        S(1,IOUTS,4)=S(1,INS,4)
C       CONTROL SIGNALS
        S(1,ISIGL,3)=Y(9)
        S(1,ISIGB,3)=FA/FB
        RETURN
        END
```

```
      SUBROUTINE REACT(T)
C
C     THIS SUBROUTINE SUPPLIES REACTION RATE COEFFICIENTS AND HEATS
C     OF REACTION OF THE WILLIAMS-OTTO PLANT REACTION SCHEME BELOW
C
C        A + B = C
C        C + B = P + E
C        P + C = G
C
C     T IS DEFINED IN DEGREES RANKINE
C
      COMMON /RETN/ K1,K2,K3,H1,H2,H3,B1,B2,B3
      REAL K1,K2,K3
C
C     CONSTANTS OF ARRHENIUS EQUATION :  K = A*EXP(-B/T)
C
      DATA A1,A2,A3/5.9755E+09,2.5962E+12,9.6283E+15/
      DATA B1,B2,B3/12000.0,15000.0,20000.0/
C
C     ARRHENIUS REACTION COEFFICIENTS
C
      IF(T.LT.500.0) T=500.0
      IF(T.GT.1000.0) T=1000.0
      K1=A1*EXP(-B1/T)
      K2=A2*EXP(-B2/T)
      K3=A3*EXP(-B3/T)
C
C     HEATS OF REACTION
C
      H1=-125.0
      H2=-50.0
      H3=-143.0
      RETURN
      END
```

```
      SUBROUTINE TYPE11
C
C     SUBROUTINE EXCH1
C
C     THIS MODULE REPRESENTS A SHELL AND TUBE HEAT EXCHANGER
C     BOTH THE SHELL AND TUBE SIDE ARE MODELLED AS WELL-STIRRED TANKS
C     MODULE IS NONSTIFF
C
C     EQUIPMENT PARAMETERS
C
C     1 - VHT - TUBE SIDE MASS HOLDUP (LB)
C     2 - VHS - SHELL SIDE (COOLANT) MASS HOLDUP (LB H2O)
C     3 - AH  - AREA OF HEAT TRANSFER (FT**2)
C     4 - HH  - OVERALL HEAT TRANSFER COEFFICIENT
C               (BTU/(DEGREE F*FT**2*HR))
C     5 - CPW - COOLANT HEAT CAPACITY (BTU/(LB*DEGREE F))
C
C     STREAMS
C
C     1 - INT   - INLET TUBE SIDE
C     2 - INS   - INLET SHELL SIDE
C     3 - IOUTT - OUTLET TUBE SIDE
C     4 - IOUTS - OUTLET SHELL SIDE
C
C     NOMENCLATURE
C
C     FT   - TUBE SIDE FLOW RATE (LB/HR)
C     FS   - SHELL SIDE FLOW RATE (LB/HR)
C     TINT - INLET TUBE SIDE TEMPERATURE (DEGREES F)
C     TINS - INLET SHELL SIDE TEMPERATURE (DEGREES F)
C     CPR  - TUBE SIDE HEAT CAPACITY (BTU/(LB*DEGREE F))
C     TLM  - LOG MEAN TEMPERATURE
C
      COMMON/MAT/MP(35,13),EP(35,10),S(2,45,11),EX(1)
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF
      COMMON /PTAB/ IGFLAG,PP(10,20)
      COMMON /UNIT/ IM,NMP
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO
      DIMENSION Y(8),DERY(8)
      DATA CPR/0.4/
C
C     CALCULATE MODULE PARAMETERS
C
      IF(JSTART.NE.0.OR.IG.EQ.1) GO TO 2
      VHT=EP(IM,1)
      VHS=EP(IM,2)
      AH=EP(IM,3)
      HH=EP(IM,4)
      CPW=EP(IM,5)
      INT=MP(IM,3)
      INS=MP(IM,4)
```

```
      IOUTT=IABS(MP(IM,5))
      IOUTS=IABS(MP(IM,6))
      Y(1)=S(IG,IOUTT,4)
      Y(2)=S(IG,IOUTS,4)
      DO 9 I=1,NCOMP
    9 Y(I+2)=S(IG,IOUTT,I+5)
    2 CONTINUE
C     USE DIRECT ITERATION IF STIFF OPTION IS USED
      ITER=0
      FT=S(IG,INT,3)
      FS=S(IG,INS,3)
      TINT=S(IG,INT,4)
      TINS=S(IG,INS,4)
    1 CONTINUE
C
C     CALCULATE DERIVATIVES
C
C     CALCULATE LOG MEAN TEMPERATURE
      TLM=((TINT-Y(2))-(Y(1)-TINS))/ALOG((TINT-Y(2))/(Y(1)-TINS))
C     HEAT BALANCE (TUBE SIDE)
      DERY(1)=1.0/(VHT*CPR)*(FT*CPR*(TINT-Y(1))-HH*AH*TLM)
C     HEAT BALANCE (SHELL SIDE)
      DERY(2)=1.0/(VHS*CPW)*(FS*CPW*(TINS-Y(2))+HH*AH*TLM)
C     COMPONENT MASS BALANCES (TUBE SIDE)
      DO 10 I=1,NCOMP
   10 DERY(I+2)=FT/VHT*(S(IG,INT,I+5)-Y(I+2))
      N=2+NCOMP
      CALL DIFSUB(N,Y,DERY)
      IF(IDERY.NE.0) GO TO 1
C
C     CALCULATE STREAM OUTPUT
C
      S(1,IOUTT,4)=Y(1)
      S(1,IOUTS,4)=Y(2)
      S(1,IOUTS,3)=S(1,INS,3)
      SUM=0.0
      DO 15 I=3,N
   15 SUM=SUM+Y(I)
C     PUT NORMALIZED MASS FRACTIONS INTO OUTLET TUBE SIDE STREAM
      DO 16 I=3,N
      Y(I)=Y(I)/SUM
   16 S(1,IOUTT,I+3)=Y(I)
      RETURN
      END
```

```
      SUBROUTINE TYPE12
C
C     SUBROUTINE DCAN1
C
C     THIS MODULE REPRESENTS A DECANTER
C     ONE COMPONENT IS SEPARATED BY GRAVITY FROM THE ENTERING STREAM
C     BOTH TOP AND BOTTOM LAYERS ARE MODELLED AS WELL-STIRRED TANKS
C     FUNCTION X6 GIVES WEIGHT FRACTION OF SEPARATED COMPONENT ENTERING
C        TOP LAYER OF DECANTER AS FUNCTION OF TEMPERATURE
C     CONTROL SIGNALS FOR THE HOLDUP IN TOP AND BOTTOM LAYERS ARE
C        EACH SENT TO CONTROLLERS
C     MODULE IS NONSTIFF
C
C     EQUIPMENT PARAMETERS
C
C     1 - HUPT - INITIAL HOLDUP IN TOP LAYER (LB)
C     2 - HUPB - INITIAL HOLDUP IN BOTTOM LAYER (LB)
C     3 - ISEP - NUMBER OF COMPONENT TO BE SEPARATED FROM ENTERING STREAM
C
C     STREAMS
C
C     1 - IN    - INLET
C     2 - IOUTT - OUTLET FROM TOP OF DECANTER
C     3 - IOUTB - OUTLET FROM BOTTOM OF DECANTER
C     4 - ISIGT - LEVEL CONTROL SIGNAL (TOP)
C     5 - ISIGB - LEVEL CONTROL SIGNAL (BOTTOM)
C
C     NOMENCLATURE
C
C     FIN   - TOTAL INLET FLOW (LB/HR)
C     FINT  - INLET FLOW ENTERING TOP LAYER (LB/HR)
C     FINB  - INLET FLOW ENTERING BOTTOM LAYER (LB/HR)
C     FOUT  - TOTAL OUTLET FLOW (LB/HR)
C     FOUTT - OUTLET FLOW FROM TOP LAYER (LB/HR)
C     FOUTB - OUTLET FLOW FROM BOTTOM LAYER (LB/HR)
C     FT    - MASS FRACTIONS ENTERING TOP LAYER
C     FEC   - MASS FRACTIONS IN TOP LAYER
C     WFG   - MASS FRACTION OF SEPARATED COMPONENT ENTERING TOP LAYER
C     TEMP  - DECANTER TEMPERATURE (DEGREES F)
C
      COMMON/MAT/MP(35,13),EP(35,10),S(2,45,61),EX(1)
      COMMON /CON/ IG,NCOMP,NCS,M,NE,NS,NPP,NPOL,TMAX,IORDER,NGRAPH
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NQVMAX,ICONV,NOPPT,ISTIFF
      COMMON /PTAB/ IGFLAG,PP(10,20)
      COMMON /UNIT/ IM,NMP
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO
      DIMENSION Y(9),DERY(9),FT(6),FEC(6)
C
C     CALCULATE MODULE PARAMETERS
C
      IF(JSTART.NE.0.OR.IG.EQ.1) GO TO 2
      HUPT=EP(IM,1)
```

```
      HUPB=EP(IM,2)
      ISEP=EP(IM,3)
      IN=MP(IM,3)
      IOUTT=IABS(MP(IM,4))
      IOUTB=IABS(MP(IM,5))
      ISIGT=IABS(MP(IM,6))
      ISIGB=IABS(MP(IM,7))
      Y(1)=HUPT+HUPB
      Y(2)=S(IG,IOUTT,4)
      N=NCOMP+2
      DO 4 I=3,N
    4 Y(I)=HUPT*S(IG,IOUTT,I+3)
      Y(N+1)=HUPB
    2 CONTINUE
C     USE DIRECT ITERATION IF STIFF OPTION IS USED
      ITER=0
    1 CONTINUE
C
C     CALCULATE DERIVATIVES
C
      FIN=S(IG,IN,3)
      FOUTT=S(IG,IOUTT,3)
      FOUTB=S(IG,IOUTB,3)
C     TOTAL MASS BALANCE
      DERY(1)=FIN-FOUTT-FOUTB
C     TOTAL HEAT BALANCE
      DERY(2)=1.0/Y(1)*(FIN*S(IG,IN,4)-(FOUTT+FOUTB)*Y(2))
      TEMP=Y(2)
C     FUNCTION XG DETERMINES MASS FRACTION OF SEPARATED COMPONENT
C        ENTERING TOP LAYER
      WFG=XG(TEMP)
C     WFG CANNOT BE GREATER THAN ENTERING MASS FRACTION
      IF(WFG.GT.S(IG,IN,ISEP+5)) WFG=S(IG,IN,ISEP+5)
      FINT=FIN*(1.0-S(IG,IN,ISEP+5))/(1.0-WFG)
      FINB=FIN-FINT
      DO 101 I=1,NCOMP
  101 FT(I)=FIN*S(IG,IN,I+5)/FINT
      FT(ISEP)=WFG
      SUM=0.0
      NF=NCOMP+2
      DO 3 I=3,NF
    3 SUM=SUM+Y(I)
      DO 14 I=1,NCOMP
   14 FEC(I)=Y(I+2)/SUM
C     COMPONENT MASS BALANCES FOR TOP LAYER
      DO 20 I=3,NF
   20 DERY(I)=FINT*FT(I-2)-FOUTT*FEC(I-2)
C     MASS BALANCE FOR BOTTOM LAYER
      N=NCOMP+3
      DERY(N)=FINB-FOUTB
      CALL DIFSUB(N,Y,DERY)
      IF(IDERY.NE.0) GO TO 1
```

```
C
C       CALCULATE STREAM OUTPUT
C
C
C       BOTTOM OUTLET STREAM
        S(1,IOUTB,4)=TEMP
C       TOP OUTLET STREAM
        S(1,IOUTT,4)=TEMP
        SUM=0.0
        DO 15 I=1,NCOMP
   15   SUM=SUM+Y(I,2)
        DO 16 I=1,NCOMP
   16   S(1,IOUTT,I+5)=Y(I,2)/SUM
C       CONTROL SIGNALS
        S(1,ISIGT,3)=Y(1)
        S(1,ISIGB,3)=Y(9)
        RETURN
        END
```

```
      FUNCTION XG(T)
C
C     THIS FUNCTION CALCULATES THE WEIGHT FRACTION OF SEPARATED
C     COMPONENT (G IN THE WILLIAMS-OTTO PLANT) ENTERING THE TOP LAYER
C     OF DECANTER AS A FUNCTION OF TEMPERATURE
C
C     T IS TEMPERATURE IN DEGREES F
C
      IF(T.LE.100.0) GO TO 1
      IF(T-120.0) 2,3,4
    1 XG=0.0
      RETURN
    2 XG=0.005769+0.13365*(0.01*T-1.1)+0.7596*(0.01*T-1.1)**2
      IF(XG.LT.0.0) XG=0.0
      RETURN
    3 XG=0.050865
      RETURN
    4 XG=0.0894+0.3125*(0.01*T-1.24)-0.9*(0.01*T-1.24)**2
      IF(T.GT.128.0) PRINT 10,T
   10 FORMAT(23H DECANTER TEMPERATURE =,F6.1,42H REQUIRES EXTRAPOLATION
     1OF SOLUBILITY DATA)
      RETURN
      END
```

```
      SUBROUTINE TYPE13
C
C     SUBROUTINE CONV1
C
C     THIS MODULE CHANGES MASS FLOW AND MASS FRACTIONS TO MOLAR FLOW
C     AND MOLE FRACTIONS OR VICE-VERSA
C
C     EQUIPMENT PARAMETERS
C
C     1 - ICONV - =1,CHANGES MASS TO MOLES
C                 =2,CHANGES MOLES TO MASS
C
      COMMON/MAT/MP(35,13),EP(35,10),S(2,48,11),EX(1)
      COMMON /CON/ IG,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON /PTAB/ IGFLAG,PP(10,20)
      COMMON /UNIT/ IM,NMP
      DIMENSION XMW(10),X(10),F(10),FNEW(10)
C
C     GET MODULE PARAMETERS
C
      ICONV=EP(IM,1)
      IF(ICONV.NE.1.AND.ICONV.NE.2) GO TO 75
      IN=MP(IM,3)
      IOUT=IABS(MP(IM,4))
      FLOWT=S(IG,IN,3)
      DO 10 I=1,NCOMP
C     GET MOLECULAR WEIGHTS
      XMW(I)=PP(I,1)
      X(I)=S(IG,IN,I+5)
      F(I)=FLOWT*X(I)
   10 CONTINUE
      IF(ICONV.EQ.2) GO TO 91
C
C     CHANGE MASS TO MOLES
C
      DO 11 I=1,NCOMP
   11 FNEW(I)=F(I)/XMW(I)
      GO TO 90
C
C     CHANGE MOLES TO MASS
C
   91 DO 12 I=1,NCOMP
   12 FNEW(I)=F(I)*XMW(I)
   90 CONTINUE
      SUM=0.0
      DO 13 I=1,NCOMP
   13 SUM=SUM+FNEW(I)
      DO 14 I=1,NCOMP
   14 X(I)=FNEW(I)/SUM
C
C     CALCULATE STREAM OUTPUT
C
```

```
      S(IG,IOUT,3)=SUM
      S(IG,IOUT,4)=S(IG,IN,4)
      S(IG,IOUT,5)=S(IG,IN,5)
      DO 15 I=1,NCOMP
   15 S(IG,IOUT,I+5)=X(I)
      RETURN
   75 PRINT 20
   20 FORMAT(37H CONV01 EQUIPMENT PARAMETER NE.1.OR.2)
      STOP
      END
```

```
      SUBROUTINE TYPE14
C
C     SUBROUTINE SPLT1
C
C     THIS MODULE SPLITS AN INCOMING STREAM INTO 2 EXIT STREAMS
C     ACCORDING TO A GIVEN RATIO
C
C     EQUIPMENT PARAMETERS
C
C     1 - RATIO - DESIRED FLOW RATIO OF FIRST EXIT STREAM
C                 TO ENTERING STREAM
C
      COMMON/MAT/MP(35,13),EP(35,10),S(2,45,11),EX(1)
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON /UNIT/ IM,NMP
C
      RATIO=EP(IM,1)
      IN=MP(IM,3)
      IOUT1=IABS(MP(IM,4))
      IOUT2=IABS(MP(IM,5))
      S(1,IOUT1,3)=S(IG,IN,3)*RATIO
      S(1,IOUT2,3)=S(IG,IN,3)-S(1,IOUT1,3)
      DO 1 I=4,NC5
      S(1,IOUT1,I)=S(IG,IN,I)
    1 S(1,IOUT2,I)=S(IG,IN,I)
      RETURN
      END
```

```
      SUBROUTINE TYPE1                                                      T01    1
C                                                                           T01    2
C     SUBROUTINE VALV1                                                      T01    3
C                                                                           T01    4
C     V-PORT (PARABOLIC) CONTROL VALVE                                      T01    5
C                                                                           T01    6
C     EQUIPMENT PARAMETERS                                                  T01    7
C                                                                           T01    8
C     1 - NOT USED                                                          T01    9
C     2 - VALVE CONSTANT                                                    T01   10
C     3 - ACTION (+=DIRECT,-=REVERSE)                                       T01   11
C                                                                           T01   12
      COMMON /UNIT/ IM,NMP                                                  T01   13
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                     T01   14
      COMMON /CON/ IG,NCOMP,NCS,M,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH         T01   15
C                                                                           T01   16
      MI=MP(IM,3)                                                           T01   17
      MO=IABS(MP(IM,4))                                                     T01   18
C     CHECK THAT THE VALVE SIGNAL IS IN THE RANGE 0-100                     T01   19
      V=S(1,MI,3)                                                           T01   20
      IF (V.LT.0.) V=0.                                                     T01   21
      IF (V.GT.100.) V=100.                                                 T01   22
C     ACTION                                                                T01   23
      A=EP(IM,3)                                                            T01   24
      IF (A.LT.0.) S(1,MO,3)=EP(IM,2)*(100.-V)**2                           T01   25
      IF (A.GE.0.) S(1,MO,3)=EP(IM,2)*V**2                                  T01   26
      RETURN                                                                T01   27
      END                                                                   T01   28
```

```
      SUBROUTINE TYPE4
      COMMON /UNIT/ IM,NMP
      COMMON/MAT/MP(35,13),EP(35,10),S(2,45,11),EX(1)
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
       COMMON/CNTR/NCTR
       DIMENSION SIGMA(10)
      INTEGER OUT
      DATA SIGMA/10*0.0/
C     ***********************************************************************
C
C      CONTROLLER MODULE        BY  R.MILLARES , NOV 1974
C
C     ***********************************************************************
C
C      THIS CONTROLLER HAS FOUR OPTIONS! PROPORTIONAL, PROPOR, INTEGRAL,
C       PROP, DERIVATIVE AND PROPOR, INTEGRAL PLUS DERIVATIVE ACTIONS
C
C      THE FOLLOWING PARAMETERS ARE REQUIRED!
C       1, EP(IM,1)=   VARIABLE TO CONTROL
C       2, EP(IM,2)=   CONTROLLER UPPER LIMIT OF OPERATION (INPUT VAR)
C       3, EP(IM,3)=   CONTROLLER LOWER LIMIT
C       4, EP(IM,4)=   SET POINT, FOR VARIABLE S.P, SET THIS TO 0, AND
C             ASSIGN STREAM NUMBER ON MP= ARRAY, AND STREAM FLAG= 12
C       5, EP(IM,5)=   TYPE OF CONTROLLER
C                      1 = PROPORTIONAL
C                      2 = PI
C                      3 = PD
C                      4 = PID
C                      5 = ON-OFF (=EP(IM,6),GT,SETP,OFF,LT,SETP)
C                      6 = ON-OFF (=EP(IM,6),LT,SETP,OFF,GT,SETP)
C       6, EP(IM,6)=   DESIRED OUTPUT SIGNAL FOR ERROR= 0,
C       7, EP(IM,7)=   GAIN, SEND(+) IF DIRECT ACTION IS DESIRED OR (-)
C             FOR REVERSE ACTION
C       8, EP(IM,8)=   INTEGRAL CONSTANT (RESET)
C       9, EP(IM,9)=   DERIVATIVE CONSTANT
C       10,EP(IM,10)=   TRACE OPTION (=1)
C
C     ***********************************************************************
        NTR=EP(IM,10)
C
C
        IF(NTR.EQ.1) PRINT 500
500     FORMAT(5X,"ENTERING CONTROLLER")
C
C
        IN = MP(IM,3)
       OUT=MP(IM,4)
        K = EP(IM,1)
        N = EP(IM,5)
      IF(N.GT.4) GO TO 200
        IF(IG.NE.1)RETURN
```

```
      NCTR = NCTR+1
      RANGE = EP(IM,2)-EP(IM,3)
      IF (OUT.LT.0) GO TO 16
      SETP = EP(IM,4)*50./RANGE
      OLDERR = (S(2,IN,K)*50./RANGE)-SETP
      GO TO 18
16    IN2 = -OUT
      OUT =  -MP(IM,5)
      IF (S(1,IN2,2).NE.11.) GO TO 15
      IN = IN2
      IN2 = MP(IM,3)
15    SETP =S(1,IN2,3)
      OLDERR = (S(2,IN,K)*50./RANGE)-S(2,IN2,3)
   18 CONTINUE
      S(1,IN,K) = S(2,IN,K)
30    ERR=(S(1,IN,K)*50./RANGE)-SETP
      PD = 0.
      PI = 0.
      GO TO (25,26,27,28),N
27    PD = EP(IM,9)*(ERR-OLDERR)/H
      GO TO 25
28    PD = EP(IM,9)*(ERR-OLDERR)/H
26    SIGMA(NCTR) = SIGMA(NCTR)+(H*(ERR+OLDERR)/2.)
      PI = SIGMA(NCTR)/EP(IM,8)
25    S(1,OUT,3) = EP(IM,6)+(EP(IM,7)*(ERR+PI+PD))
      RETURN
  200 CONTINUE
      IF(IG.EQ.1) RETURN
      T=S(1,IN,K)
      IF(N.EQ.6) GO TO 201
      IF(T.GE.EP(IM,4)) T=EP(IM,6)
      IF(T.LT.EP(IM,4)) T=0.0
      GO TO 202
  201 CONTINUE
      IF(T.LE.EP(IM,4)) T=EP(IM,6)
      IF(T.GT.EP(IM,4)) T=0.0
  202 S(1,OUT,3)=T
      RETURN
      END
```

SIMULATION OF WILLIAMS-OTTO PLANT

```
BEGIN
TIME        1.0
COMPS       6.0
HMIN        0.00000001
IN/OUT      10.0
PRINTING    50.0
ORDER       2.0
LIBRARY     4.0
RORM2       21.0
REBO2       22.0
COLM1       27.0
COND1       17.0
PROCESS
CONT1       1.0
            6.0         -7.0        0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            3.0         4872.0      4408.0      4640.0      2.0
            50.0        5.0         1.0         0.0         0.0
VALV1       2.0
            7.0         -14.0       0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0         38.3844     1.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
CONT1       3.0
            14.0        -8.0        0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            4.0         0.0         0.0         175.0       6.0
            50.0        0.0         0.0         0.0         0.0
VALV1       4.0
            8.0         -9.0        0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0         0.4         1.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
CONT1       5.0
            14.0        -11.0       0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            4.0         181.0       175.0       180.0       1.0
            10.0        100.0       0.0         0.0         0.0
VALV1       6.0
            11.0        -12.0       0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0         51.148618   1.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
CONT1       7.0
            4.0         -5.0        0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
```

```
                3.0             0.50            0.37            0.43478260872.0
                50.0            0.1             2.0             0.0             0.0
VALV1           8.0
                5.0             -2.0            0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0             13.34           1.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
CSTR1           9.0
                1.0             2.0             3.0             12.0            9.0
                -14.0           -13.0           -10.0           -6.0            -4.0
                4640.0          106.08          5000.0          5000.0          945.5
                0.0             0.0             0.0             0.0             0.0
CONT1           10.0
                18.0            -15.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                4.0             101.0           80.0            100.0           1.0
                50.0            1.0             0.0             0.0             0.0
VALV1           11.0
                15.0            -16.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0             61.40           1.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
EXCH1           12.0
                14.0            16.0            -18.0           -17.0           0.0
                0.0             0.0             0.0             0.0             0.0
                274.0           830.0           569.0           82.5            1.0
                0.0             0.0             0.0             0.0             0.0
CONT1           13.0
                22.0            -23.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                3.0             10481.331       9483.109        9982.22         2.0
                50.0            2.0             1.0             0.0             0.0
VALV1           14.0
                23.0            -24.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0             36.90           1.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
CONT1           15.0
                19.0            -20.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                3.0             3241.35         2932.65         3087.0          2.0
                50.0            2.0             1.0             0.0             0.0
VALV1           16.0
                20.0            -21.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0             1.4848          1.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
DCAN1           17.0
                18.0            -24.0           -21.0           -22.0           -19.0
                0.0             0.0             0.0             0.0             0.0
                4895.22         3087.0          5.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
```

CONV1	18.0				
	24.0	-25.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	1.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
REBO2	19.0				
	39.0	38.0	-40.0	0.0	0.0
	0.0	0.0	0.0	0.0	1.0
	7682500.0	15750.0	120.0	0.5	13.0
	14.357	-3818.33	227.21	0.0	0.0
RDRM2	20.0				
	35.0	40.0	-38.0	-34.0	-36.0
	0.0	0.0	0.0	0.0	0.0
	70.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
COLM1	21.0				
	25.0	30.0	34.0	-35.0	-26.0
	0.0	-28.0	0.0	0.0	0.0
	10.0	5.0	0.0	8.0	0.1
	1.0	9.0	1.0	0.0	0.0
COND1	22.0				
	26.0	-27.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	0.0	2.0	0.0	0.0	0.0
STIR1	23.0				
	27.0	-30.0	-31.0	-33.0	0.0
	0.0	0.0	0.0	0.0	0.0
	40.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
CONT1	24.0				
	31.0	-32.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	3.0	65.0	35.0	50.0	4.0
	50.0	1.5	0.5	0.1	0.0
VALV1	25.0				
	32.0	-33.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	0.0	0.01342	1.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
CONT1	26.0				
	28.0	-29.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	4.0	110.0	80.0	95.0	2.0
	90.0	1.5	0.5	0.1	0.0
VALV1	27.0				
	29.0	-30.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
	0.0	0.24644	1.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0
CONT1	28.0				
	36.0	-37.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0

```
                3.0             100.0           60.0            80.0            4.0
                50.0            1.5             0.5             0.1             0.0
VALV1           29.0
                37.0            -38.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0             0.2644          1.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
CONV1           30.0
                38.0            -41.0           0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                2.0             0.0             0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
SPLT1           31.0
                41.0            -3.0            -42.0           0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.5992170270.0                  0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
END
STREAMS         42.0
EXPLICIT
                1.0             1.0             14500.0         70.0            44.1
                1.0             0.0             0.0             0.0             0.0
                0.0
                2.0             1.0             33350.0         70.0            44.1
                0.0             1.0             0.0             0.0             0.0
                0.0
                3.0             1.0             48111.0         100.0           44.1
                .1329111754     .4209654006     .0268382730     .3811766320     0.0
                .0381085190
                4.0             11.0            0.43478         0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0
                5.0             11.0            50.0            0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0
                6.0             11.0            4640.0          0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0
                7.0             11.0            50.0            0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0
                8.0             11.0            50.0            0.0             0.0
                0.0             0.0             0.0             0.0             0.0
                0.0
                9.0             1.0             951.877         250.0           29.825
                0.0             0.0             0.0             0.0             0.0
                0.0
                10.0            1.0             951.877         250.0           29.825
                0.0             0.0             0.0             0.0             0.0
                0.0
                11.0            11.0            0.0             0.0             0.0
                0.0             0.0             0.0             0.0             0.0
```

```
0.0
12.0          1.0           0.0           60.0          14.7
0.0           0.0           0.0           0.0           0.0
0.0
13.0          1.0           0.0           62.0          14.7
0.0           0.0           0.0           0.0           0.0
0.0
14.0          1.0           95961.0       70.0          44.1
0.30303       0.69697       0.0           0.0           0.0
0.0
15.0          11.0          50.0          0.0           0.0
0.0           0.0           0.0           0.0           0.0
0.0
16.0          1.0           153500.0      60.0          14.7
0.0           0.0           0.0           0.0           0.0
0.0
17.0          1.0           153500.0      61.0          14.7
0.0           0.0           0.0           0.0           0.0
0.0
18.0          1.0           95961.0       62.0          44.1
0.30303       0.69697       0.0           0.0           0.
0.0
19.0          11.0          3087.0        0.0           0.0
0.0           0.0           0.0           0.0           0.0
0.0
20.0          11.0          50.0          0.0           0.0
0.0           0.0           0.0           0.0           0.0
0.0
21.0          1.0           3712.0        62.0          44.1
0.0           0.0           0.0           0.0           1.0
0.0
22.0          11.0          9982.22       0.0           0.0
0.0           0.0           0.0           0.0           0.0
0.0
23.0          11.0          50.0          0.0           0.0
0.0           0.0           0.0           0.0           0.0
0.0
24.0          1.0           92249.0       62.0          44.1
0.30303       0.69697       0.0           0.0           0.
0.0
25.0          1.0           649.0         100.0         44.1
0.1261        0.3992        0.0255        0.3615        0.0
0.0877
26.0          1.0           649.0         94.0          270.0
0.1261        0.3992        0.0255        0.3615        0.0
0.0877
27.0          1.0           649.0         89.0          242.0
0.1261        0.3992        0.0255        0.3615        0.0
0.0877
28.0          11.0          0.0           80.0          0.0
0.0           0.0           0.0           0.0           0.0
0.0
```

```
            29.0        11.0        50.0        0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            30.0        1.0         649.0       89.0        242.0
            0.1261      0.3992      0.0255      0.3615      0.0
            0.0877
            31.0        11.0        50.0        0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            32.0        11.0        50.0        0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            33.0        1.0         33.0        89.0        242.0
            0.1261      0.3992      0.0255      0.3615      0.0
            0.0877
            34.0        1.0         649.0       121.0       350.0
            0.1261      0.3992      0.0255      0.3615      0.0
            0.0877
            35.0        1.0         1265.0      121.0       350.0
            0.1261      0.3992      0.0255      0.3615      0.0
            0.0877
            36.0        11.0        80.0        0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            37.0        11.0        50.0        0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            38.0        1.0         616.0       121.0       350.0
            0.1261      0.3992      0.0255      0.3615      0.0
            0.0877
            39.0        9.0         487.0       344.35      125.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            40.0        9.0         7682500.0   0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            0.0
            41.0        1.0         87487.0     100.0       44.1
            .1329111754 .4209654006 .0268382730 .3811766320 0.0
            .0381085190
            42.0        1.0         39376.0     100.0       44.1
            .1329111754 .4209654006 .0268382730 .3811766320 0.0
            .0381085190
END
PROPERTIES  20.0
            100.0       60.0        0.0         0.0         0.0
            62.24       -123916.59  2096.2      1.0         14250.0
            60.0        0.0         0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
            100.0       60.0        0.0         0.0         0.0
            62.24       -123916.59  2096.2      1.0         14250.0
            60.0        0.0         0.0         0.0         0.0
            0.0         0.0         0.0         0.0         0.0
```

```
200.0          60.0          0.0           0.0           0.0
62.24          -123916.59    2096.2        1.0           14250.0
60.0           0.0           0.0           0.0           0.0
0.0            0.0           0.0           0.0           0.0
200.0          60.0          0.0           0.0           0.0
62.24          -123916.59    2096.2        1.0           14250.0
60.0           0.0           0.0           0.0           0.0
0.0            0.0           0.0           0.0           0.0
300.0          60.0          0.0           0.0           0.0
62.24          -123916.59    2096.2        1.0           14250.0
60.0           0.0           0.0           0.0           0.0
0.0            0.0           0.0           0.0           0.0
100.0          60.0          0.0           0.0           0.0
62.933147      -123916.59    2096.2        1.0           14250.0
60.0           0.0           0.0           0.0           0.0
0.0            0.0           0.0           0.0           0.0
END
    END
```

APPENDIX I: PROGRAM LISTINGS

A listing is given in Figure I.1 of the DYNSYS 2.0 executive program except for SUBROUTINES TRGB and TRGB2. These routines were made available to the author for research purposes, but are considered proprietary by their creators. Copies of these may be obtained from:

MR. H.P. HUTCHISON
LECTURER
DEPT. OF CHEMICAL ENGINEERING
CAMBRIDGE UNIVERSITY
CAMBRIDGE, ENGLAND

A fee will probably be required from commercial users.

I.1 The DYNSYS 2.1 Executive Program

Since version 2.0 is not completely available, another version of DYNSYS, version 2.1, was created to have a sparse matrix oriented package which was completely available to the public. Key's subroutine SIMULT was used in place of TRGB. Version 2.1 is not as efficient as 2.0 since Key's program is not designed for repeated solution of linear equations. The new version, however, does illustrate the usage of DYNSYS and can be keypunched and

used by anyone.

There are some slight differences in the writings of the modules. A skeleton module is given (Figure I.2) and also a listing of the executive and a sample module (Figure I.3) which corresponds to the second example of Chapter 8 (Figure 8.7), a CSTR with first order reversible reaction. The only executive subroutines given in Figure I.3 are DYN1, DYN2, DIFSUB, SIMULT and RITE. These replace subroutines DYN1, DYN2, DIFSUB, TRGB and TRGB2 in DYNSYS 2.0. Subroutines GET, FETCH, PROPS, TEMP, TRIDAG and GRAPH are identical with those in DYNSYS 2.0. The mainline program and subroutine OUTPUT are identical with DYNSYS 2.0 except for the COMMON/GEAR2/.

The differences in writing of modules are as follows. The Jacobian is stored as described in Section 5.1.2.

COMMON/ROW/IROW(4) is not required.

COMMON/COLUMN/JCOL(4) and COMMON/JACOB/XJACOB(4) are replaced by

COMMON/COLUMN/JCOL(10,2) and COMMON/JACOB(10,2)

i.e. the column numbers and nonzero elements are now stored as two dimensional arrays. Note, however, that the row dimension of JCOL and XJACOB must be the same as

in DIFSUB.

COMMON/MODULE/IDERY,ITER,ITR1,NZERO is replaced by

COMMON/MODULE/IDERY,ITER,ITR1,MC,IPIVOT

NZERO is no longer required, but the user must supply, in SECTION #2 of the skeleton, values of MC and IPIVOT. MC is the maximum number of columns in the XJACOB matrix where the nonzero Jacobian elements are stored. This value can become larger than the maximum number of nonzeros in any Jacobian row since additional nonzero elements can be created during the solution. MC should be constant for any module and may have to be determined by trial and error. It is also a function of the pivot option. If MC is too small, the program will stop and print the message

MC SHOULD BE AT LEAST ___

MC should then be set to at least the number in the blank.

IPIVOT is the pivot option (from 1-7) used for the module. The options are described in the skeleton and in section 5.1.2.

The remainder of the module is written exactly as for DYNSYS 2.0. The only change in the data set is that MINPIVOT is not used.

The difference in execution time between version 2.0 and 2.1 will depend on the size of the system. For very small examples the times will be nearly the same. For large systems, Chapter 5 gives execution times for SIMULT and TRGB-TRGB2 for 50, 100 and 200 equations. For example for 50 equations, bandwidth 9, the best option of Key's method took 0.691 seconds while TRGB2 took 0.049 seconds. Since the linear equations must be solved at each Newton-Raphson iteration of the corrector, the time difference could become substantial for a long simulation with many time steps. For 500 time steps with an average of two Newton-Raphson iterations per step, the execution time would differ by over a minute.

```
C
C     FIGURE I.1: LISTING OF DYNSYS 2.0 EXECUTIVE
C
C
      PROGRAM DYNSYS(INPUT,OUTPUT,PUNCH,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7=PDYN    1
     1UNCH)                                                            DYN    2
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                DYN    3
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH    DYN    4
      COMMON /PLT/ NPLOTS,PLOTI,PLOTS(20,4),PLOTT                      DYN    5
      COMMON /LAG/ NSX,NSW                                             DYN    6
      COMMON /FTN/ FN(1,1),NFN                                         DYN    7
      COMMON /GEAR1/ Y(8,222),SAVE(10,222),ERROR(222),YMAX(222),DERY1(22DYN    8
     12),DERY2(222)                                                    DYN    9
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF   DYN   10
      COMMON /G/ NT,NV,TPLOT(602),V(602),VPLOT(1,602)                  DYN   11
1     CALL DYN1                                                        DYN   12
2     PLOTT=0.0                                                        DYN   13
      DO 3 I=1,222                                                     DYN   14
      DERY2(I)=0.0                                                     DYN   15
      YMAX(I)=1.0                                                      DYN   16
3     CONTINUE                                                         DYN   17
      JSTART=0                                                         DYN   18
      NBVMAX=0                                                         DYN   19
      OTIME=TIME                                                       DYN   20
      NSW=0                                                            DYN   21
      IF (NPOL.GT.0) GO TO 4                                           DYN   22
      GO TO 5                                                          DYN   23
4     CONTINUE                                                         DYN   24
      READ (5,25) X                                                    DYN   25
      NPLOTS=X                                                         DYN   26
      READ (5,25) PLOTI                                                DYN   27
      READ (5,25) ((PLOTS(I,J),J=1,4),I=1,NPLOTS)                      DYN   28
      PRINT 25, ((PLOTS(I,J),J=1,4),I=1,NPLOTS)                        DYN   29
5     CONTINUE                                                         DYN   30
C     SET UP INITIAL VALUES FOR GRAPH VECTOR COUNT                     DYN   31
      NV=0                                                             DYN   32
      NT=0                                                             DYN   33
      CALL OUTPUT                                                      DYN   34
      KPRINT=0                                                         DYN   35
6     CONTINUE                                                         DYN   36
      TIME=OTIME+H                                                     DYN   37
      KFLAG=1                                                          DYN   38
      ICONV=0                                                          DYN   39
      KPRINT=KPRINT+1                                                  DYN   40
      IG=2                                                             DYN   41
      CALL DYN2                                                        DYN   42
      IG=1                                                             DYN   43
      CALL DYN2                                                        DYN   44
      IF (NPOL.GT.0) GO TO 7                                           DYN   45
      IF (KPRINT.LT.NPR) GO TO 8                                       DYN   46
7     CONTINUE                                                         DYN   47
      CALL OUTPUT                                                      DYN   48
```

```
      KPRINT=0                                                          DYN  49
8     CONTINUE                                                          DYN  50
      NSW=1                                                             DYN  51
      IF (ICONV.EQ.1.OR.KFLAG.NE.1) GO TO 6                             DYN  52
      NSW=0                                                             DYN  53
      OTIME=TIME                                                        DYN  54
C                                                                       DYN  55
C     UPDATE STREAM MATRIX                                              DYN  56
C                                                                       DYN  57
      DO 10 I=1,NS                                                      DYN  58
      DO 9 J=3,NC5                                                      DYN  59
      S(2,I,J)=S(1,I,J)                                                 DYN  60
9     CONTINUE                                                          DYN  61
10    CONTINUE                                                          DYN  62
      IF (TIME.LE.TMAX) GO TO 6                                         DYN  63
      WRITE (6,24)                                                      DYN  64
      CALL OUTPUT                                                       DYN  65
      IF (NGRAPH.NE.1) GO TO 11                                         DYN  66
      CALL GRAPH (NPLOTS)                                               DYN  67
11    CONTINUE                                                          DYN  68
      CALL GET (N,X,0)                                                  DYN  69
      IF (N.EQ.3HEND) GO TO 19                                          DYN  70
      IF (N.EQ.6HREPEAT) GO TO 1                                        DYN  71
      IF (N.EQ.8HCONTINUE) GO TO 12                                     DYN  72
      PRINT 20                                                          DYN  73
      STOP                                                              DYN  74
12    H=0.00001                                                         DYN  75
13    CALL GET (N,X,1)                                                  DYN  76
      IF (N.EQ.4HTIME) TMAX=X                                           DYN  77
      IF (N.EQ.8HLINEPLOT) NPOL=X                                       DYN  78
      IF (N.EQ.5HGRAPH) NGRAPH=X                                        DYN  79
      IF (N.EQ.6HALTERS) GO TO 14                                       DYN  80
      IF (N.EQ.6HALTERE) GO TO 16                                       DYN  81
      IF (N.EQ.8HFUNCTION) GO TO 17                                     DYN  82
      IF (N.EQ.3HEND) GO TO 2                                           DYN  83
      GO TO 13                                                          DYN  84
14    NOS=X                                                             DYN  85
      READ 21, (S(1,NOS,J),J=1,5)                                       DYN  86
      PRINT 23, (S(1,NOS,J),J=1,5)                                      DYN  87
      READ 22, (S(1,NOS,J),J=6,NC5)                                     DYN  88
      PRINT 23, (S(1,NOS,J),J=1,NC5)                                    DYN  89
      DO 15 J=1,NC5                                                     DYN  90
      S(2,NOS,J)=S(1,NOS,J)                                             DYN  91
15    CONTINUE                                                          DYN  92
      GO TO 13                                                          DYN  93
16    NOEP=X                                                            DYN  94
      READ 21, (EP(NOEP,J),J=1,5)                                       DYN  95
      PRINT 23, (EP(NOEP,J),J=1,5)                                      DYN  96
      GO TO 13                                                          DYN  97
17    NFN=X                                                             DYN  98
      DO 18 I=1,NFN                                                     DYN  99
      READ 21, (FN(I,J),J=1,5)                                          DYN 100
```

```
      READ 22, (FN(I,J),J=6,NC5)                                        DYN 101
18    CONTINUE                                                          DYN 102
      PRINT 23, ((FN(I,J),J=1,NC5),I=1,NFN)                             DYN 103
      GO TO 13                                                          DYN 104
19    STOP                                                              DYN 105
C                                                                       DYN 106
20    FORMAT (41H LAST CARD MUST BE END,REPEAT OR CONTINUE)             DYN 107
21    FORMAT (12X,5F12.2)                                               DYN 108
22    FORMAT (12X,4F12.2)                                               DYN 109
23    FORMAT (1X,9F10.3)                                                DYN 110
24    FORMAT (/20X,23H ***TMAX APPROACHED*** )                          DYN 111
25    FORMAT (4F12.4)                                                   DYN 112
      END                                                               DYN 113
```

```
      SUBROUTINE DYN1                                                   DY1    1
C     DATA LOADING AND PRINTING                                         DY1    2
      COMMON /UNIT/ IM,NMP                                              DY1    3
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                 DY1    4
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH     DY1    5
      COMMON /FTN/ FN(1,1),NFN                                          DY1    6
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF    DY1    7
      COMMON /GEAR3/ HMIN,HMAX,XMIN,NOMES                               DY1    8
      DIMENSION AM(100)                                                 DY1    9
      DIMENSION ITAG(30)                                                DY1   10
      DATA ITAG/5HVALV1,5HVALV2,5HSTIR1,5HCONT1,5HSETL1,6HMXPLT1,5HSETT1DY1   11
     1,6HVESSL1,5HDELAY,5HCSTR1,5HEXCH1,5HDCAN1,5HCONV1,5HSPLT1,16*6H   DY1   12
     2   /                                                              DY1   13
      WRITE (6,27)                                                      DY1   14
C     DEFAULT VALUES                                                    DY1   15
      TIME=0.0                                                          DY1   16
      MAXNE=35                                                          DY1   17
      NPR=1                                                             DY1   18
      TMAX=10.0                                                         DY1   19
      NCOMP=1                                                           DY1   20
      NMP=5                                                             DY1   21
      H=0.00001                                                         DY1   22
      HMIN=0.000001                                                     DY1   23
      HMAX=0.05                                                         DY1   24
      EPS=0.001                                                         DY1   25
      NPOL=0                                                            DY1   26
      NGRAPH=0                                                          DY1   27
      IORDER=6                                                          DY1   28
      ISTIFF=1                                                          DY1   29
      NOMES=0                                                           DY1   30
      XMIN=1.0E-06                                                      DY1   31
C     READ AND COPY TITLE                                               DY1   32
1     READ (5,29) (AM(I),I=1,8)                                         DY1   33
      WRITE (6,28) (AM(I),I=1,8)                                        DY1   34
      IF (AM(1).NE.5HBEGIN) GO TO 1                                     DY1   35
C     READ INITIAL DATA                                                 DY1   36
2     CALL GET (N,X,1)                                                  DY1   37
      IF (N.EQ.5HCOMPS) NCOMP=X                                         DY1   38
      IF (N.EQ.6HIN/OUT) NMP=X                                          DY1   39
      IF (N.EQ.6HDELTAT) H=X                                            DY1   40
      IF (N.EQ.4HHMIN) HMIN=X                                           DY1   41
      IF (N.EQ.4HHMAX) HMAX=X                                           DY1   42
      IF (N.EQ.4HTIME) TMAX=X                                           DY1   43
      IF (N.EQ.8HPRINTING) NPR=X                                        DY1   44
      IF (N.EQ.9HTOLERANCE) EPS=X                                       DY1   45
      IF (N.EQ.5HORDER) IORDER=X                                        DY1   46
      IF (N.EQ.8HLINEPLOT) NPOL=1                                       DY1   47
      IF (N.EQ.5HGRAPH) NGRAPH=1                                        DY1   48
      IF (N.EQ.8HNONSTIFF) ISTIFF=0                                     DY1   49
      IF (N.EQ.9HNOMESSAGE) NOMES=1                                     DY1   50
      IF (N.EQ.8HMINPIVOT) XMIN=X                                       DY1   51
      IF (N.EQ.7HLIBRARY) GO TO 3                                       DY1   52
```

```
      IF (N.EQ.7HPROCESS) GO TO 6                                       DY1  53
      IF (N.EQ.8HFUNCTION) GO TO 5                                      DY1  54
      GO TO 2                                                           DY1  55
3     NLIB=X                                                            DY1  56
      DO 4 I=1,NLIB                                                     DY1  57
      CALL GET (N,X,0)                                                  DY1  58
      J=X                                                               DY1  59
      ITAG(J)=N                                                         DY1  60
4     CONTINUE                                                          DY1  61
      GO TO 2                                                           DY1  62
5     NFN=X                                                             DY1  63
      NC5=NCOMP+5                                                       DY1  64
      READ 39, ((FN(I,J),J=1,NC5),I=1,NFN)                              DY1  65
      PRINT 39, ((FN(I,J),J=1,NC5),I=1,NFN)                             DY1  66
      GO TO 2                                                           DY1  67
6     CONTINUE                                                          DY1  68
      NMP=NMP+2                                                         DY1  69
      MAXNMP=NMP+1                                                      DY1  70
      NC5=NCOMP+5                                                       DY1  71
      WRITE (6,30)                                                      DY1  72
      DO 9 I=1,MAXNE                                                    DY1  73
      DO 7 J=1,MAXNMP                                                   DY1  74
      MP(I,J)=0                                                         DY1  75
7     CONTINUE                                                          DY1  76
      DO 8 J=1,10                                                       DY1  77
      EP(I,J)=0.0                                                       DY1  78
8     CONTINUE                                                          DY1  79
9     CONTINUE                                                          DY1  80
      NE=0                                                              DY1  81
      NEX=1                                                             DY1  82
      TIME=0.0                                                          DY1  83
10    CALL GET (NM,X,0)                                                 DY1  84
11    IF (NM.EQ.3HEND) GO TO 16                                         DY1  85
      DO 12 I=1,30                                                      DY1  86
      IF (NM.EQ.ITAG(I)) GO TO 13                                       DY1  87
12    CONTINUE                                                          DY1  88
      WRITE (6,31) NM                                                   DY1  89
      I=0                                                               DY1  90
13    NT=I                                                              DY1  91
      NE=NE+1                                                           DY1  92
      READ (5,32) (AM(I),I=3,NMP)                                       DY1  93
      DO 14 J=3,NMP                                                     DY1  94
      MP(NE,J)=AM(J)                                                    DY1  95
14    CONTINUE                                                          DY1  96
      MP(NE,1)=X                                                        DY1  97
      MP(NE,2)=NT                                                       DY1  98
      WRITE (6,33) (MP(NE,I),I=1,NMP)                                   DY1  99
      READ (5,32) (EP(NE,I),I=1,10)                                     DY1 100
      WRITE (6,34) (EP(NE,I),I=1,10)                                    DY1 101
      CALL GET (NM,X,0)                                                 DY1 102
      IF (NM.NE.5HEXTRA) GO TO 15                                       DY1 103
      NNX=X                                                             DY1 104
```

```
      NMX=NNX+NEX-1                                                    DY1 105
      READ (5,32) (EX(J),J=NEX,NMX)                                    DY1 106
      WRITE (6,34) (EX(J),J=NEX,NMX)                                   DY1 107
      MP(NE,NMP+1)=NEX                                                 DY1 108
      NEX=NEX+NNX                                                      DY1 109
      CALL GET (NM,X,1)                                                DY1 110
15    IF (NM.NE.7HRESERVE) GO TO 11                                    DY1 111
      NNX=X                                                            DY1 112
      NEX=NEX+NNX                                                      DY1 113
      GO TO 10                                                         DY1 114
16    CALL FETCH (7HSTREAMS,NS)                                        DY1 115
      WRITE (6,35) NS                                                  DY1 116
      DO 18 I=1,NS                                                     DY1 117
      S(1,I,1)=I                                                       DY1 118
      S(1,I,2)=-1.0                                                    DY1 119
      S(1,I,3)=0.0                                                     DY1 120
      S(1,I,4)=530.0                                                   DY1 121
      S(1,I,5)=14.7                                                    DY1 122
      DO 17 J=6,NC5                                                    DY1 123
      S(1,I,J)=0                                                       DY1 124
17    CONTINUE                                                         DY1 125
18    CONTINUE                                                         DY1 126
19    CALL GET (NM,X,-1)                                               DY1 127
      IF (NM.EQ.8HEXPLICIT) GO TO 20                                   DY1 128
      IF (NM.EQ.3HEND) GO TO 25                                        DY1 129
      IF (NM.EQ.4HSPEC) GO TO 22                                       DY1 130
      GO TO 19                                                         DY1 131
20    READ (5,36) IT,(AM(I),I=1,5)                                     DY1 132
      IF (IT.EQ.3HEND) GO TO 25                                        DY1 133
      READ (5,32) (AM(I),I=6,NC5)                                      DY1 134
      N=AM(1)                                                          DY1 135
      DO 21 I=1,NC5                                                    DY1 136
      S(1,N,I)=AM(I)                                                   DY1 137
21    CONTINUE                                                         DY1 138
      GO TO 20                                                         DY1 139
22    N1=X                                                             DY1 140
      READ (5,32) (AM(I),I=1,NC5)                                      DY1 141
      N2=AM(1)                                                         DY1 142
      DO 24 I=N1,N2                                                    DY1 143
      DO 23 J=2,NC5                                                    DY1 144
      S(1,I,J)=AM(J)                                                   DY1 145
23    CONTINUE                                                         DY1 146
      S(1,I,1)=I                                                       DY1 147
24    CONTINUE                                                         DY1 148
      GO TO 19                                                         DY1 149
25    DO 26 I=1,NS                                                     DY1 150
      K=S(1,I,2)                                                       DY1 151
      WRITE (6,38) I,K,(S(1,I,J),J=3,5)                                DY1 152
      WRITE (6,37) (S(1,I,J),J=6,NC5)                                  DY1 153
      DO 26 J=1,NC5                                                    DY1 154
26    S(2,I,J)=S(1,I,J)                                                DY1 155
      XX=PROPS(0,0)                                                    DY1 156
```

```
      RETURN                                                            DY1 157
C                                                                       DY1 158
27    FORMAT (1H1)                                                      DY1 159
28    FORMAT (1X,8A10)                                                  DY1 160
29    FORMAT (8A10)                                                     DY1 161
30    FORMAT (1H1)                                                      DY1 162
31    FORMAT (18H ***FAIL***  NAME ,A12,8H NOT SET)                     DY1 163
32    FORMAT (12X,5F12.0)                                               DY1 164
33    FORMAT (/,9H   UNIT =,I3,9H=  TYPE =,I3,1H=,12I5)                 DY1 165
34    FORMAT (11X,5F14.5)                                               DY1 166
35    FORMAT (1H1,I6,8H STREAMS)                                        DY1 167
36    FORMAT (A3,9X,5F12.0)                                             DY1 168
37    FORMAT (8X,4F14.5)                                                DY1 169
38    FORMAT (11H   STREAM =,I3,4H=  (,I3,1H),3F14.5)                   DY1 170
39    FORMAT (12X,5F12.2)                                               DY1 171
      END                                                               DY1 172
```

```
      SUBROUTINE GET (NAME,X,IFG)                                        GET    1
1     READ (5,6) NAME,X                                                  GET    2
      IF (NAME.NE.10H          ) GO TO 2                                 GET    3
      IF (X.EQ.0.OR.X.EQ.-0) GO TO 1                                     GET    4
      WRITE (6,7)                                                        GET    5
      GO TO 1                                                            GET    6
2     CONTINUE                                                           GET    7
      IF (X.LE.0) GO TO 3                                                GET    8
      IF (IFG) 3,4,5                                                     GET    9
3     WRITE (6,9) NAME                                                   GET   10
      RETURN                                                             GET   11
4     N=X                                                                GET   12
      WRITE (6,8) NAME,N                                                 GET   13
      RETURN                                                             GET   14
5     WRITE (6,9) NAME,X                                                 GET   15
      RETURN                                                             GET   16
C                                                                        GET   17
6     FORMAT (A10,F14.0)                                                 GET   18
7     FORMAT (48H ---WARN---  DATA SKIPPED WHILE READING KEYWORDS)       GET   19
8     FORMAT (1X,A10,I12)                                                GET   20
9     FORMAT (1X,A10,F14.5)                                              GET   21
      END                                                                GET   22
```

```
      SURROUTINE FETCH (IWORD,I)                                        FET    1
1     READ (5,2) IWORDA,AI                                              FET    2
      IF (IWORDA.NE.IWORD) GO TO 1                                      FET    3
      I=AI                                                              FET    4
      RETURN                                                            FET    5
C                                                                       FET    6
2     FORMAT (A10,F14.2)                                                FET    7
      END                                                               FET    8
```

```
      SUBROUTINE DYN2                                                    DY2    1
      COMMON /UNIT/ IM,NMP                                               DY2    2
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                  DY2    3
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      DY2    4
      COMMON /BKV/ NBV                                                   DY2    5
      COMMON /CNTR/ NCTR                                                 DY2    6
      COMMON /LAG/ NSX,NSW                                               DY2    7
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF     DY2    8
      NBV=0                                                              DY2    9
      NSX=0                                                              DY2   10
      NOPPT=0                                                            DY2   11
      NCTR=0                                                             DY2   12
      DO 31 IM=1,NE                                                      DY2   13
      IF (ICONV.EQ.1) RETURN                                             DY2   14
      NTYPE=MP(IM,2)                                                     DY2   15
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23DY2   16
     1,24,25,26,27,28,29,30), NTYPE                                      DY2   17
1     CALL TYPE1                                                         DY2   18
      GO TO 31                                                           DY2   19
2     CALL TYPE2                                                         DY2   20
      GO TO 31                                                           DY2   21
3     CALL TYPE3                                                         DY2   22
      GO TO 31                                                           DY2   23
4     CALL TYPE4                                                         DY2   24
      GO TO 31                                                           DY2   25
5     CALL TYPE5                                                         DY2   26
      GO TO 31                                                           DY2   27
6     CALL TYPE6                                                         DY2   28
      GO TO 31                                                           DY2   29
7     CALL TYPE7                                                         DY2   30
      GO TO 31                                                           DY2   31
8     CALL TYPE8                                                         DY2   32
      GO TO 31                                                           DY2   33
9     CALL TYPE9                                                         DY2   34
      GO TO 31                                                           DY2   35
10    CALL TYPE10                                                        DY2   36
      GO TO 31                                                           DY2   37
11    CALL TYPE11                                                        DY2   38
      GO TO 31                                                           DY2   39
12    CALL TYPE12                                                        DY2   40
      GO TO 31                                                           DY2   41
13    CALL TYPE13                                                        DY2   42
      GO TO 31                                                           DY2   43
14    CALL TYPE14                                                        DY2   44
      GO TO 31                                                           DY2   45
15    CALL TYPE15                                                        DY2   46
      GO TO 31                                                           DY2   47
16    CALL TYPE16                                                        DY2   48
      GO TO 31                                                           DY2   49
17    CALL TYPE17                                                        DY2   50
      GO TO 31                                                           DY2   51
18    CALL TYPE18                                                        DY2   52
```

```
      GO TO 31                                                          DY2  53
19    CALL TYPE19                                                       DY2  54
      GO TO 31                                                          DY2  55
20    CALL TYPE20                                                       DY2  56
      GO TO 31                                                          DY2  57
21    CALL TYPE21                                                       DY2  58
      GO TO 31                                                          DY2  59
22    CALL TYPE22                                                       DY2  60
      GO TO 31                                                          DY2  61
23    CALL TYPE23                                                       DY2  62
      GO TO 31                                                          DY2  63
24    CALL TYPE24                                                       DY2  64
      GO TO 31                                                          DY2  65
25    CALL TYPE25                                                       DY2  66
      GO TO 31                                                          DY2  67
26    CALL TYPE26                                                       DY2  68
      GO TO 31                                                          DY2  69
27    CALL TYPE27                                                       DY2  70
      GO TO 31                                                          DY2  71
28    CALL TYPE28                                                       DY2  72
      GO TO 31                                                          DY2  73
29    CALL TYPE29                                                       DY2  74
      GO TO 31                                                          DY2  75
30    CALL TYPE30                                                       DY2  76
31    CONTINUE                                                          DY2  77
      RETURN                                                            DY2  78
      END                                                               DY2  79
```

```
      SUBROUTINE OUTPUT                                                  OUT    1
C     THIS ROUTINE PRINTS OUT THE RESULTS                                OUT    2
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                  OUT    3
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      OUT    4
      COMMON /PLT/ NPLOTS,PLOTI,PLOTS(20,4),PLOTT                        OUT    5
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF     OUT    6
      COMMON /FTN/ FN(1,1),NFN                                           OUT    7
      COMMON /FF/ TITLE(11),MPLOT(20,2),PLOTR(20,2)                      OUT    8
      COMMON /G/ NT,NV,TPLOT(602),V(602),VPLOT(1,602)                    OUT    9
      DIMENSION SYMBOL(15), SYMBOL1(4), ALINE(125)                       OUT   10
      DATA SYMBOL/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL,1HM,1HOUT   11
     1N,1HO/                                                             OUT   12
      DATA SYMBOL1/1H.,1H-,1H ,1H,/                                      OUT   13
      DATA TITLE/4HSTRM,4HFLAG,4HFLOW,4HTEMP,4HPRES,5HCOMP1,5HCOMP2,5HCOOUT   14
     1MP3,5HCOMP4,5HCOMP5,5HCOMP6/                                       OUT   15
      IF (NPOL.GT.0) GO TO 3                                             OUT   16
      WRITE (6,14) TIME                                                  OUT   17
      WRITE (6,15)                                                       OUT   18
      DO 1 I=1,NS                                                        OUT   19
      IF (S(2,I,2).LT.0.0) GO TO 1                                       OUT   20
      NN=S(2,I,1)                                                        OUT   21
      WRITE (6,17) NN,(S(1,I,J),J=3,NC5)                                 OUT   22
1     CONTINUE                                                           OUT   23
      DO 2 I=1,NE                                                        OUT   24
      IF (MP(I,1).GT.0) GO TO 2                                          OUT   25
      K=IABS(MP(I,1))                                                    OUT   26
      WRITE (6,16) K,(EP(I,J),J=1,5)                                     OUT   27
2     CONTINUE                                                           OUT   28
      RETURN                                                             OUT   29
3     CONTINUE                                                           OUT   30
      IF (JSTART.NE.0) GO TO 6                                           OUT   31
C     START PLOTTING RESULTS                                             OUT   32
      WRITE (6,19)                                                       OUT   33
      DO 4 I=1,NPLOTS                                                    OUT   34
      N1=PLOTS(I,1)                                                      OUT   35
      N2=PLOTS(I,2)                                                      OUT   36
      MPLOT(I,1)=N1                                                      OUT   37
      MPLOT(I,2)=N2                                                      OUT   38
      PLOTR(I,1)=PLOTS(I,3)                                              OUT   39
      PLOTR(I,2)=PLOTS(I,4)                                              OUT   40
      WRITE (6,20) SYMBOL(I),N1,TITLE(N2),PLOTS(I,3),PLOTS(I,4)          OUT   41
4     CONTINUE                                                           OUT   42
      X=0.                                                               OUT   43
      DO 5 I=1,21                                                        OUT   44
      ALINE(I)=X                                                         OUT   45
      X=X+5.                                                             OUT   46
5     CONTINUE                                                           OUT   47
      WRITE (6,21) (ALINE(I),I=1,21)                                     OUT   48
      WRITE (6,22)                                                       OUT   49
6     CONTINUE                                                           OUT   50
C     CALCULATE NUMBER OF LINES TO SKIP                                  OUT   51
      XSPACE=(TIME-PLOTT)/PLOTI                                          OUT   52
```

```
      NSPACE=XSPACE                                                          OUT  53
      XXSPACE=NSPACE                                                         OUT  54
      IF ((XSPACE-XXSPACE).GT.0.5) NSPACE=NSPACE+1                           OUT  55
      IF (JSTART.EQ.0) NSPACE=2                                              OUT  56
      IF (NSPACE.LT.1) RETURN                                                OUT  57
C     SET UP ALINE                                                           OUT  58
      DO 7 I=2,120                                                           OUT  59
      ALINE(I)=SYMBOL1(3)                                                    OUT  60
7     CONTINUE                                                               OUT  61
      ALINE(1)=SYMBOL1(1)                                                    OUT  62
      ALINE(101)=SYMBOL1(1)                                                  OUT  63
      PLOTT=TIME                                                             OUT  64
      NSPACE=NSPACE-1                                                        OUT  65
      IF (NSPACE.EQ.0) GO TO 9                                               OUT  66
      DO 8 I=1,NSPACE                                                        OUT  67
      WRITE (6,18) (ALINE(J),J=1,101)                                        OUT  68
8     CONTINUE                                                               OUT  69
C     SET UP NEW LINE()                                                      OUT  70
9     N5=1                                                                   OUT  71
      N6=NPLOTS                                                              OUT  72
      NV=NV+1                                                                OUT  73
      IF (NV.GT.602.OR.NV.LT.1) PRINT 24                                     OUT  74
      DO 13 I=N5,N6                                                          OUT  75
      N1=PLOTS(I,1)                                                          OUT  76
      N2=PLOTS(I,2)                                                          OUT  77
      XN2=(S(1,N1,N2)-PLOTS(I,3))*100./(PLOTS(I,4)-PLOTS(I,3))               OUT  78
      VPLOT(I,NV)=XN2                                                        OUT  79
      XN3=XN2+1.                                                             OUT  80
      N3=XN3                                                                 OUT  81
      XXN3=N3                                                                OUT  82
      IF ((XN3-XXN3).GT.0.5) N3=N3+1                                         OUT  83
      IF (N3.LT.2.OR.N3.GT.100) GO TO 13                                     OUT  84
      IF (ALINE(N3).NE.SYMBOL1(3)) GO TO 10                                  OUT  85
      ALINE(N3)=SYMBOL(I)                                                    OUT  86
      GO TO 13                                                               OUT  87
10    DO 11 J=102,108                                                        OUT  88
      IF (ALINE(J).EQ.SYMBOL1(3)) GO TO 12                                   OUT  89
11    CONTINUE                                                               OUT  90
12    J=109                                                                  OUT  91
      ALINE(J)=ALINE(N3)                                                     OUT  92
      ALINE(J+1)=SYMBOL(I)                                                   OUT  93
      ALINE(J+2)=SYMBOL1(4)                                                  OUT  94
13    CONTINUE                                                               OUT  95
      WRITE (6,23) TIME,SYMBOL1(2),(ALINE(J),J=1,110)                        OUT  96
      NT=NT+1                                                                OUT  97
      IF (NT.GT.602.OR.NV.LT.1) PRINT 24                                     OUT  98
      TPLOT(NT)=PLOTT                                                        OUT  99
      RETURN                                                                 OUT 100
C                                                                            OUT 101
14    FORMAT (////20X,5(1HS),5X,30HPROCESS VARIABLES AT TIME =    ,E11.5,    OUT 102
     15X,5(1HS)/)                                                            OUT 103
15    FORMAT (61H STREAM    TOT FLOW    TEMP    PRESS    COMP 1    COMP 2   OUT 104
```

```
     1COMP 3,/)                                                                    OUT 105
  16  FORMAT (1H0,5X,I3,5X,6(3X,F12.5))                                            OUT 106
  17  FORMAT (I6,F12.3,2F8.1,9F9.5)                                                OUT 107
  18  FORMAT (19X,101A1)                                                           OUT 108
  19  FORMAT (1H1,20X,6HSYMBOL,11X,6HSTREAM,23X,8HVARIABLE,19X,6HRANGES)           OUT 109
  20  FORMAT (1H0,22X,A2,13X,I3,28X,A7,3X,2(2X,F12.5))                             OUT 110
  21  FORMAT (////15X,21(1X,F4.0))                                                 OUT 111
  22  FORMAT (14X,5(1H.),21(5HI....))                                              OUT 112
  23  FORMAT (10X,F6.3,2X,112A1)                                                   OUT 113
  24  FORMAT (1X,25H NT,NV,OR N3 OUT OF RANGE)                                     OUT 114
      END                                                                          OUT 115-
```

```
      SUBROUTINE GRAPH (NPLOTS)                                          GRA   1
      COMMON /UNIT/ IM,NMP                                               GRA   2
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      GRA   3
      COMMON /G/ NT,NV,TPLOT(602),V(602),VPLOT(1,602)                    GRA   4
      COMMON /FF/ TITLE(11),MPLOT(20,2),PLOTR(20,2)                      GRA   5
      V(NT+1)=0.0                                                        GRA   6
      V(NT+2)=12.5                                                       GRA   7
C     AAAA=X,AX IS ACTUAL INCHES                                         GRA   8
      AAAA=10.                                                           GRA   9
      TPLOT(NT+1)=0.0                                                    GRA  10
      TPLOT(NT+2)=TMAX/AAAA                                              GRA  11
      CALL PLOTS (60,24.,10.75,2)                                        GRA  12
      CALL PLOT (0.5,0.50,-3)                                            GRA  13
C                                                                        GRA  14
C     SET UP AXIS                                                        GRA  15
      CALL AXIS3 (0.0,0.0,9HTIME(HRS),-9,AAAA,0.0,TPLOT(NT+1),TPLOT(NT+2GRA  16
     1),1,0.5,2)                                                         GRA  17
      CALL AXIS3 (0.0,0.0,6HV-AXIS,6,8.00,90.0,V(NT+1),V(NT+2),1,0.4,2) GRA  18
C                                                                        GRA  19
C     PLOT K=1,NPLOTS CURVES,WITH DIFFERENT SYMBOLS                      GRA  20
      DO 2 K=1,NPLOTS                                                    GRA  21
      DO 1 J=1,NT                                                        GRA  22
      V(J)=VPLOT(K,J)                                                    GRA  23
1     CONTINUE                                                           GRA  24
      CALL LINE3 (TPLOT,V,NT,1,2,K,0)                                    GRA  25
2     CONTINUE                                                           GRA  26
C                                                                        GRA  27
C     WRITE HEADING                                                      GRA  28
      XD=11.5                                                            GRA  29
      YD=7.0                                                             GRA  30
      CALL SYMBOL (XD,YD,0.2,7HLEGEND ,0.,7)                             GRA  31
      XD=10.5                                                            GRA  32
      YD=YD-0.3                                                          GRA  33
      DO 3 K=1,NPLOTS                                                    GRA  34
C                                                                        GRA  35
C     WRITE SYMBOL,PEN UP                                                GRA  36
      YD=YD+0.05                                                         GRA  37
      CALL SYMBOL (XD,YD,0.1,K,0.0,-1)                                   GRA  38
C                                                                        GRA  39
C     WRITE STREAM AND NUMBER                                            GRA  40
      YD=YD-0.05                                                         GRA  41
      CALL SYMBOL (XD+0.3,YD,0.1,5HSTRM-,0.,5)                           GRA  42
      CALL WHERE (P,Q,F)                                                 GRA  43
      STRMNO=MPLOT(K,1)                                                  GRA  44
      CALL NUMBER (P,Q,0.1,STRMNO,0.0,-1)                                GRA  45
      CALL WHERE (P,Q,F)                                                 GRA  46
C                                                                        GRA  47
C     WRITE VARIABLE NAMES                                               GRA  48
      CALL SYMBOL (P+0.3,Q,0.1,TITLE(MPLOT(K,2)),0.0,4)                  GRA  49
      CALL WHERE (P,Q,F)                                                 GRA  50
C                                                                        GRA  51
C     WRITE RANGES FOR VARIABLES                                         GRA  52
```

```
      CALL NUMBER (P+0.3,Q+0.1,PLOTR(K,1),0.0,2)                    GRA  53
      CALL WHERE (P,Q,F)                                            GRA  54
      CALL SYMBOL (P,Q,0.1,3H = ,0.0,3)                             GRA  55
      CALL WHERE (P,Q,F)                                            GRA  56
      CALL NUMBER (P,Q,0.1,PLOTR(K,2),0.0,2)                        GRA  57
      YD=YD-0.2                                                     GRA  58
3     CONTINUE                                                      GRA  59
      CALL ENDPLT                                                   GRA  60
      RETURN                                                        GRA  61
      END                                                           GRA  62-
```

```
      REAL FUNCTIONPROPS(IG,IS)                                          PRP    1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                  PRP    2
      COMMON /CON/ IU,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      PRP    3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                     PRP    4
C     ENTER HERE ON FIRST CALL                                           PRP    5
      PROPS=0.0                                                          PRP    6
      IGFLAG=0                                                           PRP    7
      CALL GET (NAME,X,0)                                                PRP    8
      NPP=X                                                              PRP    9
      IF (NPP) 1,3,5                                                     PRP   10
C     -VE,TREAT AS WATER, NO TEMP COEFF.                                 PRP   11
1     CONTINUE                                                           PRP   12
      DO 2 I=1,NCOMP                                                     PRP   13
      DO 2 J=1,20                                                        PRP   14
      PP(I,J)=0.                                                         PRP   15
      IF (J.EQ.1) PP(I,J)=18.                                            PRP   16
      IF (J.EQ.2) PP(I,J)=1.                                             PRP   17
      IF (J.EQ.16) PP(I,J)=62.4                                          PRP   18
2     CONTINUE                                                           PRP   19
      WRITE (6,7)                                                        PRP   20
      RETURN                                                             PRP   21
3     IGFLAG=1                                                           PRP   22
C     TREAT AS I.G., MW. AS AIR.                                         PRP   23
      DO 4 I=1,NCOMP                                                     PRP   24
      DO 4 J=1,20                                                        PRP   25
      PP(I,J)=0.                                                         PRP   26
      IF (J.EQ.1) PP(I,J)=.29                                            PRP   27
      IF (J.EQ.2) PP(I,J)=.26                                            PRP   28
4     CONTINUE                                                           PRP   29
      WRITE (6,8)                                                        PRP   30
      RETURN                                                             PRP   31
5     WRITE (6,10) NPP                                                   PRP   32
      DO 6 I=1,NCOMP                                                     PRP   33
      READ (5,9) (PP(I,J),J=1,NPP)                                       PRP   34
      WRITE (6,11) (PP(I,J),J=1,NPP)                                     PRP   35
6     CONTINUE                                                           PRP   36
C                                                                        PRP   37
      RETURN                                                             PRP   38
C                                                                        PRP   39
7     FORMAT (20H PROPERTIES AS WATER)                                   PRP   40
8     FORMAT (18H PROPERTIES AS AIR)                                     PRP   41
9     FORMAT (12X,5F12.0)                                                PRP   42
10    FORMAT (15H PROPERTY TABLE,I4,8H ENTRIES).                         PRP   43
11    FORMAT (5F12.5)                                                    PRP   44
      END                                                                PRP   45
```

```
      FUNCTION CP (IG,IS)                                                    CP     1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                      CP     2
      COMMON /CON/ IU,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH          CP     3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                         CP     4
      IF (IGFLAG.NE.1) GO TO 1                                               CP     5
      CP=0.26                                                                CP     6
      RETURN                                                                 CP     7
    1 SUM=0.0                                                                CP     8
      T=S(IG,IS,4)+PP(1,20)                                                  CP     9
      DO 2 I=1,NCOMP                                                         CP    10
      SUM=SUM+S(IG,IS,I+5)*(((PP(I,5)*T+PP(I,4))*T+PP(I,3))*T+PP(I,2))       CP    11
    2 CONTINUE                                                               CP    12
      CP=SUM                                                                 CP    13
      RETURN                                                                 CP    14
      END                                                                    CP    15
```

```
      FUNCTION MW (IG,IS)                                                MW     1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                  MW     2
      COMMON /CON/ IU,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      MW     3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                     MW     4
      DO 1 I=1,NCOMP                                                     MW     5
      SUM=SUM+PP(I,1)*S(IG,IS,I+5)                                       MW     6
1     CONTINUE                                                           MW     7
      MW=SUM                                                             MW     8
      RETURN                                                             MW     9
      END                                                                MW    10
```

```
      FUNCTION SG (IG,IS)                                                    SG     1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                      SG     2
      COMMON /CON/ IU,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH          SG     3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                         SG     4
      IF (IGFLAG.NE.1) GO TO 2                                               SG     5
      SUM=0.                                                                 SG     6
      DO 1 I=1,NCOMP                                                         SG     7
      SUM=SUM+PP(I,16)*S(IG,IS,I+5)                                          SG     8
1     CONTINUE                                                               SG     9
      PROPS=S(IG,IS,5)/(10.71*S(IG,IS,4))                                    SG    10
      SG=PROPS*SUM                                                           SG    11
      RETURN                                                                 SG    12
2     SUM=0.                                                                 SG    13
      T=S(IG,IS,4)                                                           SG    14
      DO 3 I=1,NCOMP                                                         SG    15
      SUM=SUM+S(IG,IS,I+5)*(PP(I,17)+PP(I,18)*T)                             SG    16
3     CONTINUE                                                               SG    17
      SG=SUM                                                                 SG    18
      RETURN                                                                 SG    19
      END                                                                    SG    20
```

```
      FUNCTION ENTHL (IG,IS)                                                ETL    1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                     ETL    2
      COMMON /CON/ IU,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH         ETL    3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                        ETL    4
      T=S(IG,IS,4)+PP(1,20)                                                 ETL    5
      HL=0.                                                                 ETL    6
      DO 1 I=1,NCOMP                                                        ETL    7
      HL=HL+S(IG,IS,I+5)*(((1./4.*PP(I,5)*T+1./3.*PP(I,4))*T+1./2.*PP(I,ETL    8
     13))*T+PP(I,2))                                                        ETL    9
1     CONTINUE                                                              ETL   10
      ENTHL=HL*T                                                            ETL   11
      RETURN                                                                ETL   12
      END                                                                   ETL   13
```

```
      FUNCTION ENTHV (IG,IS)                                                    ETV    1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                         ETV    2
      COMMON /CON/ IU,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH             ETV    3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                            ETV    4
      T=S(IG,IS,4)+PP(1,20)                                                     ETV    5
      HV=0.                                                                     ETV    6
      DO 1 I=1,NCOMP                                                            ETV    7
      HV=HV+S(IG,IS,I+5)*((((PP(I,15)*T+PP(I,14))*T+PP(I,13))*T+PP(I,12ETV    8
     1))*T+PP(I,11))*T+PP(I,10))
    1 CONTINUE                                                                  ETV    9
      ENTHV=HV                                                                  ETV   10
      RETURN                                                                    ETV   11
      END                                                                       ETV   12
                                                                                ETV   13
```

```
      FUNCTION EQUIL (IL,IV)                                                    EQL   1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                         EQL   2
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,YMAX,IORDER,NGRAPH             EQL   3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                            EQL   4
      COMMON /KVALU/ HKK(10)                                                    EQL   5
C     DUMMY STATEMENT                                                           EQL   6
      EQUIL=0.0                                                                 EQL   7
      T=S(IG,IL,4)+PP(1,20)                                                     EQL   8
      IF (T.EQ.0.) T=100.                                                       EQL   9
      K=0                                                                       EQL  10
1     CONTINUE                                                                  EQL  11
      K=K+1                                                                     EQL  12
      IF (K.GE.50) PRINT 3, T,SUM,IL,IV                                         EQL  13
      IF (K.GE.50) PRINT 4, (S(IG,IL,J),S(IG,IV,J),J=1,10)                      EQL  14
      IF (K.GE.50) RETURN                                                       EQL  15
      DUM=KVAL(IL,IV)                                                           EQL  16
      SUM=0.0                                                                   EQL  17
      SUMY=0.0                                                                  EQL  18
      DO 2 NC=6,NC5                                                             EQL  19
      N=NC-5                                                                    EQL  20
      S(IG,IV,NC)=HKK(N)*S(IG,IL,NC)                                            EQL  21
      SUM=SUM+S(IG,IV,NC)                                                       EQL  22
      DY=-S(IG,IV,NC)*PP(N,7)/(T+PP(N,8))**2                                    EQL  23
      SUMY=SUMY+DY                                                              EQL  24
2     CONTINUE                                                                  EQL  25
      YER=1.-SUM                                                                EQL  26
      T=T+YER/SUMY                                                              EQL  27
      S(IG,IL,4)=T-PP(1,20)                                                     EQL  28
      IF (ABS(YER).GT.0.01) GO TO 1                                             EQL  29
      S(IG,IL,4)=T-PP(1,20)                                                     EQL  30
      S(IG,IV,4)=S(IG,IL,4)                                                     EQL  31
      RETURN                                                                    EQL  32
C                                                                               EQL  33
3     FORMAT (" ITER,SUMY,IL,IV ",2E15.6,2I5)                                   EQL  34
4     FORMAT (2E15.5)                                                           EQL  35
      END                                                                       EQL  36
```

```
      REAL FUNCTIONKVAL(IL,IV)                                                KVL   1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                       KVL   2
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH           KVL   3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                          KVL   4
      COMMON /KVALU/ HKK(10)                                                  KVL   5
C     DUMMY STATEMENT                                                         KVL   6
      KVAL=0.0                                                                KVL   7
      T=S(IG,IL,4)+PP(I,20)                                                   KVL   8
      P=S(IG,IL,5)                                                            KVL   9
      DUM=ACTY(IL)                                                            KVL  10
      DO 1 I=1,NCOMP                                                          KVL  11
      HKK(I)=EXP(PP(I,6)+PP(I,7)/(T+PP(I,8)))*PP(I,9)/P                       KVL  12
1     CONTINUE                                                                KVL  13
      RETURN                                                                  KVL  14
      END                                                                     KVL  15
```

```
      FUNCTION ACTY (IS)                                                    ATY   1
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                     ATY   2
      COMMON /CON/ IU,NCOMP,NCS,H,NE,NS,NPR,NPO,TMAX,IORDER,NGRAPH.         ATY   3
      COMMON /PTAB/ IGFLAG,PP(10,20)                                        ATY   4
C     DUMMY STATEMENT                                                       ATY   5
      ACTY=0.0                                                              ATY   6
      DO 1 I=1,NCOMP                                                        ATY   7
      PP(I,9)=1.                                                            ATY   8
1     CONTINUE                                                              ATY   9
      RETURN                                                                ATY  10
      END                                                                   ATY  11
```

```
      FUNCTION TEMPE (Q,MASS,N)                                          TMP    1
      REAL MASS                                                          TMP    2
      COMMON /MAT/ MP(35,13),EP(35,10),S(2,45,11),EX(1)                  TMP    3
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      TMP    4
      COMMON /PTAB/ IGFLAG,PP(10,20)                                     TMP    5
      T=S(IG,N,4)+PP(1,20)                                               TMP    6
      IF (Q.LE.0..AND.MASS.LE.0.) GO TO 3                                TMP    7
      X1=0.                                                              TMP    8
      X2=0.                                                              TMP    9
      X3=0.                                                              TMP   10
      X4=0.                                                              TMP   11
      DO 1 I=1,NCOMP                                                     TMP   12
      J=I+5                                                              TMP   13
      X1=X1+S(IG,N,J)*PP(I,2)                                            TMP   14
      X2=X2+S(IG,N,J)*PP(I,3)/2.                                         TMP   15
      X3=X3+S(IG,N,J)*PP(I,4)/3.                                         TMP   16
      X4=X4+S(IG,N,J)*PP(I,5)/4.                                         TMP   17
    1 CONTINUE                                                           TMP   18
    2 CONTINUE                                                           TMP   19
      F=Q-(((X4*T+X3)*T+X2)*T+X1)*T                                      TMP   20
      DF=-(((4.*X4*T+3.*X3)*T+2.*X2)*T+X1)                               TMP   21
      T2=T-F/DF                                                          TMP   22
      IF (ABS(T-T2).LT..01) GO TO 3                                      TMP   23
      T=T2                                                               TMP   24
      GO TO 2                                                            TMP   25
    3 TEMPE=T-PP(1,20)                                                   TMP   26
      RETURN                                                             TMP   27
      END                                                                TMP   28
```

```
      SUBROUTINE DIFSUB (N,YY,DERY)                                          DIF    1
      COMMON /GEAR1/ Y(8,222),SAVE(10,222),ERROR(222),YMAX(222),DERY1(22DIF    2
     12),DERY2(222)                                                          DIF    3
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,NOPPT,ISTIFF         DIF    4
      COMMON /CON/ IG,NCOMP,NCS,H,NE,NS,NRR,NPOL,TMAX,IORDER,NGRAPH          DIF    5
      COMMON /GEAR3/ HMIN,HMAX,XMIN,NOMES                                    DIF    6
      COMMON /BKV/ NBV                                                       DIF    7
      COMMON /ROW/ IROW(1887)                                                DIF    8
      COMMON /COLUMN/ JCOL(1887)                                             DIF    9
      COMMON /JACOB/ PW(2109)                                                DIF   10
      COMMON /SUBDI/ ASUB(49)                                                DIF   11
      COMMON /DIAG/ BDIAG(49)                                                DIF   12
      COMMON /SUPERD/ CSUP(49)                                               DIF   13
      COMMON /MODULE/ IDERY,ITER,ITRI,NZERO                                  DIF   14
      DIMENSION A(8), PERTST(7,2,3)                                          DIF   15
      DIMENSION NOP(1000), NW(500), PWSAVE(200)                              DIF   16
      DIMENSION NOPS(1000)                                                   DIF   17
      DIMENSION YY(222), DERY(222)                                           DIF   18
      DATA ITRI,IDERY/2*0/                                                   DIF   19
      DATA PERTST/2.0,4.5,7.333,10.42,13.7,17.15,1.0,2.0,12.0,24.0,37.89DIF   20
     1,53.33,70.08,87.97,3.0,6.0,9.167,12.5,15.98,1.0,1.0,12.0,24.0,37.8DIF   21
     29,53.33,70.08,87.97,1.0,1.0,1.0,0.5,0.1667,0.04133,0.008267,1.0,1.DIF   22
     30,1.0,2.0,1.0,0.3157,0.07407,0.0139/                                 DIF   23
      DATA A(2)/-1.0/                                                        DIF   24
      IF (IDERY.EQ.1) GO TO 52                                               DIF   25
      NBV1=NBV+1                                                             DIF   26
      NBV=NBV+N                                                              DIF   27
      IF (NBVMAX.LT.NBV) NBVMAX=NBV                                          DIF   28
      DO 1 I=NBV1,NBV                                                        DIF   29
      II=I-NBV1+1                                                            DIF   30
      DERY1(I)=DERY(II)                                                      DIF   31
    1 CONTINUE                                                               DIF   32
      IF (IG.EQ.1) GO TO 26                                                  DIF   33
C                                                                            DIF   34
C     PREDICTOR SECTION                                                      DIF   35
C                                                                            DIF   36
      IRET=1                                                                 DIF   37
      IF (JSTART.EQ.0) GO TO 4                                               DIF   38
    2 DO 3 I=NBV1,NBV                                                        DIF   39
      DO 3 J=1,K                                                             DIF   40
    3 SAVE(J,I)=Y(J,I)                                                       DIF   41
      HOLD=H                                                                 DIF   42
      NQOLD=NQ                                                               DIF   43
      RACUM=1.0                                                              DIF   44
      IF (JSTART.GT.0) GO TO 23                                              DIF   45
      GO TO 6                                                                DIF   46
    4 CONTINUE                                                               DIF   47
      NQ=1                                                                   DIF   48
      N3=N                                                                   DIF   49
      N4=N*N                                                                 DIF   50
      DO 5 I=NBV1,NBV                                                        DIF   51
      II=I-NBV1+1                                                            DIF   52
```

```
      Y(1,I)=YY(II)                                                    DIF  53
      Y(2,I)=DERY1(I)*H                                                DIF  54
5     CONTINUE                                                         DIF  55
      HNEW=H                                                           DIF  56
      K=2                                                              DIF  57
      GO TO 2                                                          DIF  58
6     IF (ISTIFF.EQ.0) GO TO 7                                         DIF  59
      IF (NQ.GT.6.OR.NQ.LT.1) GO TO 8                                  DIF  60
      GO TO (16,17,18,19,20,21), NQ                                    DIF  61
7     IF (NQ.GT.7.OR.NQ.LT.1) GO TO 8                                  DIF  62
      GO TO (9,10,11,12,13,14,15), NQ                                  DIF  63
8     PRINT 84, NQ                                                     DIF  64
      STOP                                                             DIF  65
C                                                                      DIF  66
C     NONSTIFF COEFFICIENTS,ORDER 1-7                                  DIF  67
C                                                                      DIF  68
9     A(1)=-1.0                                                        DIF  69
      GO TO 22                                                         DIF  70
10    A(1)=-0.500000000                                                DIF  71
      A(3)=-0.500000000                                                DIF  72
      GO TO 22                                                         DIF  73
11    A(1)=-0.41666666666667                                           DIF  74
      A(3)=-0.750000000                                                DIF  75
      A(4)=-0.16666666666667                                           DIF  76
      GO TO 22                                                         DIF  77
12    A(1)=-0.375000000                                                DIF  78
      A(3)=-0.91666666666667                                           DIF  79
      A(4)=-0.33333333333333                                           DIF  80
      A(5)=-0.41666666666667E-01                                       DIF  81
      GO TO 22                                                         DIF  82
13    A(1)=-0.34861111111111                                           DIF  83
      A(3)=-1.0416666666667                                            DIF  84
      A(4)=-0.48611111111111                                           DIF  85
      A(5)=-0.10416666666667                                           DIF  86
      A(6)=-0.83333333333333E-02                                       DIF  87
      GO TO 22                                                         DIF  88
14    A(1)=-0.32986111111111                                           DIF  89
      A(3)=-1.1416666666667                                            DIF  90
      A(4)=-0.625000000                                                DIF  91
      A(5)=-0.17708333333333                                           DIF  92
      A(6)=-0.025000000                                                DIF  93
      A(7)=-0.13888888888889E-02                                       DIF  94
      GO TO 22                                                         DIF  95
15    A(1)=-0.31559193121693                                           DIF  96
      A(3)=-1.235000000                                                DIF  97
      A(4)=-0.75185185185                                              DIF  98
      A(5)=-0.25520833333333                                           DIF  99
      A(6)=-0.48611111111111E-01                                       DIF 100
      A(7)=-0.48611111111111E-02                                       DIF 101
      A(8)=-0.1984126984127E-03                                        DIF 102
      GO TO 22                                                         DIF 103
C                                                                      DIF 104
```

```
C       STIFF COEFFICIENTS,ORDER 1-6                                    DIF 105
C                                                                       DIF 106
16      A(1)=-1.000000000                                               DIF 107
        GO TO 22                                                        DIF 108
17      A(1)=-0.66666666666667                                          DIF 109
        A(3)=-0.33333333333333                                          DIF 110
        GO TO 22                                                        DIF 111
18      A(1)=-0.54545454545455                                          DIF 112
        A(3)=A(1)                                                       DIF 113
        A(4)=-0.90909090909091E-01                                      DIF 114
        GO TO 22                                                        DIF 115
19      A(1)=-0.48000000                                                DIF 116
        A(3)=-0.700000000                                               DIF 117
        A(4)=-0.200000000                                               DIF 118
        A(5)=-0.020000000                                               DIF 119
        GO TO 22                                                        DIF 120
20      A(1)=-0.43795620437956                                          DIF 121
        A(3)=-0.82116788321169                                          DIF 122
        A(4)=-0.31021897810219                                          DIF 123
        A(5)=-0.54744525547445E-01                                      DIF 124
        A(6)=-0.36496350364964E-02                                      DIF 125
        GO TO 22                                                        DIF 126
21      A(1)=-0.40816326530612                                          DIF 127
        A(3)=-0.92063492063492                                          DIF 128
        A(4)=-0.41666666666666                                          DIF 129
        A(5)=-0.99206349206349E-01                                      DIF 130
        A(6)=-0.11904761904762E-01                                      DIF 131
        A(7)=-0.56689342403628E-03                                      DIF 132
22      K=NQ+1                                                          DIF 133
        IDOUB=K                                                         DIF 134
        MTYP=2-ISTIFF                                                   DIF 135
        ENQ2=0.5/FLOAT(NQ+1)                                            DIF 136
        ENQ3=0.5/FLOAT(NQ+2)                                            DIF 137
        ENQ1=0.5/FLOAT(NQ)                                              DIF 138
        EUP=(PERTST(NQ,MTYP,2)*EPS)**2                                  DIF 139
        E=(PERTST(NQ,MTYP,1)*EPS)**2                                    DIF 140
        EDWN=(PERTST(NQ,MTYP,3)*EPS)**2                                 DIF 141
        IF (EDWN.EQ.0) GO TO 83                                         DIF 142
        IF (IRET.EQ.2) GO TO 75                                         DIF 143
        IF (IRET.EQ.3) RETURN                                           DIF 144
23      CONTINUE                                                        DIF 145
        DO 24 J=2,K                                                     DIF 146
        DO 24 J1=J,K                                                    DIF 147
        J2=K-J1+J-1                                                     DIF 148
        DO 24 I=NBV1,NBV                                                DIF 149
24      Y(J2,I)=Y(J2,I)+Y(J2+1,I)                                       DIF 150
        DO 25 I=1,N                                                     DIF 151
        II=I+NBV1-1                                                     DIF 152
        YY(I)=Y(1,II)                                                   DIF 153
25      CONTINUE                                                        DIF 154
        RETURN                                                          DIF 155
26      CONTINUE                                                        DIF 156
```

```
C                                                                       DIF 157
C     CORRECTOR SECTION                                                 DIF 158
C                                                                       DIF 159
      DO 27 I=NBV1,NBV                                                  DIF 160
      ERROR(I)=0.0                                                      DIF 161
27    CONTINUE                                                          DIF 162
      IWEVAL=ISTIFF                                                     DIF 163
      L=1                                                               DIF 164
28    CONTINUE                                                          DIF 165
C                                                                       DIF 166
C     ITER=0 - DIRECT ITERATION OF CORRECTOR                            DIF 167
C     ITER=1 - NEWTON-RAPHSON ITERATION OF CORRECTOR                    DIF 168
C                                                                       DIF 169
      IF (ITER.EQ.0.OR.ISTIFF.EQ.0) GO TO 41                            DIF 170
      R=A(1)*H                                                          DIF 171
      IF (ITRI.EQ.1) GO TO 43                                           DIF 172
C                                                                       DIF 173
C     TRGB OPTION                                                       DIF 174
C                                                                       DIF 175
      IF (IWEVAL.EQ.-1) GO TO 30                                        DIF 176
      DO 29 I=1,NZERO                                                   DIF 177
      PW(I)=PW(I)*R                                                     DIF 178
      IF (IROW(I).EQ.JCOL(I)) PW(I)=1.0+PW(I)                           DIF 179
29    CONTINUE                                                          DIF 180
      IWEVAL=-1                                                         DIF 181
30    DO 31 I=1,N                                                       DIF 182
      IROW(I+NZERO)=I                                                   DIF 183
      JCOL(I+NZERO)=0                                                   DIF 184
      II=I+NBV1-1                                                       DIF 185
      PW(I+NZERO)=Y(2,II)-DERY1(II)*H                                   DIF 186
31    CONTINUE                                                          DIF 187
      NEM=NZERO+N                                                       DIF 188
      DO 32 I=1,NEM                                                     DIF 189
      PWSAVE(I)=PW(I)                                                   DIF 190
32    CONTINUE                                                          DIF 191
      IF (JSTART.EQ.0) GO TO 34                                         DIF 192
      NOP2=NOPS(NOPPT+2)                                                DIF 193
      DO 33 I=1,NOP2                                                    DIF 194
      II=NOPPT+I                                                        DIF 195
      NOP(I)=NOPS(II)                                                   DIF 196
33    CONTINUE                                                          DIF 197
      CALL TRGB2 (NOP,PW,XMIN,NTEST,NELEM)                              DIF 198
      IF (NELEM.EQ.0) GO TO 38                                          DIF 199
      IF (NOMES.EQ.0) PRINT 85,NELEM,NBV1,NBV                           DIF 200
34    CALL TRGB (NOP,NW,PW,JCOL,IROW,N,N,NEM,XMIN)                      DIF 201
      IF (JSTART.EQ.0) GO TO 35                                         DIF 202
      NOP(2)=NOPS(NOPPT+2)                                              DIF 203
      GO TO 36                                                          DIF 204
35    CONTINUE                                                          DIF 205
C                                                                       DIF 206
C     INCREASE LENGTH OF OPERATOR LIST BY 10 PERCENT IN CASE IT MUST    DIF 207
C     BE RE-CREATED                                                     DIF 208
```

```
C                                                                       DIF 209
      NOP(2)=NOP(2)+NOP(2)/10                                           DIF 210
36    CONTINUE                                                          DIF 211
      NOP2=NOP(2)                                                       DIF 212
      DO 37 I=1,NOP2                                                    DIF 213
      II=NOPPT+I                                                        DIF 214
      NOPS(II)=NOP(I)                                                   DIF 215
37    CONTINUE                                                          DIF 216
38    CONTINUE                                                          DIF 217
      DO 39 I=1,NEM                                                     DIF 218
      PW(I)=PWSAVE(I)                                                   DIF 219
39    CONTINUE                                                          DIF 220
      NOP3=NOP(3)                                                       DIF 221
      DO 40 I=1,N                                                       DIF 222
      II=I+NBV1-1                                                       DIF 223
      SAVE(9,II)=PW(NOP3+I)                                             DIF 224
40    CONTINUE                                                          DIF 225
      GO TO 49                                                          DIF 226
C                                                                       DIF 227
C     DIRECT ITERATION OF CORRECTOR                                     DIF 228
C                                                                       DIF 229
41    CONTINUE                                                          DIF 230
      DO 42 I=NBV1,NBV                                                  DIF 231
      SAVE(9,I)=Y(2,I)-DERY1(I)*H                                       DIF 232
42    CONTINUE                                                          DIF 233
      GO TO 49                                                          DIF 234
C                                                                       DIF 235
C     TRIDIAGONAL OPTION                                                DIF 236
C                                                                       DIF 237
43    CONTINUE                                                          DIF 238
      IF (IWEVAL.EQ.-1) GO TO 46                                        DIF 239
      DO 44 I=1,N                                                       DIF 240
      ASUB(I)=ASUB(I)*R                                                 DIF 241
      BDIAG(I)=BDIAG(I)*R                                               DIF 242
      CSUP(I)=CSUP(I)*R                                                 DIF 243
44    CONTINUE                                                          DIF 244
      DO 45 I=1,N                                                       DIF 245
      BDIAG(I)=1.0+BDIAG(I)                                             DIF 246
45    CONTINUE                                                          DIF 247
      IWEVAL=-1                                                         DIF 248
46    CONTINUE                                                          DIF 249
      DO 47 I=NBV1,NBV                                                  DIF 250
      II=I-NBV1+1                                                       DIF 251
      DERY2(II)=Y(2,I)-DERY1(I)*H                                       DIF 252
47    CONTINUE                                                          DIF 253
      CALL TRIDAG (N,ASUB,BDIAG,CSUP,DERY2,YY,DERY,PWSAVE)              DIF 254
      DO 48 I=1,N                                                       DIF 255
      SAVE(9,I)=YY(I)                                                   DIF 256
48    CONTINUE                                                          DIF 257
      ITRI=0                                                            DIF 258
49    NT=N                                                              DIF 259
      BND=EPS*ENQ3/FLOAT(N)                                             DIF 260
```

```
      DO 50 I=NBV1,NBV                                                  DIF 261
      Y(1,I)=Y(1,I)+A(1)*SAVE(9,I)                                      DIF 262
      Y(2,I)=Y(2,I)-SAVE(9,I)                                           DIF 263
      ERROR(I)=ERROR(I)+SAVE(9,I)                                       DIF 264
      IF (ABS(SAVE(9,I)).LE.(BND*YMAX(I))) NT=NT-1                      DIF 265
   50 CONTINUE                                                          DIF 266
      DO 51 I=1,N                                                       DIF 267
      II=I+NBV1-1                                                       DIF 268
      YY(I)=Y(1,II)                                                     DIF 269
   51 CONTINUE                                                          DIF 270
      IF (NT.LE.0) GO TO 56                                             DIF 271
      IF (L.EQ.3) GO TO 54                                              DIF 272
C                                                                       DIF 273
C     RETURN TO MODULE TO GET NEW DERIVATIVE VALUES                     DIF 274
C                                                                       DIF 275
      IDERY=1                                                           DIF 276
      RETURN                                                            DIF 277
   52 IDERY=0                                                           DIF 278
      DO 53 I=NBV1,NBV                                                  DIF 279
      II=I-NBV1+1                                                       DIF 280
      DERY1(I)=DERY(II)                                                 DIF 281
   53 CONTINUE                                                          DIF 282
      L=L+1                                                             DIF 283
      IF (L.LT.4) GO TO 28                                              DIF 284
   54 CONTINUE                                                          DIF 285
      ICONV=1                                                           DIF 286
      RACUM=RACUM*0.25                                                  DIF 287
      IF (NOMES.EQ.0) PRINT 86,NBV1,NBV                                 DIF 288
      IF (H.LE.(HMIN*1.00001)) GO TO 55                                 DIF 289
      GO TO 80                                                          DIF 290
   55 PRINT 87, NBV1,NBV                                                DIF 291
      STOP                                                              DIF 292
   56 CONTINUE                                                          DIF 293
      IF (ITRI.EQ.0.AND.ITER.EQ.1) NOPPT=NOPPT+NOP(2)                   DIF 294
      IF (NBV.NE.NBVMAX) RETURN                                         DIF 295
C                                                                       DIF 296
C     ERROR SECTION                                                     DIF 297
C                                                                       DIF 298
      D=0.0                                                             DIF 299
      DO 57 I=1,NBVMAX                                                  DIF 300
      D=D+(ERROR(I)/YMAX(I))**2                                         DIF 301
   57 CONTINUE                                                          DIF 302
      IF (D.GT.E) GO TO 61                                              DIF 303
      IF (K.LT.3) GO TO 59                                              DIF 304
      DO 58 J=3,K                                                       DIF 305
      DO 58 I=1,NBVMAX                                                  DIF 306
   58 Y(J,I)=Y(J,I)+A(J)*ERROR(I)                                       DIF 307
   59 CONTINUE                                                          DIF 308
      IF (IDOUB.LE.1) GO TO 62                                          DIF 309
      IDOUB=IDOUB-1                                                     DIF 310
      IF (IDOUB.GT.1) GO TO 77                                          DIF 311
      DO 60 I=1,NBVMAX                                                  DIF 312
```

```
      SAVE(10,I)=ERROR(I)                                              DIF 313
60    CONTINUE                                                         DIF 314
      GO TO 77                                                         DIF 315
61    KFLAG=KFLAG-2                                                    DIF 316
      IF (NOMES.EQ.0) PRINT 88                                         DIF 317
      IF (H.LE.(HMIN*1.00001)) GO TO 79                                DIF 318
62    PR2=(D/E)**ENQ2*1.2                                              DIF 319
      PR3=1.0E+20                                                      DIF 320
      IF ((NQ.GE.IORDER).OR.(KFLAG.LE.-1)) GO TO 64                    DIF 321
      D=0.0                                                            DIF 322
      DO 63 I=1,NBVMAX                                                 DIF 323
      D=D+((ERROR(I)-SAVE(10,I))/YMAX(I))**2                           DIF 324
63    CONTINUE                                                         DIF 325
      PR3=(D/EUP)**ENQ3*1.4                                            DIF 326
64    PR1=1.0E+20                                                      DIF 327
      IF (NQ.LE.1) GO TO 66                                            DIF 328
      D=0.0                                                            DIF 329
      DO 65 I=1,NBVMAX                                                 DIF 330
      D=D+(Y(K,I)/YMAX(I))**2                                          DIF 331
65    CONTINUE                                                         DIF 332
      PR1=(D/EDWN)**ENQ1*1.3                                           DIF 333
66    CONTINUE                                                         DIF 334
      IF (PR2.LE.PR3) GO TO 72                                         DIF 335
      IF (PR3.LT.PR1) GO TO 73                                         DIF 336
67    R=1.0/AMAX1(PR1,1.0E-04)                                         DIF 337
      NEWQ=NQ-1                                                        DIF 338
68    IDOUB=10                                                         DIF 339
      IF ((KFLAG.EQ.1).AND.(R.LT.1.1)) GO TO 77                        DIF 340
      IF (NEWQ.LE.NQ) GO TO 70                                         DIF 341
      DO 69 I=1,NBVMAX                                                 DIF 342
      Y(NEWQ+1,I)=ERROR(I)*A(K)/FLOAT(K)                               DIF 343
69    CONTINUE                                                         DIF 344
70    K=NEWQ+1                                                         DIF 345
      IF (KFLAG.EQ.1) GO TO 74                                         DIF 346
      RACUM=RACUM*R                                                    DIF 347
      IRET1=3                                                          DIF 348
      GO TO 80                                                         DIF 349
71    CONTINUE                                                         DIF 350
      IRET1=0                                                          DIF 351
      IF (NEWQ.EQ.NQ) RETURN                                           DIF 352
      NQ=NEWQ                                                          DIF 353
      IRET=3                                                           DIF 354
      GO TO 6                                                          DIF 355
72    IF (PR2.GT.PR1) GO TO 67                                         DIF 356
      NEWQ=NQ                                                          DIF 357
      R=1.0/AMAX1(PR2,1.0E-04)                                         DIF 358
      GO TO 68                                                         DIF 359
73    R=1.0/AMAX1(PR3,1.0E-04)                                         DIF 360
      NEWQ=NQ+1                                                        DIF 361
      GO TO 68                                                         DIF 362
74    CONTINUE                                                         DIF 363
      R=AMIN1(R,HMAX/ABS(H))                                           DIF 364
```

```
      H=H*R                                                              DIF 365
      IF (NQ.EQ.NEWQ) GO TO 75                                           DIF 366
      NQ=NEWQ                                                            DIF 367
      IRET=2                                                             DIF 368
      GO TO 6                                                            DIF 369
75    R1=1.0                                                             DIF 370
      DO 76 J=2,K                                                        DIF 371
      R1=R1*R                                                            DIF 372
      DO 76 I=1,NBVMAX                                                   DIF 373
76    Y(J,I)=Y(J,I)*R1                                                   DIF 374
      IDOUB=K                                                            DIF 375
77    CONTINUE                                                           DIF 376
      DO 78 I=1,NBVMAX                                                   DIF 377
      YMAX(I)=AMAX1(YMAX(I),ABS(Y(1,I)))                                 DIF 378
78    CONTINUE                                                           DIF 379
      JSTART=NQ                                                          DIF 380
      RETURN                                                             DIF 381
79    PRINT 89                                                           DIF 382
      STOP                                                               DIF 383
80    CONTINUE                                                           DIF 384
      RACUM=AMAX1(ABS(HMIN/HOLD),RACUM)                                  DIF 385
      RACUM=AMIN1(RACUM,ABS(HMAX/HOLD))                                  DIF 386
      H=HOLD*RACUM                                                       DIF 387
      R1=1.0                                                             DIF 388
      DO 81 J=2,K                                                        DIF 389
      R1=R1*RACUM                                                        DIF 390
      DO 81 I=1,NBVMAX                                                   DIF 391
81    Y(J,I)=SAVE(J,I)*R1                                                DIF 392
      DO 82 I=1,NBVMAX                                                   DIF 393
      Y(1,I)=SAVE(1,I)                                                   DIF 394
82    CONTINUE                                                           DIF 395
      IDOUB=K                                                            DIF 396
      IF (IRET1.EQ.3) GO TO 71                                           DIF 397
      RETURN                                                             DIF 398
83    PRINT 90                                                           DIF 399
      STOP                                                               DIF 400
C                                                                        DIF 401
84    FORMAT (41H THE MAXIMUM ORDER SPECIFIED IS TOO LARGE,I5)           DIF 402
85    FORMAT (43H THE CORRECTOR CANNOT BE SOLVED BECAUSE PW),I4,47H ) ISDIF 403
     1 A PIVOT ELEMENT WHOSE VALUE IS BELOW XMIN,/,23H DIFFERENTIAL EQUADIF 404
     2TIONS,I4,3H TO,I4,/,33H OPERATOR LIST WILL BE RE-CREATED)          DIF 405
86    FORMAT (49H THE CORRECTOR FAILED TO CONVERGE IN 3 ITERATIONS,23H DDIF 406
     1IFFERENTIAL EQUATIONS,I4,3H TO,I4)                                DIF 407
87    FORMAT (58H CORRECTOR CONVERGENCE COULD NOT BE ACHIEVED FOR H.GT.HDIF 408
     1MIN,/,23H DIFFERENTIAL EQUATIONS,I4,3H TO,I4)                     DIF 409
88    FORMAT (30H TRUNCATION ERROR IS TOO LARGE)                         DIF 410
89    FORMAT (72H THE STEP WAS TAKEN WITH H=HMIN BUT THE REQUESTED ERRORDIF 411
     1 WAS NOT ACHIEVED)                                                DIF 412
90    FORMAT (68H THE REQUESTED ERROR IS SMALLER THAN CAN BE HANDLED FORDIF 413
     1 THIS PROBLEM)                                                    DIF 414
      END                                                                DIF 415
```

```
      SUBROUTINE TRIDAG (N,A,B,C,D,V,BETA,GAMMA)                        TRI    1
C                                                                       TRI    2
C     SUBROUTINE FOR SOLVING A SYSTEM OF N LINEAR SIMULTANEOUS EQUATIONSTRI    3
C     HAVING A TRIDIAGONAL COEFFICIENT MATRIX.                          TRI    4
C                                                                       TRI    5
C     SOURCE - APPLIED NUMERICAL METHODS,CARNAHAN LUTHER, & WILKES,1969 TRI    6
C                                                                       TRI    7
C     ARGUMENTS                                                         TRI    8
C                                                                       TRI    9
C     N - NUMBER OF EQUATIONS                                           TRI   10
C     A - SUBDIAGONAL COEFFICIENTS ARE STORED IN A(2)...A(N)            TRI   11
C         A(1) IS NOT USED                                              TRI   12
C     B - DIAGONAL COEFFICIENTS                                         TRI   13
C     C - SUPERDIAGONAL COEFFICIENTS ARE STORED IN C(1)...C(N-1)        TRI   14
C         C(N) IS NOT USED                                              TRI   15
C     D - RIGHT HAND SIDE VECTOR                                        TRI   16
C     V - SOLUTION VECTOR                                               TRI   17
C     BETA,GAMMA - INTERMEDIATE ARRAYS                                  TRI   18
C                                                                       TRI   19
C     DIMENSIONS - ALL ARGUMENTS EXCEPT N MUST BE DIMENSIONED AT LEAST  TRI   20
C                  N IN CALLING PROGRAM                                 TRI   21
C                                                                       TRI   22
      DIMENSION A(N), B(N), C(N), D(N), V(N), BETA(N), GAMMA(N)         TRI   23
C                                                                       TRI   24
C     COMPUTE INTERMEDIATE ARRAYS BETA AND GAMMA                        TRI   25
C                                                                       TRI   26
      BETA(1)=B(1)                                                      TRI   27
      GAMMA(1)=D(1)/BETA(1)                                             TRI   28
      DO 1 I=2,N                                                        TRI   29
      BETA(I)=B(I)-A(I)*C(I-1)/BETA(I-1)                                TRI   30
      GAMMA(I)=(D(I)-A(I)*GAMMA(I-1))/BETA(I)                           TRI   31
1     CONTINUE                                                          TRI   32
C                                                                       TRI   33
C     COMPUTE FINAL SOLUTION VECTOR V                                   TRI   34
C                                                                       TRI   35
      V(N)=GAMMA(N)                                                     TRI   36
      LAST=N-1                                                          TRI   37
      DO 2 K=1,LAST                                                     TRI   38
      I=N-K                                                             TRI   39
      V(I)=GAMMA(I)-C(I)*V(I+1)/BETA(I)                                 TRI   40
2     CONTINUE                                                          TRI   41
      RETURN                                                            TRI   42
      END                                                               TRI   43
```

FIGURE I.2: SKELETON OF MODULE FOR DYNSYS 2.1

```
      SUBROUTINE TYPE2.1
C
C     COMMENTS DESCRIBING MODULE
C
C     THE FOLLOWING COMMON BLOCKS ARE OR MAY BE REQUIRED:
C     MAT,CON,PTAB,UNIT,GEAR2,MODULE,ROW,COLUMN,JACOB,SUBDI,DIAG,SUPERD
C
      COMMON/MAT/MP( , ),EP( , ),S( , , ),EX( )
      COMMON/CON/IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH
      COMMON/PTAB/IGFLAG,PP( , )
      COMMON/UNIT/IM,NMP
      COMMON/GEAR2/EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,ISTIFF
      COMMON/MODULE/IDERY,ITER,ITRI,MC,IPIVOT
C
C     THE ROW DIMENSION OF JCOL AND XJACOB MUST BE THE SAME AS IN DIFSUB
C     MC IS MAXIMUM NUMBER OF COLUMNS IN XJACOB AND JCOL
C
      COMMON/COLUMN/JCOL( ,MC)     (NORMAL OPTION)
      COMMON/JACOB/XJACOB( ,MC)    (NORMAL OPTION)
C
C     N IS NUMBER OF ODES
C
      COMMON/SUBDI/A(N)     (TRIDIAGONAL OPTION)
      COMMON/DIAG/B(N)      (TRIDIAGONAL OPTION)
      COMMON/SUPERD/C(N)    (TRIDIAGONAL OPTION)
C
C*************************************************************************
C                  SECTION #1 : PARAMETER CALCULATIONS
C*************************************************************************
C
C     CALCULATE MODULE PARAMETERS : STREAM INPUT,INITIAL CONDITIONS,
C                                   VALUES OF ITER ETC.
C
C     INPUT STREAM INFORMATION IS OBTAINED FROM S(IG, , )
C     VALUE OF INDEPENDENT VARIABLE (Y) NEED ONLY BE SPECIFIED ON FIRST
C        INTEGRATION STEP (I.E. INITIAL CONDITIONS)
C     JSTART=0 ON FIRST INTEGRATION STEP
C           =CURRENT ORDER OF INTEGRATION TECHNIQUE ON LATER STEPS
C
C     IF ISTIFF=0,NONSTIFF COEFFICIENTS WILL BE USED IN INTEGRATION ALGORITHM
C                 (DIFSUB) FOR ALL MODULES,
C                 JACOBIAN MATRIX IS NOT REQUIRED
C
C     IF ISTIFF=1,STIFF COEFFICIENTS WILL BE USED FOR ALL MODULES
C                 THEN,IF ITER=0 DIRECT ITERATION OF CORRECTOR WILL BE USED
C                                (NONSTIFF MODULE)
C                                JACOBIAN IS NOT REQUIRED
C                      IF ITER=1 NEWTON-RAPHSON ITERATION OF CORRECTOR WILL
C                                BE USED (STIFF MODULE)
```

```
C                                  JACOBIAN MATRIX MUST BE SUPPLIED
C
C     ITER MUST BE SPECIFIED 0 OR 1
C     IT IS NOT USED UNLESS ISTIFF=1
C
      IF(IG.EQ.2) GO TO 2
C
C*****************************************************************************
C                    SECTION #2 : JACOBIAN EVALUATION
C*****************************************************************************
C
C     SECTION #2 IS OMITTED FOR NONSTIFF MODULE
C
C     JACOBIAN MATRIX IS REQUIRED ONLY IF MODULE EQUATIONS ARE STIFF
C     CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS ONLY
C     JACOBIAN NEED NOT BE VERY ACCURATE,AS IT IS USED ONLY FOR CONVERGENCE
C        OF CORRECTOR
C     ONLY NONZERO ELEMENTS ARE CALCULATED,BUT ALL DIAGONAL ELEMENTS MUST
C        BE STORED WHETHER OR NOT THEY ARE ZERO (NORMAL OPTION)
C
C             NORMAL OPTION
C
C     IF JACOBIAN MATRIX IS NOT TRIDIAGONAL,KEYS STORAGE SCHEME IS USED
C     COLUMN NUMBERS OF NONZERO ELEMENTS IN ROW I ARE STORED IN
C             (JCOL(I,J),J=1,NUMBER OF NONZEROS IN THAT ROW)
C     SIMILARLY THE VALUES OF THE NONZERO ELEMENTS OF ROW I ARE
C             STORED IN XJACOB(I, )
C     REMEMBER TO STORE ALL DIAGONAL ELEMENTS EVEN IF THEY ARE ZERO
C
C     MC      - MAXIMUM NUMBER OF COLUMNS IN XJACOB MATRIX
C               MC CAN BE LARGER THAN THE MAXIMUM NUMBER OF NONZERO ELEMENTS
C               IN ANY JACOBIAN ROW SINCE ADDITIONAL NONZERO ELEMENTS CAN BE
C               CREATED DURING THE SOLUTION,MC IS CONSTANT FOR ANY SET OF ODES
C               AND MAY HAVE TO BE DETERMINED BY TRIAL AND ERROR
C     IPIVOT - PIVOT OPTION USED IN SIMULT (1-7)
C        IPIVOT=1 : SIMPLE GAUSS-JORDAN ELIMINATION
C        IPIVOT=2 : GAUSS-JORDAN PARTIAL PIVOTING
C        IPIVOT=3 : GAUSS-JORDAN FULL PIVOTING
C        IPIVOT=4 : MINIMUM ROW-MINIMUM COLUMN
C        IPIVOT=5 : MINIMUM COLUMN-MINIMUM ROW
C        IPIVOT=6 : MAXIMUM COLUMN-MINIMUM ROW
C        IPIVOT=7 : MINIMUM OF ROW ENTRIES TIMES COLUMN ENTRIES
C
C     ICOL(I,J)    - COLUMN NUMBER OF JTH NONZERO ELEMENT IN ROW I
C     XJACOB(I,J) - VALUE OF JTH NONZERO ELEMENT OF ROW I
C
      MC=
      IPIVOT=
      IROW(1,1)=
      XJACOB(1,1)=
         .
         .
```

```
      IROW(N, )=
      XJACOB(N, )=
C
C              TRIDIAGONAL OPTION
C
C     FOR TRIDIAGONAL JACOBIAN MATRIX,THE SUBDIAGONAL,DIAGONAL AND SUPERDIAGONAL
C     ELEMENT VALUES ARE STORED IN ARRAYS A,B AND C FROM COMMON BLOCKS SUBDI,
C     DIAG AND SUPERD RESPECTIVELY
C
C     N - NUMBER OF ODES
C     A - VALUES OF SUBDIAGONAL ELEMENTS ARE STORED IN A(2)...A(N)
C         A(1) IS NOT USED
C     B - VALUES OF DIAGONAL ELEMENTS
C     C - VALUES OF SUPERDIAGONAL ELEMENTS ARE STORED IN C(1)...C(N-1)
C         C(N) IS NOT USED
C
      B(1)=
      C(1)=
      A(2)=
      B(2)=
      C(2)=
         .
         .
         .
      A(N-1)=
      B(N-1)=
      C(N-1)=
      A(N)=
      B(N)=
    1 CONTINUE
C
C     IF TRIDIAGONAL OPTION IS BEING USED,ITRI MUST BE SET TO 1 HERE,
C     IF NORMAL OPTION IS USED ITRI MAY BE IGNORED
C
C     ITRI=1
C
    2 CONTINUE
C
C*****************************************************************************
C                    SECTION #3 : DERIVATIVE CALCULATION
C*****************************************************************************
C
C     CALCULATE DERIVATIVES
C
      DERY(1)=
         .
         .
         .
      DERY(N)=
C
C*****************************************************************************
```

```
C                  SECTION #4 : CALL DIFSUB
C*****************************************************************************
C
C     CALL DIFSUB TO SOLVE ODES FOR MODULE
C     DIFSUB MAY BE CALLED MORE THAN ONCE FROM A MODULE FOR EXAMPLE IF THE
C        MODULE CONTAINS A SET OF STIFF O.D.E.S AND ANOTHER SET OF NONSTIFF
C        O.D.E.S
C
C     ARGUMENTS :  N    - NUMBER OF ODES
C                  Y    - INDEPENDENT VARIABLE
C                  DERY - DERIVATIVES
C
      CALL DIFSUB(N,Y,DERY)
C
C     IF IDERY IS NOT ZERO,THE DERIVATIVES WILL BE RE-EVALUATED AND
C        RETURNED TO DIFSUB
C     ITRI MUST ALSO BE RESET IF IT IS 1
C
      IF(IDERY.NE.0) GO TO 1
C
C*****************************************************************************
C                  SECTION #5 : STREAM OUTPUT CALCULATION
C*****************************************************************************
C
C     CALCULATE STREAM OUTPUT (STORED IN S(1, , ))
C
      S(1, , )=
         .
         .
         .
      RETURN
      END
```

```
C     FIGURE I.3: LISTING OF DYNSYS 2.1 EXECUTIVE                                    
C                 (ROUTINES IDENTICAL OR NEARLY IDENTICAL WITH THOSE                 
C                  IN DYNSYS 2.0 ARE NOT LISTED)                                     
C                                                                                    
C                                                                                    
      SUBROUTINE DYN1                                                          DY1    1
C     DATA LOADING AND PRINTING                                                DY1    2
      COMMON /UNIT/ IM,NMP                                                     DY1    3
      COMMON /MAT/ MP(15,5),EP(15,5),S(2,16,7),EX(1)                           DY1    4
      COMMON /CON/ IG,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH            DY1    5
      COMMON /FTN/ FN(1,1),NFN                                                 DY1    6
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,ISTIFF                 DY1    7
      COMMON /GEAR3/ HMIN,HMAX,NOMES                                           DY1    8
      DIMENSION AM(100)                                                        DY1    9
      DIMENSION ITAG(30)                                                       DY1   10
      DATA ITAG/5HVALV1,5HVALV2,5HSTIR1,5HCONT1,5HSETL1,6HMXPLT1,5HSETT1DY1   11
     1,6HVESSL1,5HDELAY,5HCSTR1,5HEXCH1,5HDCAN1,5HCONV1,5HSPLT1,16*6H   DY1   12
     2   /                                                                       DY1   13
      WRITE (6,27)                                                             DY1   14
C     DEFAULT VALUES                                                           DY1   15
      TIME=0.0                                                                 DY1   16
      MAXNE=20                                                                 DY1   17
      NPR=1                                                                    DY1   18
      TMAX=10.0                                                                DY1   19
      NCOMP=1                                                                  DY1   20
      NMP=5                                                                    DY1   21
      H=0.00001                                                                DY1   22
      HMIN=0.000001                                                            DY1   23
      HMAX=0.05                                                                DY1   24
      EPS=0.001                                                                DY1   25
      NPOL=0                                                                   DY1   26
      NGRAPH=0                                                                 DY1   27
      IORDER=6                                                                 DY1   28
      ISTIFF=1                                                                 DY1   29
      NOMES=0                                                                  DY1   30
C     READ AND COPY TITLE                                                      DY1   31
1     READ (5,29) (AM(I),I=1,8)                                                DY1   32
      WRITE (6,28) (AM(I),I=1,8)                                               DY1   33
      IF (AM(1).NE.5HBEGIN) GO TO 1                                            DY1   34
C     READ INITIAL DATA                                                        DY1   35
2     CALL GET (N,X,1)                                                         DY1   36
      IF (N.EQ.5HCOMPS) NCOMP=X                                                DY1   37
      IF (N.EQ.6HIN/OUT) NMP=X                                                 DY1   38
      IF (N.EQ.6HDELTAT) H=X                                                   DY1   39
      IF (N.EQ.4HHMIN) HMIN=X                                                  DY1   40
      IF (N.EQ.4HHMAX) HMAX=X                                                  DY1   41
      IF (N.EQ.4HTIME) TMAX=X                                                  DY1   42
      IF (N.EQ.8HPRINTING) NPR=X                                               DY1   43
      IF (N.EQ.9HTOLERANCE) EPS=X                                              DY1   44
      IF (N.EQ.5HORDER) IORDER=X                                               DY1   45
      IF (N.EQ.8HLINEPLOT) NPOL=1                                              DY1   46
      IF (N.EQ.5HGRAPH) NGRAPH=1                                               DY1   47
```

```
      IF (N.EQ.8HNONSTIFF) ISTIFF=0                                    DY1  48
      IF (N.EQ.9HNOMESSAGE) NOMES=1                                    DY1  49
      IF (N.EQ.7HLIBRARY) GO TO 3                                      DY1  50
      IF (N.EQ.7HPROCESS) GO TO 6                                      DY1  51
      IF (N.EQ.8HFUNCTION) GO TO 5                                     DY1  52
      GO TO 2                                                          DY1  53
3     NLIB=X                                                           DY1  54
      DO 4 I=1,NLIB                                                    DY1  55
      CALL GET (N,X,0)                                                 DY1  56
      J=X                                                              DY1  57
      ITAG(J)=N                                                        DY1  58
4     CONTINUE                                                         DY1  59
      GO TO 2                                                          DY1  60
5     NFN=X                                                            DY1  61
      NC5=NCOMP+5                                                      DY1  62
      READ 39, ((FN(I,J),J=1,NC5),I=1,NFN)                             DY1  63
      PRINT 39, ((FN(I,J),J=1,NC5),I=1,NFN)                            DY1  64
      GO TO 2                                                          DY1  65
6     CONTINUE                                                         DY1  66
      NMP=NMP+2                                                        DY1  67
      MAXNMP=NMP+1                                                     DY1  68
      NC5=NCOMP+5                                                      DY1  69
      WRITE (6,30)                                                     DY1  70
      DO 9 I=1,MAXNE                                                   DY1  71
      DO 7 J=1,MAXNMP                                                  DY1  72
      MP(I,J)=0                                                        DY1  73
7     CONTINUE                                                         DY1  74
      DO 8 J=1,5                                                       DY1  75
      EP(I,J)=0                                                        DY1  76
8     CONTINUE                                                         DY1  77
9     CONTINUE                                                         DY1  78
      NE=0                                                             DY1  79
      NEX=1                                                            DY1  80
      TIME=0.0                                                         DY1  81
10    CALL GET (NM,X,0)                                                DY1  82
11    IF (NM.EQ.3HEND) GO TO 16                                        DY1  83
      DO 12 I=1,30                                                     DY1  84
      IF (NM.EQ.ITAG(I)) GO TO 13                                      DY1  85
12    CONTINUE                                                         DY1  86
      WRITE (6,31) NM                                                  DY1  87
      I=0                                                              DY1  88
13    NT=I                                                             DY1  89
      NE=NE+1                                                          DY1  90
      READ (5,32) (AM(I),I=3,NMP)                                      DY1  91
      DO 14 J=3,NMP                                                    DY1  92
      MP(NE,J)=AM(J)                                                   DY1  93
14    CONTINUE                                                         DY1  94
      MP(NE,1)=X                                                       DY1  95
      MP(NE,2)=NT                                                      DY1  96
      WRITE (6,33) (MP(NE,I),I=1,NMP)                                  DY1  97
      READ (5,32) (EP(NE,I),I=1,5)                                     DY1  98
      WRITE (6,34) (EP(NE,I),I=1,5)                                    DY1  99
```

```
      CALL GET (NM,X,0)                                                 DY1 100
      IF (NM.NE.5HEXTRA) GO TO 15                                       DY1 101
      NNX=X                                                             DY1 102
      NMX=NNX+NEX-1                                                     DY1 103
      READ (5,32) (EX(J),J=NEX,NMX)                                     DY1 104
      WRITE (6,34) (EX(J),J=NEX,NMX)                                    DY1 105
      MP(NE,NMP+1)=NEX                                                  DY1 106
      NEX=NEX+NNX                                                       DY1 107
      CALL GET (NM,X,1)                                                 DY1 108
   15 IF (NM.NE.7HRESERVE) GO TO 11                                     DY1 109
      NNX=X                                                             DY1 110
      NEX=NEX+NNX                                                       DY1 111
      GO TO 10                                                          DY1 112
   16 CALL FETCH (7HSTREAMS,NS)                                         DY1 113
      WRITE (6,35) NS                                                   DY1 114
      DO 18 I=1,NS                                                      DY1 115
      S(1,I,1)=I                                                        DY1 116
      S(1,I,2)=-1.0                                                     DY1 117
      S(1,I,3)=0.0                                                      DY1 118
      S(1,I,4)=530.0                                                    DY1 119
      S(1,I,5)=14.7                                                     DY1 120
      DO 17 J=6,NC5                                                     DY1 121
      S(1,I,J)=0                                                        DY1 122
   17 CONTINUE                                                          DY1 123
   18 CONTINUE                                                          DY1 124
   19 CALL GET (NM,X,-1)                                                DY1 125
      IF (NM.EQ.8HEXPLICIT) GO TO 20                                    DY1 126
      IF (NM.EQ.3HEND) GO TO 25                                         DY1 127
      IF (NM.EQ.4HSPEC) GO TO 22                                        DY1 128
      GO TO 19                                                          DY1 129
   20 READ (5,36) IT,(AM(I),I=1,5)                                      DY1 130
      IF (IT.EQ.3HEND) GO TO 25                                         DY1 131
      READ (5,32) (AM(I),I=6,NC5)                                       DY1 132
      N=AM(1)                                                           DY1 133
      DO 21 I=1,NC5                                                     DY1 134
      S(1,N,I)=AM(I)                                                    DY1 135
   21 CONTINUE                                                          DY1 136
      GO TO 20                                                          DY1 137
   22 N1=X                                                              DY1 138
      READ (5,32) (AM(I),I=1,NC5)                                       DY1 139
      N2=AM(1)                                                          DY1 140
      DO 24 I=N1,N2                                                     DY1 141
      DO 23 J=2,NC5                                                     DY1 142
      S(1,I,J)=AM(J)                                                    DY1 143
   23 CONTINUE                                                          DY1 144
      S(1,I,1)=I                                                        DY1 145
   24 CONTINUE                                                          DY1 146
      GO TO 19                                                          DY1 147
   25 DO 26 I=1,NS                                                      DY1 148
      K=S(1,I,2)                                                        DY1 149
      WRITE (6,38) I,K,(S(1,I,J),J=3,5)                                 DY1 150
      WRITE (6,37) (S(1,I,J),J=6,NC5)                                   DY1 151
```

```
      DO 26 J=1,NC5                                                    DY1 152
   26 S(2,I,J)=S(1,I,J)                                                DY1 153
      XX=PROPS(0,0)                                                    DY1 154
      RETURN                                                           DY1 155
C                                                                      DY1 156
   27 FORMAT (1H1)                                                     DY1 157
   28 FORMAT (1X,8A10)                                                 DY1 158
   29 FORMAT (8A10)                                                    DY1 159
   30 FORMAT (1H1)                                                     DY1 160
   31 FORMAT (18H ***FAIL***  NAME ,A12,8H NOT SET)                    DY1 161
   32 FORMAT (12X,5F12.0)                                              DY1 162
   33 FORMAT (/,9H   UNIT =,I3,9H=  TYPE =,I3,1H=,12I5)                DY1 163
   34 FORMAT (11X,5F14.5)                                              DY1 164
   35 FORMAT (1H1,I6,8H STREAMS)                                       DY1 165
   36 FORMAT (A3,9X,5F12.0)                                            DY1 166
   37 FORMAT (8X,4F14.5)                                               DY1 167
   38 FORMAT (11H   STREAM =,I3,4H=  (,I3,1H),3F14.5)                  DY1 168
   39 FORMAT (12X,5F12.2)                                              DY1 169
      END                                                              DY1 170-
```

```
      SUBROUTINE DYN2                                                    DY2   1
      COMMON /UNIT/ IM,NMP                                               DY2   2
      COMMON /MAT/ MP(15,5),EP(15,5),S(2,16,7),EX(1)                     DY2   3
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH      DY2   4
      COMMON /BKV/ NBV                                                   DY2   5
      COMMON /CNTR/ NCTR                                                 DY2   6
      COMMON /LAG/ NSX,NSW                                               DY2   7
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,ISTIFF           DY2   8
      NBV=0                                                              DY2   9
      NSX=0                                                              DY2  10
      NCTR=0                                                             DY2  11
      DO 31 IM=1,NE                                                      DY2  12
      IF (ICONV.EQ.1) RETURN                                             DY2  13
      NTYPE=MP(IM,2)                                                     DY2  14
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23DY2  15
     1,24,25,26,27,28,29,30), NTYPE                                      DY2  16
                                                                         DY2  17
1     CALL TYPE1                                                         DY2  18
      GO TO 31                                                           DY2  19
2     CALL TYPE2                                                         DY2  20
      GO TO 31                                                           DY2  21
3     CALL TYPE3                                                         DY2  22
      GO TO 31                                                           DY2  23
4     CALL TYPE4                                                         DY2  24
      GO TO 31                                                           DY2  25
5     CALL TYPE5                                                         DY2  26
      GO TO 31                                                           DY2  27
6     CALL TYPE6                                                         DY2  28
      GO TO 31                                                           DY2  29
7     CALL TYPE7                                                         DY2  30
      GO TO 31                                                           DY2  31
8     CALL TYPE8                                                         DY2  32
      GO TO 31                                                           DY2  33
9     CALL TYPE9                                                         DY2  34
      GO TO 31                                                           DY2  35
10    CALL TYPE10                                                        DY2  36
      GO TO 31                                                           DY2  37
11    CALL TYPE11                                                        DY2  38
      GO TO 31                                                           DY2  39
12    CALL TYPE12                                                        DY2  40
      GO TO 31                                                           DY2  41
13    CALL TYPE13                                                        DY2  42
      GO TO 31                                                           DY2  43
14    CALL TYPE14                                                        DY2  44
      GO TO 31                                                           DY2  45
15    CALL TYPE15                                                        DY2  46
      GO TO 31                                                           DY2  47
16    CALL TYPE16                                                        DY2  48
      GO TO 31                                                           DY2  49
17    CALL TYPE17                                                        DY2  50
      GO TO 31                                                           DY2  51
18    CALL TYPE18                                                        DY2  52
      GO TO 31
```

```
19      CALL TYPE19
        GO TO 31
20      CALL TYPE20
        GO TO 31
21      CALL TYPE21
        GO TO 31
22      CALL TYPE22
        GO TO 31
23      CALL TYPE23
        GO TO 31
24      CALL TYPE24
        GO TO 31
25      CALL TYPE25
        GO TO 31
26      CALL TYPE26
        GO TO 31
27      CALL TYPE27
        GO TO 31
28      CALL TYPE28
        GO TO 31
29      CALL TYPE29
        GO TO 31
30      CALL TYPE30
31      CONTINUE
        RETURN
        END
```

```
DY2   53
DY2   54
DY2   55
DY2   56
DY2   57
DY2   58
DY2   59
DY2   60
DY2   61
DY2   62
DY2   63
DY2   64
DY2   65
DY2   66
DY2   67
DY2   68
DY2   69
DY2   70
DY2   71
DY2   72
DY2   73
DY2   74
DY2   75
DY2   76
DY2   77
DY2   78-
```

```
      SUBROUTINE DIFSUB (N,YY,DERY)                                          DIF    1
      COMMON /GEAR1/ Y(8,222),SAVE(10,222),ERROR(222),YMAX(222),DERY1(222DIF    2
     12),DERY2(222)                                                          DIF    3
      COMMON /GEAR2/ EPS,TIME,KFLAG,JSTART,NBVMAX,ICONV,ISTIFF               DIF    4
      COMMON /CON/ IG,NCOMP,NCS,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH          DIF    5
      COMMON /GEAR3/ HMIN,HMAX,NOMES                                         DIF    6
      COMMON /BKV/ NBV                                                       DIF    7
      COMMON /COLUMN/ JCOL(10,10)                                            DIF    8
      COMMON /JACOB/ PW(10,10)                                               DIF    9
      COMMON /SUBDI/ ASUB(49)                                                DIF   10
      COMMON /DIAG/ BDIAG(49)                                                DIF   11
      COMMON /SUPERD/ CSUP(49)                                               DIF   12
      COMMON /MODULE/ IDERY,ITER,ITRI,MC,IPIVOT                              DIF   13
      DIMENSION IRC(2), ICC(2)                                               DIF   14
      DIMENSION PWSAVE(2,2), JCOLS(2,2)                                      DIF   15
      DIMENSION A(8), PERTST(7,2,3)                                          DIF   16
      DIMENSION YY(222), DERY(222)                                           DIF   17
      DATA MR/10/                                                            DIF   18
      DATA ITRI,IDERY/2*0/                                                   DIF   19
      DATA PERTST/2.0,4.5,7.333,10.42,13.7,17.15,1.0,2.0,12.0,24.0,37.89DIF   20
     1,53.33,70.08,87.97,3.0,6.0,9.167,12.5,15.98,1.0,1.0,12.0,24.0,37.8DIF   21
     29,53.33,70.08,87.97,1.0,1.0,1.0,0.5,0.1667,0.04133,0.008267,1.0,1.DIF   22
     30,1.0,2.0,1.0,0.3157,0.07407,0.0139/                                   DIF   23
      DATA A(2)/-1.0/                                                        DIF   24
      IF (IDERY.EQ.1) GO TO 47                                               DIF   25
      NBV1=NBV+1                                                             DIF   26
      NBV=NBV+N                                                              DIF   27
      IF (NBVMAX.LT.NBV) NBVMAX=NBV                                          DIF   28
      DO 1 I=NBV1,NBV                                                        DIF   29
      II=I-NBV1+1                                                            DIF   30
      DERY1(I)=DERY(II)                                                      DIF   31
1     CONTINUE                                                               DIF   32
      IF (IG.EQ.1) GO TO 26                                                  DIF   33
C                                                                            DIF   34
C     PREDICTOR SECTION                                                      DIF   35
C                                                                            DIF   36
      IRET=1                                                                 DIF   37
      IF (JSTART.EQ.0) GO TO 4                                               DIF   38
2     DO 3 I=NBV1,NBV                                                        DIF   39
      DO 3 J=1,K                                                             DIF   40
3     SAVE(J,I)=Y(J,I)                                                       DIF   41
      HOLD=H                                                                 DIF   42
      NQOLD=NQ                                                               DIF   43
      RACUM=1.0                                                              DIF   44
      IF (JSTART.GT.0) GO TO 23                                              DIF   45
      GO TO 6                                                                DIF   46
4     CONTINUE                                                               DIF   47
      NQ=1                                                                   DIF   48
      N3=N                                                                   DIF   49
      N4=N*N                                                                 DIF   50
      DO 5 I=NBV1,NBV                                                        DIF   51
      II=I-NBV1+1                                                            DIF   52
```

```
      Y(1,I)=YY(II)                                                    DIF  53
      Y(2,I)=DERY1(I)*H                                                DIF  54
5     CONTINUE                                                         DIF  55
      HNEW=H                                                           DIF  56
      K=2                                                              DIF  57
      GO TO 2                                                          DIF  58
6     IF (ISTIFF.EQ.0) GO TO 7                                         DIF  59
      IF (NQ.GT.6.OR.NQ.LT.1) GO TO 8                                  DIF  60
      GO TO (16,17,18,19,20,21), NQ                                    DIF  61
7     IF (NQ.GT.7.OR.NQ.LT.1) GO TO 8                                  DIF  62
      GO TO (9,10,11,12,13,14,15), NQ                                  DIF  63
8     PRINT 79, NQ                                                     DIF  64
      STOP                                                             DIF  65
C                                                                      DIF  66
C     NONSTIFF COEFFICIENTS,ORDER 1-7                                  DIF  67
C                                                                      DIF  68
9     A(1)=-1.0                                                        DIF  69
      GO TO 22                                                         DIF  70
10    A(1)=-0.500000000                                                DIF  71
      A(3)=-0.500000000                                                DIF  72
      GO TO 22                                                         DIF  73
11    A(1)=-0.41666666666667                                           DIF  74
      A(3)=-0.750000000                                                DIF  75
      A(4)=-0.16666666666667                                           DIF  76
      GO TO 22                                                         DIF  77
12    A(1)=-0.375000000                                                DIF  78
      A(3)=-0.91666666666667                                           DIF  79
      A(4)=-0.33333333333333                                           DIF  80
      A(5)=-0.41666666666667E-01                                       DIF  81
      GO TO 22                                                         DIF  82
13    A(1)=-0.34861111111111                                           DIF  83
      A(3)=-1.0416666666667                                            DIF  84
      A(4)=-0.48611111111111                                           DIF  85
      A(5)=-0.10416666666667                                           DIF  86
      A(6)=-0.83333333333333E-02                                       DIF  87
      GO TO 22                                                         DIF  88
14    A(1)=-0.32986111111111                                           DIF  89
      A(3)=-1.1416666666667                                            DIF  90
      A(4)=-0.625000000                                                DIF  91
      A(5)=-0.17708333333333                                           DIF  92
      A(6)=-0.025000000                                                DIF  93
      A(7)=-0.13888888888889E-02                                       DIF  94
      GO TO 22                                                         DIF  95
15    A(1)=-0.31559193121693                                           DIF  96
      A(3)=-1.235000000                                                DIF  97
      A(4)=-0.75185185185                                              DIF  98
      A(5)=-0.25520833333333                                           DIF  99
      A(6)=-0.48611111111111E-01                                       DIF 100
      A(7)=-0.48611111111111E-02                                       DIF 101
      A(8)=-0.1984126984127E-03                                        DIF 102
      GO TO 22                                                         DIF 103
C                                                                      DIF 104
```

```
C     STIFF COEFFICIENTS,ORDER 1-6                                    DIF 105
C                                                                     DIF 106
      A(1)=-1.000000000                                               DIF 107
      GO TO 22                                                        DIF 108
17    A(1)=-0.66666666666667                                          DIF 109
      A(3)=-0.33333333333333                                          DIF 110
      GO TO 22                                                        DIF 111
18    A(1)=-0.54545454545455                                          DIF 112
      A(3)=A(1)                                                       DIF 113
      A(4)=-0.90909090909091E-01                                      DIF 114
      GO TO 22                                                        DIF 115
19    A(1)=-0.480000000                                               DIF 116
      A(3)=-0.700000000                                               DIF 117
      A(4)=-0.200000000                                               DIF 118
      A(5)=-0.020000000                                               DIF 119
      GO TO 22                                                        DIF 120
20    A(1)=-0.43795620437956                                          DIF 121
      A(3)=-0.82116788321169                                          DIF 122
      A(4)=-0.31021897810219                                          DIF 123
      A(5)=-0.54744525547445E-01                                      DIF 124
      A(6)=-0.36496350364964E-02                                      DIF 125
      GO TO 22                                                        DIF 126
21    A(1)=-0.40816326530612                                          DIF 127
      A(3)=-0.92063492063492                                          DIF 128
      A(4)=-0.41666666666666                                          DIF 129
      A(5)=-0.99206349206349E-01                                      DIF 130
      A(6)=-0.11904761904762E-01                                      DIF 131
      A(7)=-0.56689342403628E-03                                      DIF 132
22    K=NQ+1                                                          DIF 133
      IDOUB=K                                                         DIF 134
      MTYP=2-ISTIFF                                                   DIF 135
      ENQ2=0.5/FLOAT(NQ+1)                                            DIF 136
      ENQ3=0.5/FLOAT(NQ+2)                                            DIF 137
      ENQ1=0.5/FLOAT(NQ)                                              DIF 138
      EUP=(PERTST(NQ,MTYP,2)*EPS)**2                                  DIF 139
      E=(PERTST(NQ,MTYP,1)*EPS)**2                                    DIF 140
      EDWN=(PERTST(NQ,MTYP,3)*EPS)**2                                 DIF 141
      IF (EDWN.EQ.0) GO TO 78                                         DIF 142
      IF (IRET.EQ.2) GO TO 70                                         DIF 143
      IF (IRET.EQ.3) RETURN                                           DIF 144
23    CONTINUE                                                        DIF 145
      DO 24 J=2,K                                                     DIF 146
      DO 24 J1=J,K                                                    DIF 147
      J2=K-J1+J-1                                                     DIF 148
      DO 24 I=NBV1,NBV                                                DIF 149
24    Y(J2,I)=Y(J2,I)+Y(J2+1,I)                                       DIF 150
      DO 25 I=1,N                                                     DIF 151
      II=I+NBV1-1                                                     DIF 152
      YY(I)=Y(1,II)                                                   DIF 153
25    CONTINUE                                                        DIF 154
      RETURN                                                          DIF 155
26    CONTINUE                                                        DIF 156
```

```
C                                                                       DIF 157
C     CORRECTOR SECTION                                                 DIF 158
C                                                                       DIF 159
      DO 27 I=NBVI,NBV                                                  DIF 160
      ERROR(I)=0.0                                                      DIF 161
27    CONTINUE                                                          DIF 162
      IWEVAL=ISTIFF                                                     DIF 163
      L=1                                                               DIF 164
28    CONTINUE                                                          DIF 165
C                                                                       DIF 166
C     ITER=0 - DIRECT ITERATION OF CORRECTOR                            DIF 167
C     ITER=1 - NEWTON-RAPHSON ITERATION OF CORRECTOR                    DIF 168
C                                                                       DIF 169
      IF (ITER.EQ.0.OR.ISTIFF.EQ.0) GO TO 36                            DIF 170
      R=A(1)*H                                                          DIF 171
      IF (ITRI.EQ.1) GO TO 38                                           DIF 172
C                                                                       DIF 173
C     SIMULT OPTION                                                     DIF 174
C                                                                       DIF 175
      IF (IWEVAL.EQ.-1) GO TO 31                                        DIF 176
      DO 30 J=1,MC                                                      DIF 177
      DO 29 I=1,N                                                       DIF 178
      PW(I,J)=PW(I,J)*R                                                 DIF 179
      IF (JCOL(I,J).EQ.I) PW(I,J)=1.0+PW(I,J)                           DIF 180
      PWSAVE(I,J)=PW(I,J)                                               DIF 181
      JCOLS(I,J)=JCOL(I,J)                                              DIF 182
29    CONTINUE                                                          DIF 183
30    CONTINUE                                                          DIF 184
      IWEVAL=-1                                                         DIF 185
31    DO 32 I=NBV1,NBV                                                  DIF 186
      II=I-NBV1+1                                                       DIF 187
      DERY2(II)=Y(2,I)-DERY1(I)*H                                       DIF 188
32    CONTINUE                                                          DIF 189
      CALL SIMULT (N,MR,MC,PW,DERY2,JCOL,1.0E-10,ICC,IRC,IPIVOT)        DIF 190
      DO 33 I=1,N                                                       DIF 191
      II=I+NBV1-1                                                       DIF 192
      SAVE(9,II)=PW(I,1)                                                DIF 193
33    CONTINUE                                                          DIF 194
      DO 35 J=1,MC                                                      DIF 195
      DO 34 I=1,N                                                       DIF 196
      PW(I,J)=PWSAVE(I,J)                                               DIF 197
      JCOL(I,J)=JCOLS(I,J)                                              DIF 198
34    CONTINUE                                                          DIF 199
35    CONTINUE                                                          DIF 200
      GO TO 44                                                          DIF 201
C                                                                       DIF 202
C     DIRECT ITERATION OF CORRECTOR                                     DIF 203
C                                                                       DIF 204
36    CONTINUE                                                          DIF 205
      DO 37 I=NBV1,NBV                                                  DIF 206
      SAVE(9,I)=Y(2,I)-DERY1(I)*H                                       DIF 207
37    CONTINUE                                                          DIF 208
```

```
      GO TO 44                                                          DIF 209
C                                                                       DIF 210
C     TRIDIAGONAL OPTION                                                DIF 211
C                                                                       DIF 212
C                                                                       DIF 213
38    CONTINUE                                                          DIF 214
      IF (IWEVAL.EQ.-1) GO TO 41                                        DIF 215
      DO 39 I=1,N                                                       DIF 216
      ASUB(I)=ASUB(I)*R                                                 DIF 217
      BDIAG(I)=BDIAG(I)*R                                               DIF 218
      CSUP(I)=CSUP(I)*R                                                 DIF 219
39    CONTINUE                                                          DIF 220
      DO 40 I=1,N                                                       DIF 221
      BDIAG(I)=1.0+BDIAG(I)                                             DIF 222
40    CONTINUE                                                          DIF 223
      IWEVAL=-1                                                         DIF 224
41    CONTINUE                                                          DIF 225
      DO 42 I=NBV1,NBV                                                  DIF 226
      II=I-NBV1+1                                                       DIF 227
      DERY2(II)=Y(2,I)-DERY1(I)*H                                       DIF 228
42    CONTINUE                                                          DIF 229
      CALL TRIDAG (N,ASUB,BDIAG,CSUP,DERY2,YY,DERY,PWSAVE)              DIF 230
      DO 43 I=1,N                                                       DIF 231
      SAVE(9,I)=YY(I)                                                   DIF 232
43    CONTINUE                                                          DIF 233
      ITRI=0                                                            DIF 234
44    NT=N                                                              DIF 235
      BND=EPS*ENQ3/FLOAT(N)                                             DIF 236
      DO 45 I=NBV1,NBV                                                  DIF 237
      Y(1,I)=Y(1,I)+A(1)*SAVE(9,I)                                      DIF 238
      Y(2,I)=Y(2,I)-SAVE(9,I)                                           DIF 239
      ERROR(I)=ERROR(I)+SAVE(9,I)                                       DIF 240
      IF (ABS(SAVE(9,I)).LE.(BND*YMAX(I))) NT=NT-1                      DIF 241
45    CONTINUE                                                          DIF 242
      DO 46 I=1,N                                                       DIF 243
      II=I+NBV1-1                                                       DIF 244
      YY(I)=Y(1,II)                                                     DIF 245
46    CONTINUE                                                          DIF 246
      IF (NT.LE.0) GO TO 51                                             DIF 247
      IF (L.EQ.3) GO TO 49                                              DIF 248
C                                                                       DIF 249
C     RETURN TO MODULE TO GET NEW DERIVATIVE VALUES                     DIF 250
C                                                                       DIF 251
      IDERY=1                                                           DIF 252
      RETURN                                                            DIF 253
47    IDERY=0                                                           DIF 254
      DO 48 I=NBV1,NBV                                                  DIF 255
      II=I-NBV1+1                                                       DIF 256
      DERY1(I)=DERY(II)                                                 DIF 257
48    CONTINUE                                                          DIF 258
      L=L+1                                                             DIF 259
      IF (L.LT.4) GO TO 28                                              DIF 260
49    CONTINUE
```

```
      ICONV=1                                                          DIF 261
      RACUM=RACUM*0.25                                                 DIF 262
      IF (NOMES.EQ.0) PRINT 80, NBV1,NBV                               DIF 263
      IF (H.LE.(HMIN*1.00001)) GO TO 50                                DIF 264
      GO TO 75                                                         DIF 265
50    PRINT 81, NBV1,NBV                                               DIF 266
      STOP                                                             DIF 267
51    CONTINUE                                                         DIF 268
      IF (NBV.NE.NBVMAX) RETURN                                        DIF 269
C                                                                      DIF 270
C     ERROR SECTION                                                    DIF 271
C                                                                      DIF 272
      D=0.0                                                            DIF 273
      DO 52 I=1,NBVMAX                                                 DIF 274
      D=D+(ERROR(I)/YMAX(I))**2                                        DIF 275
52    CONTINUE                                                         DIF 276
      IF (D.GT.E) GO TO 56                                             DIF 277
      IF (K.LT.3) GO TO 54                                             DIF 278
      DO 53 J=3,K                                                      DIF 279
      DO 53 I=1,NBVMAX                                                 DIF 280
53    Y(J,I)=Y(J,I)+A(J)*ERROR(I)                                      DIF 281
54    CONTINUE                                                         DIF 282
      IF (IDOUB.LE.1) GO TO 57                                         DIF 283
      IDOUB=IDOUB-1                                                    DIF 284
      IF (IDOUB.GT.1) GO TO 72                                         DIF 285
      DO 55 I=1,NBVMAX                                                 DIF 286
      SAVE(10,I)=ERROR(I)                                              DIF 287
55    CONTINUE                                                         DIF 288
      GO TO 72                                                         DIF 289
56    KFLAG=KFLAG-2                                                    DIF 290
      IF (NOMES.EQ.0) PRINT 82                                         DIF 291
      IF (H.LE.(HMIN*1.00001)) GO TO 74                                DIF 292
57    PR2=(D/E)**ENQ2*1.2                                              DIF 293
      PR3=1.0E+20                                                      DIF 294
      IF ((NQ.GE.IORDER).OR.(KFLAG.LE.-1)) GO TO 59                    DIF 295
      D=0.0                                                            DIF 296
      DO 58 I=1,NBVMAX                                                 DIF 297
      D=D+((ERROR(I)-SAVE(10,I))/YMAX(I))**2                           DIF 298
58    CONTINUE                                                         DIF 299
      PR3=(D/EUP)**ENQ3*1.4                                            DIF 300
59    PR1=1.0E+20                                                      DIF 301
      IF (NQ.LE.1) GO TO 61                                            DIF 302
      D=0.0                                                            DIF 303
      DO 60 I=1,NBVMAX                                                 DIF 304
      D=D+(Y(K,I)/YMAX(I))**2                                          DIF 305
60    CONTINUE                                                         DIF 306
      PR1=(D/EDWN)**ENQ1*1.3                                           DIF 307
61    CONTINUE                                                         DIF 308
      IF (PR2.LE.PR3) GO TO 67                                         DIF 309
      IF (PR3.LT.PR1) GO TO 68                                         DIF 310
62    R=1.0/AMAX1(PR1,1.0E-04)                                         DIF 311
      NEWQ=NQ-1                                                        DIF 312
```

```
63      IDOUB=10                                                        DIF 313
        IF ((KFLAG.EQ.1).AND.(R.LT.1.1)) GO TO 72                       DIF 314
        IF (NEWQ.LE.NQ) GO TO 65                                        DIF 315
        DO 64 I=1,NBVMAX                                                DIF 316
        Y(NEWQ+1,I)=ERROR(I)*A(K)/FLOAT(K)                              DIF 317
64      CONTINUE                                                        DIF 318
65      K=NEWQ+1                                                        DIF 319
        IF (KFLAG.EQ.1) GO TO 69                                        DIF 320
        RACUM=RACUM*R                                                   DIF 321
        IRET1=3                                                         DIF 322
        GO TO 75                                                        DIF 323
66      CONTINUE                                                        DIF 324
        IRET1=0                                                         DIF 325
        IF (NEWQ.EQ.NQ) RETURN                                          DIF 326
        NQ=NEWQ                                                         DIF 327
        IRET=3                                                          DIF 328
        GO TO 6                                                         DIF 329
67      IF (PR2.GT.PR1) GO TO 62                                        DIF 330
        NEWQ=NQ                                                         DIF 331
        R=1.0/AMAX1(PR2,1.0E-04)                                        DIF 332
        GO TO 63                                                        DIF 333
68      R=1.0/AMAX1(PR3,1.0E-04)                                        DIF 334
        NEWQ=NQ+1                                                       DIF 335
        GO TO 63                                                        DIF 336
69      CONTINUE                                                        DIF 337
        R=AMIN1(R,HMAX/ABS(H))                                          DIF 338
        H=H*R                                                           DIF 339
        IF (NQ.EQ.NEWQ) GO TO 70                                        DIF 340
        NQ=NEWQ                                                         DIF 341
        IRET=2                                                          DIF 342
        GO TO 6                                                         DIF 343
70      R1=1.0                                                          DIF 344
        DO 71 J=2,K                                                     DIF 345
        R1=R1*R                                                         DIF 346
        DO 71 I=1,NBVMAX                                                DIF 347
71      Y(J,I)=Y(J,I)*R1                                                DIF 348
        IDOUB=K                                                         DIF 349
72      CONTINUE                                                        DIF 350
        DO 73 I=1,NBVMAX                                                DIF 351
        YMAX(I)=AMAX1(YMAX(I),ABS(Y(1,I)))                              DIF 352
73      CONTINUE                                                        DIF 353
        JSTART=NQ                                                       DIF 354
        RETURN                                                          DIF 355
74      PRINT 83                                                        DIF 356
        STOP                                                            DIF 357
75      CONTINUE                                                        DIF 358
        RACUM=AMAX1(ABS(HMIN/HOLD),RACUM)                               DIF 359
        RACUM=AMIN1(RACUM,ABS(HMAX/HOLD))                               DIF 360
        H=HOLD*RACUM                                                    DIF 361
        R1=1.0                                                          DIF 362
        DO 76 J=2,K                                                     DIF 363
        R1=R1*RACUM                                                     DIF 364
```

```
      DO 76 I=1,NBVMAX                                                  DIF 365
76    Y(J,I)=SAVE(J,I)*R1                                               DIF 366
      DO 77 I=1,NBVMAX                                                  DIF 367
      Y(1,I)=SAVE(1,I)                                                  DIF 368
77    CONTINUE                                                          DIF 369
      IDOUB=K                                                           DIF 370
      IF (IRET1.EQ.3) GO TO 66                                          DIF 371
      RETURN                                                            DIF 372
78    PRINT 84                                                          DIF 373
      STOP                                                              DIF 374
C                                                                       DIF 375
79    FORMAT (41H THE MAXIMUM ORDER SPECIFIED IS TOO LARGE,I5)          DIF 376
80    FORMAT (49H THE CORRECTOR FAILED TO CONVERGE IN 3 ITERATIONS,/,23HDIF 377
     1 DIFFERENTIAL EQUATIONS,I4,3H TO,I4)                              DIF 378
81    FORMAT (58H CORRECTOR CONVERGENCE COULD NOT BE ACHIEVED FOR H.GT.HDIF 379
     1MIN,/,23H DIFFERENTIAL EQUATIONS,I4,3H TO,I4)                     DIF 380
82    FORMAT (30H TRUNCATION ERROR IS TOO LARGE)                        DIF 381
83    FORMAT (72H THE STEP WAS TAKEN WITH H=HMIN BUT THE REQUESTED ERRORDIF 382
     1 WAS NOT ACHIEVED)                                                DIF 383
84    FORMAT (68H THE REQUESTED ERROR IS SMALLER THAN CAN BE HANDLED FORDIF 384
     1 THIS PROBLEM)                                                    DIF 385
      END                                                               DIF 386
```

```
      SUBROUTINE SIMULT (NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,IPIVOT)          SIM   1
                                                                          SIM   2
C                                                                         SIM   3
C     ARGUMENTS                                                           SIM   4
C                                                                         SIM   5
C     NU    : NUMBER OF UNKNOWNS                                          SIM   6
C     MR    : MAXIMUM ROW DIMENSION OF MATRIX A IN CALLING PROGRAM        SIM   7
C     MC    : MAXIMUM NUMBER OF COLUMNS OF MATRIX A                       SIM   8
C     A     : MATRIX CONTAINING NONZERO ELEMENTS                          SIM   9
C     P     : RIGHT HAND SIDE VECTOR                                      SIM  10
C     ICOL  : RECORDS POSITION OF NONZERO ELEMENTS                        SIM  11
C     ICC,IRC : DIMENSIONED IN CALLING PROGRAM AND USED ONLY IN           SIM  12
C               SUBROUTINE SIMULT                                         SIM  13
C     IPIVOT : PIVOT OPTION 1-7                                           SIM  14
C              1 : SIMPLE GAUSS-JORDAN ELIMINATION                        SIM  15
C              2 : GAUSS-JORDAN PARTIAL PIVOTING                          SIM  16
C              3 : GAUSS-JORDAN FULL PIVOTING                             SIM  17
C              4 : MINIMUM ROW-MINIMUM COLUMN                             SIM  18
C              5 : MINIMUM COLUMN-MINIMUM ROW                             SIM  19
C              6 : MAXIMUM COLUMN-MINIMUM ROW                             SIM  20
C              7 : MINIMUM OF ROW ENTRIES TIMES COLUMN ENTRIES            SIM  21
C                                                                         SIM  22
C     SOLUTION VECTOR RETURNED IN FIRST COLUMN OF A : A(I,1)              SIM  23
C                                                                         SIM  24
      INTEGER PIVROW,PIVCOL,OPROW                                         SIM  25
      DIMENSION IRC(NU), ICC(NU), A(MR,MC), ICOL(MR,MC), P(NU)            SIM  26
C                                                                         SIM  27
C     INITIALIZE ROW ENTRY COUNTER                                        SIM  28
      DO 1 I=1,NU                                                         SIM  29
      IRC(I)=0                                                            SIM  30
1     CONTINUE                                                            SIM  31
C     COUNT ROW AND COLUMN ENTRIES                                        SIM  32
C     JELE=0                                                              SIM  33
      DO 3 I=1,NU                                                         SIM  34
      DO 2 J=1,MC                                                         SIM  35
      IC=ICOL(I,J)                                                        SIM  36
      IF (IC.EQ.0) GO TO 3                                                SIM  37
C     JELE=JELE+1                                                         SIM  38
      ICC(I)=J                                                            SIM  39
      IRC(IC)=IRC(IC)+1                                                   SIM  40
2     CONTINUE                                                            SIM  41
3     CONTINUE                                                            SIM  42
C     JCOL=0                                                              SIM  43
C     JELIM=0                                                             SIM  44
C     DO 800 I=1,NU                                                       SIM  45
C     800 IF(ICC(I).GT.JCOL) JCOL=ICC(I)                                  SIM  46
      LKJ=1                                                               SIM  47
      GO TO 26                                                            SIM  48
4     CONTINUE                                                            SIM  49
C     NORMALIZE PIVROW                                                    SIM  50
      X=A(PIVROW,IY)                                                      SIM  51
      IC=ICC(PIVROW)                                                      SIM  52
      DO 5 J=1,IC
```

```
      A(PIVROW,J)=A(PIVROW,J)/X                                          SIM  53
5     CONTINUE                                                           SIM  54
      A(PIVROW,IY)=1.0                                                   SIM  55
      P(PIVROW)=P(PIVROW)/X                                              SIM  56
C     SELECT ROWS THAT CAN BE OPERATED ON                                SIM  57
      DO 21 I=1,NU                                                       SIM  58
      IF (IRC(PIVCOL).EQ.1) GO TO 22                                     SIM  59
      IF (I.EQ.PIVROW) GO TO 21                                          SIM  60
      IC=IABS(ICC(I))                                                    SIM  61
      DO 20 J=1,IC                                                       SIM  62
      IF (ICOL(I,J)-PIVCOL) 20,6,21                                      SIM  63
C     IF YOU CAN GET TO THIS POINT OPROW CONTAINS PIVOTAL ELEMENT        SIM  64
6     OPROW=I                                                            SIM  65
      JKOP=1                                                             SIM  66
      JKPI=1                                                             SIM  67
      C=-A(OPROW,J)                                                      SIM  68
      P(OPROW)=P(PIVROW)*C+P(OPROW)                                      SIM  69
7     CONTINUE                                                           SIM  70
      IF (ICOL(PIVROW,JKPI).EQ.0) GO TO 21                               SIM  71
      IF (ICOL(OPROW,JKOP).EQ.0) GO TO 8                                 SIM  72
      IF (ICOL(PIVROW,JKPI)-ICOL(OPROW,JKOP)) 8,12,19                    SIM  73
C     OPROW DOES NOT CONTAIN THIS ELEMENT,ADD ELEMENT TO OPROW           SIM  74
8     ICC(I)=ICC(I)+1                                                    SIM  75
      IF (ICC(I).LE.0) ICC(I)=ICC(I)-2                                   SIM  76
      II=IABS(ICC(I))                                                    SIM  77
C     IF(II.GT.JCOL) JCOL=II                                             SIM  78
      IF (II.GT.MC) GO TO 9                                              SIM  79
      GO TO 10                                                           SIM  80
9     CONTINUE                                                           SIM  81
      PRINT 47, II                                                       SIM  82
      STOP                                                               SIM  83
10    CONTINUE                                                           SIM  84
      JKL=JKOP+1                                                         SIM  85
11    IX=II-1                                                            SIM  86
      A(OPROW,II)=A(OPROW,IX)                                            SIM  87
      ICOL(OPROW,II)=ICOL(OPROW,IX)                                      SIM  88
      II=IX                                                              SIM  89
      IF (II.GE.JKL) GO TO 11                                            SIM  90
      A(OPROW,JKOP)=A(PIVROW,JKPI)*C                                     SIM  91
      ICOL(OPROW,JKOP)=ICOL(PIVROW,JKPI)                                 SIM  92
      IX=ICOL(OPROW,JKOP)                                                SIM  93
      IRC(IX)=IRC(IX)+1                                                  SIM  94
      GO TO 18                                                           SIM  95
C     PIVROW AND OPROW CONTAIN THIS ELEMENT *SHIFT BOTH AND OPERATE ON   SIM  96
C     OPR                                                                SIM  97
12    IX=ICOL(OPROW,JKOP)                                                SIM  98
      IF (IX.EQ.PIVCOL) GO TO 13                                         SIM  99
      X=A(PIVROW,JKPI)*C+A(OPROW,JKOP)                                   SIM 100
      A(OPROW,JKOP)=X                                                    SIM 101
C     TEST OPROW TO SEE IF ANY ELEMENTS WERE ELIMINATED OTHER THAN       SIM 102
C     THOSE IN THE PIVOTAL COLUMN                                        SIM 103
      ATEST=ABS(X)-ZTEST                                                 SIM 104
```

```
      IF (ATEST.GT.0.) GO TO 18                                          SIM 105
   13 IRC(IX)=IRC(IX)-1                                                  SIM 106
C     JELIM=JELIM+1                                                      SIM 107
      ICC(OPROW)=ICC(OPROW)-1                                            SIM 108
      IF (ICC(OPROW)) 15,14,16                                           SIM 109
   14 GO TO 24                                                           SIM 110
   15 CONTINUE                                                           SIM 111
      ICC(OPROW)=ICC(OPROW)+2                                            SIM 112
   16 IX=IABS(ICC(OPROW))                                                SIM 113
      DO 17 NK=JKOP,IX                                                   SIM 114
      A(I,NK)=A(I,NK+1)                                                  SIM 115
      ICOL(I,NK)=ICOL(I,NK+1)                                            SIM 116
   17 CONTINUE                                                           SIM 117
      IX=IX+1                                                            SIM 118
      ICOL(I,IX)=0                                                       SIM 119
      JKPI=JKPI+1                                                        SIM 120
      GO TO 7                                                            SIM 121
   18 JKPI=JKPI+1                                                        SIM 122
C     PIVROW DOES NOT CONTAIN THIS ELEMENT*SHIFT OPROW AND CONTINUE      SIM 123
   19 JKOP=JKOP+1                                                        SIM 124
      GO TO 7                                                            SIM 125
   20 CONTINUE                                                           SIM 126
   21 CONTINUE                                                           SIM 127
   22 CONTINUE                                                           SIM 128
C     ELIMINATES PIVROW AND PIVCOL FROM BEING CONSIDERED AGAIN           SIM 129
      ICC(PIVROW)=-ICC(PIVROW)                                           SIM 130
      IRC(PIVCOL)=-IRC(PIVCOL)                                           SIM 131
      LKJ=LKJ+1                                                          SIM 132
      IF (LKJ.LE.NU) GO TO 26                                            SIM 133
C     UNSCRAMBLE AND STORE SOLUTION IN FIRST COLUMN OF A                 SIM 134
      DO 23 I=1,NU                                                       SIM 135
      II=ICOL(I,1)                                                       SIM 136
      A(II,1)=P(I)                                                       SIM 137
   23 CONTINUE                                                           SIM 138
      RETURN                                                             SIM 139
   24 CONTINUE                                                           SIM 140
C     SUBROUTINE SINGLR                                                  SIM 141
      IB=1                                                               SIM 142
      DO 25 M=1,NU                                                       SIM 143
      II=ICC(M)                                                          SIM 144
      IF (II.GT.IB) IB=II                                                SIM 145
   25 CONTINUE                                                           SIM 146
      WRITE (6,48) LKJ,PIVROW,PIVCOL,OPROW                               SIM 147
      WRITE (6,49) (M,IRC(M),ICC(M),M=1,NU)                              SIM 148
      WRITE (6,50)                                                       SIM 149
      CALL RITE (1,NU,IB,MR,MC,A,ICOL)                                   SIM 150
      WRITE (6,51)                                                       SIM 151
      CALL RITE (2,NU,IB,MR,MC,A,ICOL)                                   SIM 152
      STOP                                                               SIM 153
   26 CONTINUE                                                           SIM 154
      GO TO (27,28,30,33,36,40,44), IPIVOT                               SIM 155
   27 CONTINUE                                                           SIM 156
```

```
C                                                                       SIM 157
C     PIVOT OPTION 1 : SIMPLE GAUSS-JORDAN ELIMINATION                  SIM 158
C                                                                       SIM 159
C     SUBROUTINE PIVSEL                                                 SIM 160
C     GAUSS ELIMINATION                                                 SIM 161
      PIVROW=LKJ                                                        SIM 162
      PIVCOL=LKJ                                                        SIM 163
      IY=1                                                              SIM 164
C     END GAUSS ELIMINATION                                             SIM 165
C     RETURN                                                            SIM 166
      GO TO 4                                                           SIM 167
28    CONTINUE                                                          SIM 168
C                                                                       SIM 169
C     PIVOT OPTION 2 : GAUSS-JORDAN PARTIAL PIVOTING                    SIM 170
C                                                                       SIM 171
C     SUBROUTINE PIVSEL                                                 SIM 172
C     GAUSS JORDAN-PARTIAL PIVOTING                                     SIM 173
      PIVCOL=LKJ                                                        SIM 174
      ATEST=0.                                                          SIM 175
      DO 29 I=1,NU                                                      SIM 176
      IC=ICC(I)                                                         SIM 177
      IF (IC.LE.0) GO TO 29                                             SIM 178
      II=ICOL(I,1)                                                      SIM 179
      IF (II.GT.PIVCOL) GO TO 29                                        SIM 180
      AA=ABS(A(I,1))                                                    SIM 181
      IF (AA.LE.ATEST) GO TO 29                                         SIM 182
      ATEST=AA                                                          SIM 183
      PIVROW=I                                                          SIM 184
29    CONTINUE                                                          SIM 185
      IY=1                                                              SIM 186
C     END GAUSS JORDAN-PARTIAL PIVOTING                                 SIM 187
C     RETURN                                                            SIM 188
      GO TO 4                                                           SIM 189
30    CONTINUE                                                          SIM 190
C                                                                       SIM 191
C     PIVOT OPTION 3 : GAUSS-JORDAN FULL PIVOTING                       SIM 192
C                                                                       SIM 193
C     SUBROUTINE PIVSEL                                                 SIM 194
C     GAUSS JORDAN FULL PIVOTING                                        SIM 195
      ATEST=0.                                                          SIM 196
      DO 32 I=1,NU                                                      SIM 197
      IC=ICC(I)                                                         SIM 198
      IF (IC.LE.0) GO TO 32                                             SIM 199
      DO 31 J=1,IC                                                      SIM 200
      II=ICOL(I,J)                                                      SIM 201
      IR=IRC(II)                                                        SIM 202
      IF (IR.LE.0) GO TO 31                                             SIM 203
      AA=ABS(A(I,J))                                                    SIM 204
      IF (AA.LE.ATEST) GO TO 31                                         SIM 205
      ATEST=AA                                                          SIM 206
      PIVROW=I                                                          SIM 207
      PIVCOL=II                                                         SIM 208
```

```
      IY=J                                                              SIM 209
   31 CONTINUE                                                          SIM 210
   32 CONTINUE                                                          SIM 211
C     END GAUSS JORDAN FULL PIVOTING                                    SIM 212
C     RETURN                                                            SIM 213
      GO TO 4                                                           SIM 214
   33 CONTINUE                                                          SIM 215
C                                                                       SIM 216
C     PIVOT OPTION 4 : MINIMUM ROW-MINIMUM COLUMN                       SIM 217
C                                                                       SIM 218
C     SUBROUTINE PIVSEL                                                 SIM 219
C     SELECT FIRST MIN ROW THEN FIRST MIN COL                           SIM 220
C     SELECT ROW WITH MINIMUM ENTRIES                                   SIM 221
      IK=100000                                                         SIM 222
      DO 34 I=1,NU                                                      SIM 223
      IC=ICC(I)                                                         SIM 224
      IF (IC.GE.IK.OR.IC.LE.0) GO TO 34                                 SIM 225
      PIVROW=I                                                          SIM 226
      IK=IC                                                             SIM 227
   34 CONTINUE                                                          SIM 228
C     SELECT SMALLEST AVAILABLE COLUMN FROM PIVROW                      SIM 229
      IK=100000                                                         SIM 230
      IC=ICC(PIVROW)                                                    SIM 231
      DO 35 I=1,IC                                                      SIM 232
      II=ICOL(PIVROW,I)                                                 SIM 233
      IR=IRC(II)                                                        SIM 234
      IF (IR.GE.IK.OR.IR.LE.0) GO TO 35                                 SIM 235
      PIVCOL=II                                                         SIM 236
      IK=IR                                                             SIM 237
      IY=I                                                              SIM 238
   35 CONTINUE                                                          SIM 239
C     END FIRST MIN ROW THEN FIRST MIN COL                              SIM 240
C     RETURN                                                            SIM 241
      GO TO 4                                                           SIM 242
   36 CONTINUE                                                          SIM 243
C                                                                       SIM 244
C     PIVOT OPTION 5 : MINIMUM COLUMN-MINIMUM ROW                       SIM 245
C                                                                       SIM 246
C     SUBROUTINE PIVSEL                                                 SIM 247
C     SELECT FIRST MIN COL THEN FIRST MIN ROW                           SIM 248
      IK=100000                                                         SIM 249
      DO 37 I=1,NU                                                      SIM 250
      IR=IRC(I)                                                         SIM 251
      IF (IR.GE.IK.OR.IR.LE.0) GO TO 37                                 SIM 252
      PIVCOL=I                                                          SIM 253
      IK=IR                                                             SIM 254
   37 CONTINUE                                                          SIM 255
      IK=100000                                                         SIM 256
      DO 39 I=1,NU                                                      SIM 257
      IC=ICC(I)                                                         SIM 258
      IF (IC.LE.0) GO TO 39                                             SIM 259
      DO 38 J=1,IC                                                      SIM 260
```

```
      IF (ICOL(I,J).LT.PIVCOL) GO TO 38                                 SIM 261
      IF (ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GO TO 39                     SIM 262
      IK=IC                                                             SIM 263
      PIVROW=I                                                          SIM 264
      IY=J                                                              SIM 265
38    CONTINUE                                                          SIM 266
39    CONTINUE                                                          SIM 267
C     END SELECT FIRST MIN COL THEN FIRST MIN ROW                       SIM 268
C     RETURN                                                            SIM 269
      GO TO 4                                                           SIM 270
40    CONTINUE                                                          SIM 271
C                                                                       SIM 272
C     PIVOT OPTION 6 : MAXIMUM COLUMN-MINIMUM ROW                       SIM 273
C                                                                       SIM 274
C     SUBROUTINE PIVSEL                                                 SIM 275
C     SELECT FIRST MAX COL THEN FIRST MIN ROW                           SIM 276
      IK=-1                                                             SIM 277
      DO 41 I=1,NU                                                      SIM 278
      IR=IRC(I)                                                         SIM 279
      IF (IR.LE.IK.OR.IR.LE.0) GO TO 41                                 SIM 280
      PIVCOL=I                                                          SIM 281
      IK=IR                                                             SIM 282
41    CONTINUE                                                          SIM 283
      DO 43 I=1,NU                                                      SIM 284
      IC=ICC(I)                                                         SIM 285
      IF (IC.LE.0) GO TO 43                                             SIM 286
      DO 42 J=1,IC                                                      SIM 287
      IF (ICOL(I,J).LT.PIVCOL) GO TO 42                                 SIM 288
      IF (ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GO TO 43                     SIM 289
      IK=IC                                                             SIM 290
      PIVROW=I                                                          SIM 291
      IY=J                                                              SIM 292
42    CONTINUE                                                          SIM 293
43    CONTINUE                                                          SIM 294
C     END SELECT FIRST MAX COL THEN FIRST MIN ROW                       SIM 295
C     RETURN                                                            SIM 296
      GO TO 4                                                           SIM 297
44    CONTINUE                                                          SIM 298
C                                                                       SIM 299
C     PIVOT OPTION 7 : MINIMUM OF ROW ENTRIES TIMES COLUMN ENTRIES      SIM 300
C                                                                       SIM 301
C     SUBROUTINE PIVSEL                                                 SIM 302
C     SELECT FIRST MIN(ROW*COL)                                         SIM 303
      IK=100000                                                         SIM 304
      DO 46 I=1,NU                                                      SIM 305
      IC=ICC(I)                                                         SIM 306
      IF (IC.LE.0) GO TO 46                                             SIM 307
      DO 45 J=1,IC                                                      SIM 308
      II=ICOL(I,J)                                                      SIM 309
      IR=IRC(II)                                                        SIM 310
      IF (IR.LE.0) GO TO 45                                             SIM 311
      III=IC*IR                                                         SIM 312
```

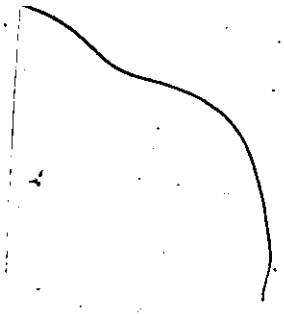
```
      IF (III.GE.IK) GO TO 45                                          SIM 313
      PIVROW=I                                                         SIM 314
      PIVCOL=II                                                        SIM 315
      IK=III                                                           SIM 316
      IY=J                                                             SIM 317
45    CONTINUE                                                         SIM 318
46    CONTINUE                                                         SIM 319
C     END SELECT FIRST MIN(ROW*COL)                                    SIM 320
C     RETURN                                                           SIM 321
      GO TO 4                                                          SIM 322
C                                                                      SIM 323
47    FORMAT (22H MC SHOULD BE AT LEAST,I3)                            SIM 324
48    FORMAT (*1  MATRIX SINGULAR*,//,*  NO. CYCLES COMPLETED=*,I5,*  PISIM 325
     1VROW=*,I5,*  PIVCOL=*I5,*  OPROW=*I5)                           SIM 326
49    FORMAT (*0 COL,/ROW NO.,NO. COL. ENTRIES,NO. ROW ENTRIES*/(6(I7,2ISIM 327
     15)))                                                            SIM 328
50    FORMAT (*1 COEFFICIENT MATRIX*)                                  SIM 329
51    FORMAT (*1  COLUMN IDENTIFICATION*)                              SIM 330
      END                                                              SIM 331-
```

```
      SUBROUTINE RITE (IDUM,NR,NC,MR,MC,A,ICOL)                          RIT    1.
      DIMENSION A(MR,MC), ICOL(MR,MC)                                    RIT    2
      IPRINT=12                                                          RIT    3
      IF (IDUM.NE.1) IPRINT=30                                           RIT    4
      IPR=IPRINT-1                                                       RIT    5
      DO 4 K=1,NC,IPRINT                                                 RIT    6
      MAX=K+IPR                                                          RIT    7
      IF (MAX.GT.NC) MAX=NC                                              RIT    8
      IF (K.NE.1) WRITE (6,7)                                            RIT    9
      IF (IDUM.EQ.1) GO TO 2                                             RIT   10
      WRITE (6,5) (I,I=K,MAX)                                            RIT   11
      DO 1 J=1,NR                                                        RIT   12
      WRITE (6,8) J,(ICOL(J,I),I=K,MAX)                                  RIT   13
1     CONTINUE                                                           RIT   14
      GO TO 4                                                            RIT   15
2     WRITE (6,6) (I,I=K,MAX)                                            RIT   16
      DO 3 J=1,NR                                                        RIT   17
      WRITE (6,9) J,(A(J,I),I=K,MAX)                                     RIT   18
3     CONTINUE                                                           RIT   19
4     CONTINUE                                                           RIT   20
      RETURN                                                             RIT   21
C                                                                        RIT   22
5     FORMAT (6X,30I4)                                                   RIT   23
6     FORMAT (6X,12I10)                                                  RIT   24
7     FORMAT (1H1)                                                       RIT   25
8     FORMAT (1X,I5,30I4)                                                RIT   26
9     FORMAT (1X,I5,12G10.3)                                             RIT   27
      END                                                                RIT   28.
```

```
      SUBROUTINE TYPE10                                                 T10    1
C                                                                       T10    2
CC    SUBROUTINE REAC1                                                  T10    3
CC                                                                      T10    4
CC    THIS MODULE REPRESENTS A CSTR WITH A FIRST ORDER REVERSIBLE       T10    5
CC    REACTION, 1 INPUT STREAM, 1 OUTPUT STREAM, CONSTANT TEMPERATURE.  T10    6
CC                                                                      T10    7
CC    EQUIPMENT PARAMETERS                                              T10    8
CC                                                                      T10    9
CC    1 - REACTOR VOLUME - FT**3                                        T10   10
CC    2 - K1 - FORWARD REACTION RATE CONSTANT - MIN**-1                 T10   11
CC    3 - K2 - BACKWARD REACTION RATE CONSTANT - MIN**-1                T10   12
CC    4 - ITER - ITER=0 - USE DIRECT ITERATION WITH STIFF OPTION        T10   13
CC               ITER=1 - USE NEWTON-RAPHSON ITERATION WITH STIFF OPTIONT10   14
CC               FOR NONSTIFF OPTION ITER IS NOT USED                 T10   15
CC                                                                      T10   16
      COMMON /MAT/ MP(15,5),EP(15,5),S(2,16,7),EX(1)                    T10   17
      COMMON /CON/ IG,NCOMP,NC5,H,NE,NS,NPR,NPOL,TMAX,IORDER,NGRAPH     T10   18
      COMMON /PTAB/ IGFLAG,PP(10,10)                                    T10   19
      COMMON /UNIT/ IM                                                  T10   20
      COMMON /COLUMN/ JCOL(10,10)                                       T10   21
      COMMON /JACOB/ XJACOB(10,10)                                      T10   22
      COMMON /MODULE/ IDERY,ITER,ITRI,MC,IPIVOT                         T10   23
      DIMENSION Y(2), DERY(2)                                           T10   24
      REAL K1,K2                                                        T10   25
C                                                                       T10   26
CC    CALCULATE MODULE PARAMETERS.                                      T10   27
CC                                                                      T10   28
C     GET VOLUME OF REACTOR FROM EP - VOL                               T10   29
      VOL=EP(IM,1)                                                      T10   30
C     GET REACTION RATE CONSTANTS FROM EP - K1,K2                       T10   31
      K1=EP(IM,2)                                                       T10   32
      K2=EP(IM,3)                                                       T10   33
C     GET ITERATION OPTION FROM EP                                      T10   34
      ITER=EP(IM,4)                                                     T10   35
C     GET STREAM NUMBER OF INPUT STREAM FROM MP - IN                    T10   36
      IN=MP(IM,3)                                                       T10   37
C     GET STREAM NUMBER OF OUTPUT STREAM FROM MP - IOUT                 T10   38
      IOUT=IABS(MP(IM,4))                                               T10   39
C     GET DENSITY OF INPUT STREAM FROM PP - DENS                        T10   40
      DENS=PP(1,2)                                                      T10   41
C     CALCULATE REACTOR TIME CONSTANT - TAU                             T10   42
      TAU=VOL/(S(IG,IN,3)/DENS)                                         T10   43
C     GET INITIAL REACTOR CONCENTRATIONS FROM OUTPUT STREAM             T10   44
      Y(1)=S(IG,IOUT,6)                                                 T10   45
      Y(2)=S(IG,IOUT,7)                                                 T10   46
C                                                                       T10   47
CC    CALCULATE JACOBIAN MATRIX ON CORRECTOR PASS                       T10   48
CC                                                                      T10   49
      IF (IG.EQ.2) GO TO 1                                              T10   50
      MC=2                                                              T10   51
      IPIVOT=3                                                          T10   52
```

```
      JCOL(1,1)=1                                                        T10  53
      JCOL(2,1)=1                                                        T10  54
      JCOL(1,2)=2                                                        T10  55
      JCOL(2,2)=2                                                        T10  56
      XJACOB(1,1)=-(K1+1.0/TAU)                                          T10  57
      XJACOB(2,1)=K1                                                     T10  58
      XJACOB(1,2)=K2                                                     T10  59
      XJACOB(2,2)=-(K2+1.0/TAU)                                          T10  60
1     CONTINUE                                                           T10  61
C                                                                        T10  62
C     CALCULATE DERIVATIVES                                              T10  63
C                                                                        T10  64
C     MASS BALANCE FOR COMPONENT A (UNITS=MIN**-1)                       T10  65
C     RATE OF CHANGE OF MASS FRACTION OF COMPONENT A                     T10  66
C      = RATE OF INPUT OF A                                              T10  67
C        -RATE OF OUTPUT OF A                                            T10  68
C        +RATE AT WHICH B REACTS INTO A                                  T10  69
C        -RATE AT WHICH A REACTS INTO B                                  T10  70
      DERY(1)=-(K1+1.0/TAU)*Y(1)+K2*Y(2)+S(IG,IN,6)/TAU                  T10  71
C     MASS BALANCE FOR COMPONENT B (UNITS=MIN**-1)                       T10  72
C     RATE OF CHANGE OF MASS FRACTION OF COMPONENT B                     T10  73
C      = RATE OF INPUT OF B                                              T10  74
C        -RATE OF OUTPUT OF B                                            T10  75
C        +RATE AT WHICH A REACTS INTO B                                  T10  76
C        -RATE AT WHICH B REACTS INTO A                                  T10  77
      DERY(2)=K1*Y(1)-(K2+1.0/TAU)*Y(2)+S(IG,IN,7)/TAU                   T10  78
C                                                                        T10  79
C     CALL DIFSUB TO SOLVE ODES FOR MODULE                               T10  80
C                                                                        T10  81
      CALL DIFSUB (2,Y,DERY)                                             T10  82
      IF (IDERY.NE.0) GO TO 1                                            T10  83
C                                                                        T10  84
C     CALCULATE STREAM OUTPUT                                            T10  85
C                                                                        T10  86
C     NORMALIZE CONCENTRATIONS OF A AND B                                T10  87
      SUM=Y(1)+Y(2)                                                      T10  88
      Y(1)=Y(1)/SUM                                                      T10  89
      Y(2)=Y(2)/SUM                                                      T10  90
C     PUT MASS FRACTIONS INTO OUTPUT STREAMS                             T10  91
      S(1,IOUT,6)=Y(1)                                                   T10  92
      S(1,IOUT,7)=Y(2)                                                   T10  93
      RETURN                                                             T10  94
      END                                                                T10  95
```

```
**********************************************************************
        REACTOR SIMULATION - 2 CSTRS
**********************************************************************
BEGIN
TIME        1.0
HMAX        1.0
COMPS       2.0
LIBRARY     1.0
REAC1       10.0
PROCESS
REAC1       1.0
            1.0         -2.0
            10.0        4.90909     4.09091     1.0
REAC1       2.0
            2.0         -3.0
            10.0        686.81      312.19      1.0
END
STREAMS     3.0
EXPLICIT
            1.0         1.0         624.0       60.0        14.7
            1.0         0.0
            2.0         1.0         624.0       70.0        14.7
            1.0         0.0
            3.0         1.0         624.0       120.0       14.7
            1.0         0.0
END
PROPERTIES  -1.0
END
END
```



REFERENCES

Allen, R.H. 1969: Numerically Stable Explicit Integration Techniques Using a Linearized Runge-Kutta Extension (Abstract), Digest: Joint Conference On Mathematical and Computer Aids To Design, P. 333

Allen, R.H. and Fath, A.F. 1969: Fast Computer Analysis Of Nonlinear Systems, Proc. Seventh Annual Allerton Conf. On Circuit and System Theory, P. 206

Allen, R.H. and Pottle, C. 1968: Stable Integration Methods For Electronic Circuit Analysis With Widely Separated Time Constants, Proc. Sixth Annual Allerton Conf. On Circuit and System Theory, P. 311

Amundson, N.R. 1965: Some Further Observations On Tubular Reactor Stability, Can. J. Chem. Eng. 43, 49

Amundson, N.R. and Luss, D. 1968: Qualitative and Quantitative Observations On The Tubular Reactor, Can. J. Chem. Eng. 46, 424

Andrus, J.F. 1967: Integration Of Control Equations and The Problem Of Small Time Constants, NASA TN D-3907

Axelsson, O. 1969: A Class Of A-Stable Methods, BIT 9, 185

Axelsson, O. 1972: A Note On a Class Of Strongly A-Stable Methods, BIT 12, 1

Bending, M.J. and Hutchison, H.P. 1973: The Calculation Of Steady State Incompressible Flow In Large Networks Of Pipes, Chem. Eng. Sci. 28, 1857

Bickart, T.A., Burgess, D.A. and Sloate, H.M. 1971: High Order A-Stable Composite Multistep Methods For Numerical Integration Of Stiff Differential Equations, Proc. Ninth Annual Allerton Conf. On Circuit and System Theory, P. 465

Bickart, T.A. and Picel, Z. 1973: High Order Stiffly Stable Composite Multistep Methods For Numerical Integration Of Stiff Differential Equations, BIT 13, 3, 272

Bjurel, G. 1969: Preliminary Report On: Modified Linear Multistep Methods For A Class Of Stiff Ordinary Differential Equations, Rept. NA69.02, Dept. of Information Processing Computer Science, The Royal Institute Of Technology, Stockholm, Sweden

Bjurel, G., Dahlquist, G., Lindberg, B., Linde, S. and Oden, L. 1970: Survey Of Stiff Ordinary Differential Equations, Rept. NA70.11, Dept. Of Information Processing Computer Science, The Royal Institute Of Technology, Stockholm, Sweden

Bjurel, G. 1972: Modified Linear Multistep Methods For A Class Of Stiff Ordinary Differential Equations, BIT 12, 142

Blue, J.L. and Gummel, H.K. 1970: Rational Approximations To Matrix Exponential For System$ Of Stiff Differential Equations, J. Comp. Phys. 5, 70

Bobrow, S., Johnson, A.I. and Ponton, J.W. 1970: DYNSYS Manual and Application Studies, Dept. of Chemical Eng., McMaster Univ.

Bobrow, S., Ponton, J.W. and Johnson, A.I. 1971: Simulation Of The Transient Behaviour Of Complex Chemical Plants Using A Modular Approach, Can. J. Chem. Eng. 49, 3, 391

Brambilla, A., Caracciolo Di Forino, A., Celati, R., Kardasz, J.H. and Nardini, G. 1971: Study Of Dynamic Behaviour Of Chemical Plants By Digital Simulation, Chem. Eng. Sci. 26, 1101

Brandon, D.M. 1972: IMP - A Software System For The Direct Or Iterative Solution Of Large Differential And/Or Algebraic Systems, General Manual, Dept. Of Chem. Eng., Univ. Of Connecticut, Storrs, Conn.

Brandon, D.M. 1973: Digital Simulation Of Continuous Systems And The New IMP Program, Simuletter/IV/3 (A.C.M.)

Brandon, D.M. 1974a: The Implementation And Use Of Sparse Matrix Techniques In General Simulation Programs, Comp. J. 17, 2, 165

Brandon, D.M. 1974b: Consultant, Research and Advanced Technology, Control Data Corporation, Minneapolis, private communication, Jan. 14

Brandon, D.M. 1974c: private communication, March 22

Brandon, D.M. 1974d: private communication, May 13

Brandon, D.M. 1974e: private communication, June 24

Brandon, D.M. 1974f: A New Single-Step Implicit Integration Algorithm With A-Stability And Improved Accuracy, Simulation 23, 1, 17

Branin, F.H., Hogsett, G.R., Lunde, R.L. and Kugel, L.E. 1971: ECAP II - An Electronic Circuit Analysis Program, I.E.E.E. Spectrum 8, 6

Brayton, R.K., Gustavson, F.G. and Liniger, W. 1966: A Numerical Analysis Of The Transient Behaviour Of A Transistor Circuit, I.B.M. J. Res. Dev. 10, 4, 292

Brayton, R.K., Gustavson, F.G. and Hachtel, G.D. 1972: A New Efficient Algorithm For Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas, Proc. I.E.E.E. 60, 1, 98

Brunner, H. 1972: A Class Of A-Stable Two-Step Methods Based On Schur Polynomials, BIT 12, 468

Brunner, H. 1974: Recursive Collocation For The Numerical Solution Of Stiff Ordinary Differential Equations, Math. Comp. 28, 126, 475

Buffam, R.A. and Kropholler, H.W. 1969: Some Developments And Applications Of The Network Combing Technique In Process Dynamics, Chem. Eng. Sci. 24, 1269

Bulirsch, R. and Stoer, J. 1966: Numerical Treatment Of Ordinary Differential Equations By Extrapolation Methods, Numer. Math. 8, 1

Butcher, J.C. 1964: Implicit Runge-Kutta Processes, Math. Comp. 18, 50

Caillaud, J.B. and Padmanabhan, L. 1971: An Improved Semi-Implicit Runge-Kutta Method For Stiff Systems, Chem. Eng. J. 2, 227

Calahan, D.A. 1967: Numerical Solution Of Linear Systems With Widely Separated Time Constants, Proc. I.E.E.E. (Letters) 55, 11, 2016

Calahan, D.A. 1968a: Efficient Numerical Analysis Of Nonlinear Circuits, Proc. Sixth Annual Allerton Conf. On Circuit and System Theory, P. 321

Calahan, D.A. 1968b: A Stable, Accurate Method Of Numerical Integration For Nonlinear Systems, Proc.I.E.E.E. (Letters) 56, 744

Calahan, D.A. 1969: Numerical Considerations In The Transient Analysis And Optimal Design Of Nonlinear Circuits, Joint Conf. On Math. And Computer Aids To Design, P. 129

Calahan, D.A. 1971: Numerical Considerations For Implementation Of A Nonlinear Transient Circuit Analysis Program, I.E.E.E. Trans. Circuit Theory CT-18, 1, 66

Carnahan, B., Luther, H.A. and Wilkes, J.O. 1969: Applied Numerical Methods, Wiley

Carver, M.B. 1972: FORSIM: A Fortran Oriented Simulation Program With Particular Application To The Solution Of Stiff Equation Systems, Proc. Summer Computer Simulation Conf., San Diego, June 13-16, P. 73

Carver, M.B. 1973: FORSIM: A Fortran Oriented Simulation Package For The Transient Solution Of Simultaneous Ordinary Differential Equations, Rept. AECL-4311, Atomic Energy Of Canada Ltd., Chalk River, Ont.

Cash, J.R. 1972: The Numerical Solution Of Linear Stiff Differential Equations, Proc. Camb. Phil. Soc. 71, 505

Certaine, J. 1960: The Solution Of Ordinary Differential Equations With Large Time Constants, Chapter 11 Of Ralston, B. and Wilf, H.S. 1960: Mathematical Methods For Digital Computers, Vol. 2, Wiley

Chipman, F.H. 1971: A-Stable Runge-Kutta Processes, BIT 11, 384

Chu, C.C. and Berman, M. 1974: An Exponential Method For The Solution Of Systems Of Ordinary Differential Equations, Comm. A.C.M. 17, 12, 699

Control Data 1971: CYBER 70 Computer System Models 72, 73, 74, 76, 7600 Computer System, 6000 Computer Systems, FORTRAN Extended Version 4, Reference Manual, Control Data Corporation, Sunnyvale, California

Cooke, C.H. 1972: On Stiffly Stable Implicit Linear Multistep Methods, SIAM J. Num. Anal. 9, 1, 29

Cooke, C.H. 1973: A Characterization of Stiffly Stable Linear Multistep Methods, Int. J. Num. Meth. Eng. 7, 117

Cooper, G.J. 1969: The Numerical Solution Of Stiff Differential Equations, Computing Techniques In Biochemistry, FEBS Letters, S22, March

Crean, J.P. 1974: Manager, Sales Administration, Stone and Webster Engineering Corporation, New York, private communication, Sept. 4

Creighton, J. 1971: The Role Of Numerical Stability In The Computer Solution Of Chemical Chain Reactions, Proc. Summer Computer Simulation Conf., Boston, July 19-21, P. 440

Cryer, C.W. 1972: On the Instability Of High Order Backward-Difference Multistep Methods, BIT 12, 17

Cryer, C.W. 1973: A New Class Of Highly-Stable Methods: A_0-Stable Methods, BIT 13, 153

CSMP 1967: System/360 Continuous System Modeling Program Users Manual, Form H20-0367-3, Program Number 360A-CX-16X, I.B.M., White Plains, New York

Culver, D.A. 1972: Distillation Column Dynamics Simulation, Proc. Summer Computer Simulation Conf., San Diego, June 13-16, P. 404

Curtis, A.R. and Reid, J.K. 1971: The Solution Of Large Sparse Unsymmetric Systems Of Linear Equations, J. Inst. Maths. Applics. 8, 344

Curtiss, C.F. and Hirschfelder, J.O. 1952: Integration Of Stiff Equations, Proc. Nat. Acad. Sci. 38, 235

Dahlquist, G. 1963: A Special Stability Problem For Linear Multistep Methods, BIT 3,27

Dahlquist, G. 1969: A Numerical Method For Some Ordinary Differential Equations With Large Lipschitz Constants, Information Processing 68 (Proc. IFIP Congress 1968), North-Holland Publishing Co., Amsterdam, P. 103

Davison, E.J. 1968: The Numerical Solution Of Large Systems Of Linear Differential Equations, A.I.Ch.E. J. 14, 1, 46

Davison, E.J. 1973: An Algorithm For The Computer Simulation Of Very Large Dynamic Systems, Automatica 9, 665

DeGroat, J.J. and Abbett, M.J. 1965: A Computation Of One-Dimensional Combustion Of Methane, A.I.A.A. J. 3, 2, 381

Dill, C. and Gear, C.W. 1971: A Graphical Search For Stiffly Stable Methods For Ordinary Differential Equations, J.A.C.M. 18, 1, 75

Diperna, R.A. 1971: A Digital Simulation Method For Interconnected Continuous Systems, Proc. Ninth Annual Allerton Conf. On Circuit And System Theory, P. 474

Distefano, G.P. 1968a: Mathematical Modeling And Numerical Integration Of Multicomponent Batch Distillation Equations, A.I.Ch.E. J. 14, 1, 190

Distefano, G.P. 1968b: Stability Of Numerical Integration Techniques, A.I.Ch.E. J. 14, 6, 946

Distefano, G.P. 1968c: Causes Of Instabilities In Numerical Integration Techniques, Int. J. Comp. Math. 2, 123

Ehle, B.L. 1968: High Order A-Stable Methods For The Numerical Solution Of Systems Of D.E.S., BIT 8, 276

Emanuel, G. 1963: Problems Underlying The Numerical Integration Of The Chemical And Vibrational Rate Equations In A Near-Equilibrium Flow, AEDC-TDR-63-82, Aerospace Corp., El Segundo, Calif.

Emanuel, G. 1964: Numerical Analysis Of Stiff Equations, Rept. TDR-269(4230-20)-3, SSD-TDR-63-380, Aerospace Corporation, El Segundo, Calif.

Emanuel, G. 1967: Comments On The Paper, "A Chemical Kinetics Computer Program For Homogeneous and Free-Radical Systems Of Reactions" By R.H. Snow, J. Phys. Chem. 71, 4, 1161

Enright, W.H. 1974: Assistant Professor, Dept. Of Computer Science, Univ. Of Toronto, private communication, April 4

Eschenroeder, A.Q., Boyer, D.W. and Hall, J.G. 1962: Non-equilibrium Expansions Of Air With Coupled Chemical Reactions, Physics Of Fluids 5, 5, 615

Fairchild, B.T., Wengrow, H.R. and May, F.P. 1965: AMOS, Numerical Integration Of Differential Equations With the Adams-Moulton-Shell Method, Dept. Of Chem. Eng., Univ. Of Florida, Gainsville

Forsythe, G. and Moler, C.B. 1967: Computer Solution Of Linear Algebraic Systems, Prentice-Hall

Fowler, M.E. and Warten, R.M. 1967: A Numerical Integration Technique For Ordinary Differential Equations With Widely Separated Eigenvalues, I.B.M. J. Res. Dev. 11, 537

Franks, R.G.E. 1967: Mathematical Modeling In Chemical Engineering, Wiley

Franks, R.G.E. 1971: Dynamic Systems Programs For Chemical Engineering, Paper Presented To a Seminar Held At The 64th Annual A.I.Ch.E. Meeting, San Francisco, Nov.

Franks, R.G.E. 1972a: Modeling And Simulation In Chemical Engineering, Wiley-Interscience

Franks, R.G.E. 1972b: Notes From Workshop On Dynamic Systems And Control, Purdue Univ., Oct. 26-28

Gear, C.W. 1967: The Numerical Integration Of Ordinary Differential Equations, Math. Comp 21, 146

Gear, C.W. 1969a: The Automatic Integration Of Stiff Ordinary Differential Equations, Information Processing 68 (Proc. IFIP Congress 1968), North-Holland Publishing Co., Amsterdam, P. 187

Gear, C.W. 1969b: The Automatic Integration Of Large Systems Of Ordinary Differential Equations, Digest: Joint Conf. On Mathematical And Computer Aids To Design, P. 27

Gear, C.W. 1970: Rational Approximations By Implicit Runge-Kutta Schemes, BIT 10, 20

Gear, C.W. 1971a: The Automatic Integration Of Ordinary Differential Equations, Comm. A.C.M. 14, 3, 176

Gear, C.W. 1971b: Algorithm 407, DIFSUB For Solution Of Ordinary Differential Equations, Comm. A.C.M. 14, 3, 185

Gear, C.W. 1971c: Numerical Initial Value Problems In Ordinary Differential Equations, Prentice-Hall.

Gear, C.W. 1971d: Stimultaneous Numerical Solution Of Differential-Algebraic Equations, I.E.E.E. Trans. On Circuit Theory CT-18, 1, 89

Gear, C.W. 1972: Stiff Ordinary Differential Equations And Techniques Suited To Continuous System Simulation, Proc. Summer Computer Simulation Conf., San Diego, June 13-16, P. 223

Gear, C.W. 1974: Professor, Dept. Of Computer Science, Univ. Of Illinois At Urbana-Champaign, Urbana, Illinois, private communication, May 14

Gelinas, R.J. 1972: Stiff Systems Of Kinetic Equations - A Practitioners View, J. Comp. Phys. 9, 222

Giese, C. 1967: State Variable Difference Methods For Digital Simulation, Simulation, 8, 5, 263

Giloi, W. and Grebe, 1968: Construction Of Multistep Integration Formulas For Simulation Purposes, I.E.E.E. Trans. Computers C-17, 12, 1121

Guderley, K.G. and Hsu, C. 1972: A Predictor-Corrector Method For A Certain Class Of Stiff Differential Equations, Math. Comp. 26, 117, 51

Gustavson, F.G., Liniger, W. and Willoughby, R. 1970: Symbolic Generation Of An Optimal Crout Algorithm For Sparse Systems Of Linear Equations, J.A.C.M. 17, 1, 87

Hachtel, G.D., Brayton, R.K. and Gustavson, F.G. 1971: The Sparse Tableau Approach To Network Analysis And Design, I.E.E.E. Trans, Circuit Theory CT-18, 1, 101

Haines, C.F. 1969: Implicit Integration Processes With Error Estimate For The Numerical Solution Of Differential Equations, Comp. J. 12, 183

Ham, P.G. 1969: Users Manual, REMUS, Routine For Executive Multi-Unit Simulation, The School Of Chemical Engineering And The Moore School Of Electrical Engineering, Univ. Of Pennsylvania

Henrici, P. 1962: Discrete Variable Methods In Ordinary Differential Equations, Wiley

Hildebrand, F.B. 1956: Introduction To Numerical Analysis, McGraw-Hill

Himmelblau, D.M. (ed.) 1973: Decomposition Of Large-Scale Problems, North-Holland Publishing Co., Amsterdam-London

Hodgkins, W.R. 1969: A Method For The Numerical Integration Of Nonlinear Ordinary Differential Equations With Greatly Different Time Constants, Conference On The Numerical Solution Of Differential Equations, Lecture Notes In Mathematics #109, Springer-Verlag, Berlin, P. 172

Hronsky, P. and Martens, H.R. 1973: Computer Techniques For Stiff Differential Equations, Proc. Summer Simulation Conf., Montreal, July 17-19, P. 131

Hull, T.E. 1969: The Numerical Integration Of Ordinary Differential Equations, Information Processing 68 (Proc. IFIP Congress 1968), North-Holland Publishing Co., Amsterdam, P. 40

Hull, T.E., Enright, W.H., Fellen, B.M. and Sedgwick, A.E. 1972a: Comparing Numerical Methods For Ordinary Differential Equations, SIAM J. Num. Anal. 9, 4, 603

Hull, T.E., Enright, W.H. and Sedgwick, A.E. 1972b: Memorandum On Testing Of Stiff Techniques, Dept. Of Computer Science, University Of Toronto, May 7

IMSL 1973: IMSL Library 3, Reference Manual, Edition 3, International Mathematical And Statistical Libraries, Inc., Houston

Ingels, D.M. and Motard, R.L. 1970: PRODYC - A Simulation Program For Chemical Process Dynamics And Control, RE 4-70, Dept. Of Chem. Eng., Univ. Of Houston

Jain, R.K. 1971: A-Stable Multi-Step Methods For Stiff Ordinary Differential Equations, Proc. Manitoba Conf. On Numerical Mathematics, Univ. Of Manitoba, Winnipeg, Oct. 7-9, P. 401

Jain, R.K. 1972: Some A-Stable Methods For Stiff Ordinary Differential Equations, Math. Comp. 26, 117, 71

Jung, H.P. 1967: Numerical Integration Of "Stiff" Equations, Rept. TIS-67SD245, General Electric Co., Syracuse, N.Y.

Jung, H.P. 1968: Pade Integration Methods, Space Division Memo PIR 5540-82, General Electric Co., Syracuse, N.Y.

Kardasz, J.H. 1969: On A Language For Dynamic Simulation Of Chemical Plants, Ph.D. Thesis, Scuola Normale Superiore, Pisa

Kardasz, J.H. and Molnar, G. 1971: A High Level Structure Oriented Language For Chemical Plants, Proc. IFAC Symposium On Digital Simulation Of Continuous Processes (DISCOP), Gyor, Hungary, Sept., K3, P. 1

Kardasz, J.H. and Molnar, G. 1974: A SIMULA-Based Structure Oriented Language For The Dynamic Simulation Of Chemical Plants, Computer J. 17, 1, 28

Key, J.E. 1973: Computer Program For Solution Of Large, Sparse, Unsymmetric Systems Of Linear Equations, Int. J. Num. Meth. Eng. 6,497

Klopfenstein, R.W. and Davis, C.B. 1971: PECE Algorithms For The Solution Of Stiff Systems Of Ordinary Differential Equations, Math. Comp. 25, 115, 457

Klopfenstein, R.W. 1971: Numerical Differentiation Formulas For Stiff Systems Of Ordinary Differential Equations, RCA Review 32, 447

Koenig, D.M. 1972: OSUSIM: A Modular Approach To Dynamic Simulation, Ph.D. Thesis, Dept. Of Chem. Eng., Ohio State Univ.

Kollrack, R. 1967: Computer Program For Chemical Kinetics, J. Phys. Chem. 71, 4, 1162

Kreyszig, E. 1967: Advanced Engineering Mathematics, Wiley

Krogh, F.T. 1969: A Variable Step Variable Order Multistep Method For The Numerical Solution Of Ordinary Differential Equations, Information Processing 68, (Proc. IFIP Congress 1968), North-Holland Publishing Co., Amsterdam, P. 194

Krogh, F.T. 1971: An Integrator Design, Technical Memo 33-479, Jet Propulsion Lab, California Institute Of Technology, Pasadena

Krogh, F.T. 1973: On Testing A Subroutine For The Numerical Integration Of Ordinary Differential Equations, J. A.C.M. 20, 4, 545

Kubicek, M. 1975: A Nonlinear Explicit Second Order Algorithm For Integration Of Stiff Equations, Comm. A.C.M. (To Be Published)

Lambert, J.D. 1973: Computational Methods In Ordinary Differential Equations, Wiley

Lapidus, L. and Seinfeld, J.H. 1971: Numerical Solution Of Ordinary Differential Equations, Academic Press

Lawson, J.D. 1966: An Order Five Runge-Kutta Process With Extended Region Of Stability, SIAM J. Num. Anal. 3, 4, 593

Lawson, J.D. 1967a: Generalized Runge-Kutta Processes For Stable Systems With Large Lipschitz Constants, SIAM J. Num. Anal. 4, 3, 372

Lawson, J.D. 1967b: An Order Six Runge-Kutta Process With Extended Region Of Stability, SIAM J. Num.Anal. 4, 4, 620

Lee, H.B. 1967: Matrix Filtering As An Aid To Numerical Integration, Proc. I.E.E.E. 55, 11, 1826

Lindberg, B. 1971a: On Smoothing And Extrapolation For The Trapezoidal Rule, BIT 11, 29

Lindberg, B. 1971b: On Smoothing For The Trapezoidal Rule, Rept. NA71.31, Dept. Of Information Processing Computer Science, The Royal Institute Of Technology, Stockholm, Sweden

Lindberg, B. 1972: IMPEX, A Program Package For Solution Of Systems Of Stiff Differential Equations, Rept. NA72.50, Dept. Of Information Processing Computer Science, The Royal Institute Of Technology, Stockholm, Sweden

Liniger, W. 1968: A Criterion For A-Stability Of Linear Multistep Integration Formulae, Computing 3, 280

Liniger, W. 1969: Global Accuracy And A-Stability Of One- And Two-Step Integration Formulae For Stiff Ordinary Differential Equations, Conference On The Numerical Solution Of Differential Equations, Lecture Notes In Mathematics #109, Springer-Verlag, Berlin, P. 188

Liniger, W. 1971: A Stopping Criterion For The Newton-Raphson Method In Implicit Multistep Integration Algorithms For Nonlinear Systems Of Ordinary Differential Equations, Comm. A.C.M. 14, 9, 600

Liniger, W. and Willoughby, R.A. 1970: Efficient Integration Methods For Stiff Systems Of Ordinary Differential Equations, SIAM J. Num. Anal. 7, 1, 47

Liniger, W. and Odeh, F. 1972: A-Stable, Accurate Averaging Of Multistep Methods For Stiff Differential Equations, I.B.M. J. Res. Dev. 16, 4, 335

Loibl, J.M., Camp, D.T. and Wilkins, G.S. 1973: The Dynamic Analysis Of Chemical Processes With a User-Oriented Executive Program, Proc. Summer Computer Simulation Conf., Montreal, July 17-19, P. 367

Liou, M.L. 1966: A Novel Method Of Evaluating Transient Response, Proc. I.E.E.E. 54, 1, 20

Lomax, H. and Bailey, H.E. 1967: A Critical Analysis Of Various Numerical Integration Methods For Computing The Flow Of A Gas In Chemical Nonequilibrium, NASA TN D-4109

Lomax, H. 1968: On The Construction Of Highly Stable Explicit Numerical Methods For Integrating Coupled Ordinary Differential Equations With Parasitic Eigenvalues, NASA TN D-4547

Lopes, F.C. and Phares, W.J. 1965: Numerical Integration Of First-Order Stiff Differential Equations, Arnold Engineering Development Centre, Arnold Air Force Station, Tennessee, AEDC-TR-65-262

MacMillan, D.B. 1968: Asymptotic Methods For Systems Of Differential Equations In Which Some Variables Have Very Short Response Times, SIAM J. Appl. Math. 16, 4, 704

Mah, R.S.H., Michaelson, S. and Sargent, R.W.H. 1962: Dynamic Behaviour Of Multi-Component Multi-Stage Systems, Numerical Methods For The Solution, Chem. Eng. Sci. 17, 619

Maheshwari, R.P. 1968: Solution Of Stiff Differential Equations Arising In Large Order Chemical Engineering Systems, M.A.Sc. Thesis, Dept. of Chem. Eng., Univ. of Waterloo

Merson, R.H. 1957: An Operational Method For The Study Of Integration Processes, Proc. Of Conference On Data Processing And Automatic Computing Machines, Weapons Research Establishment, Salisbury, Australia, June 3-8, P. 110-1

Millares, R. 1975: M. Eng. Sci. Thesis, Univ. Of Western Ontario

Miller, J.C.P. 1964: Numerical Analysis - An Introduction, Walsh, J. (ed.), Academic Press

Miranker, W.L. 1971: Matricial Difference Schemes For Integrating Stiff Systems Of Ordinary Differential Equations, Math. Comp. 25, 116, 717

Miranker, W.L. 1973: Numerical Methods Of Boundary Layer Type For Stiff Systems Of Differential Equations, Computing 11, 3, 221

Moretti, G. 1965: A New Technique For The Numerical Analysis Of Nonequilibrium Flows, A.I.A.A. J. 3, 2, 223

Nichol, K.C. 1970: Digital Simulation Of Continuous Systems Using State-Space Techniques, Rept. TIS-70ELS31, General Electric Co., Syracuse, N.Y.

Nigro, B.J. 1969: An Investigation Of Optimally Stable Numerical Integration Methods With Application To Real Time Simulation, Simulation 13, 253

Nikolai, P.J. 1973: Certification Of Algorithm 407, DIFSUB For Solution Of Ordinary Differential Equations, Comm. A.C.M. 16, 7, 448

Nordsieck, A. 1962: On Numerical Integration Of Ordinary Differential Equations, Math. Comp. 16, 1, 22

Norsett, S.P. 1969: An A-Stable Modification Of The Adams-Bashforth Methods, Conference On The Numerical Solution Of Differential Equations, Lecture Notes In Mathematics #109, P. 214, Springer-Verlag, Berlin

Northcott, T.H. 1967: MIMIC - A Study Of The Technique And Application To Modeling Of An Extraction Column, M.Eng. Thesis, McMaster Univ.

Odeh, F. and Liniger, W. 1972: A Note On Unconditional Fixed-h Stability Of Linear Multistep Formulae, Information Processing 71 (Proc. IFIP Congress 1971), North-Holland Publishing Co., Amsterdam, P. 1284

Oden, L. 1971: An Experimental and Theoretical Analysis Of The SAPS-Methods For Stiff Ordinary Differential Equations, Report NA71.28, Dept. Of Information Processing Computer Science, The Royal Institute Of Technology, Stockholm, Sweden

Osborne, M.R. 1969: A New Method For The Integration Of Stiff Systems Of O.D.E.s, Information Processing 68 (Proc. IFIP Congress 1968), North-Holland Publishing Co., Amsterdam, P. 200

Oswald, R. and Smith, O.J.M. 1971: Integration Of Nonlinear Differential Systems With Wide Eigenvalue Range, I.E.E.E. Trans. Power App. Sys. PAS-6, 2586

Pope, D.A. 1963: An Exponential Method Of Numerical Integration Of Ordinary Differential Equations, Comm. A.C.M. 6, 8, 491

Price, W. W. 1970: FACE - A Digital Dynamic Simulation Program, Rept. DF-70-EU-2072, General Electric Co., Syracuse, N.Y.

Prothero, A. and Robinson, A. 1974: On The Stability And Accuracy Of One-Step Methods For Solving Stiff Systems Of Ordinary Differential Equations, Math. Comp. 28, 125, 145

Pulido, J. 1975: M. Eng. Sci. Thesis, Univ. Of Western Ontario

Ratliff, K. 1968: A Comparison Of Techniques For The Numerical Integration Of Ordinary Differential Equations, Rept. #274, Dept. Of Comp. Sci., Univ. Of Illinois At Urbana-Champaign

Reid, J.K. 1970: A FORTRAN Subroutine For The Solution Of Large Sets Of Linear Equations By Conjugate Gradients, A.E.R.E. Rept. R6545, H.M.S.O.

Reid, J.K. (ed.) 1971: Large Sparse Sets Of Linear Equations, Academic Press

Rheinboldt, W.C. 1973: Programs For The Solution Of Large Sparse Matrix Problems Based On The Arc-Graph Structure, PB-224 810, NTIS

Richards, P.I., Lanning, W.D. and Torrey, M.D. 1965: Numerical Integration Of Large Highly-Damped, Nonlinear Systems, SIAM Rev. 7, 3, 376

Robertson, H.H. 1966: Some New Formulae For The Numerical Integration Of Differential Equations, I.C.I. Management Services, Math And Comp Dept., Wilton Works, Middlesborough, MCD/66/47

Robertson, H.H. 1967: The Solution Of A Set Of Reaction Rate Equations, In Walsh, J. (ed.), Numerical Analysis, Thompson Book Co., Washington

Roe, G.M. 1967: Experiments With A New Integration Algorithm, Rept. TIS-67-C-037, General Electric Co., Syracuse, N.Y.

Rose, D.J. and Willoughby, R.A. (ed.) 1972: Sparse Matrices and Their Application, Plenum Press, New York

Rosenbrock, H.H. 1957: An Investigation Of The Transient Response Of A Distillation Column. Part 1: Solutions Of The Equations, Trans. Inst. Chem. Engrs. 35, 347

Rosenbrock, H.H. 1963: Some General Implicit Processes For The Numerical Solution Of Differential Equations, Comp. J. 5, 329

Sandberg, I.W. and Shichman, H. 1968: Numerical Integration Of Systems Of Stiff Nonlinear Differential Equations, Bell Systems Technical J. 47, 511

Sarkany, E.F. and Ball, W.E. 1969: A Predictor-Corrector Modification Of The Cohen-Flatt-Certaine Method For Solving Differential Equations (Abstract), Digest: Joint Conf. On Math. And Computer Aids To Design, P. 336

Schappelle, R. 1967: LINEQ4, A FORTRAN Subroutine For The Solution Of Sparse Linear Equations Available Through CDC User Association VIM, VIM F4 GDC CSLNQ4

Schoen, K. 1971: Fifth And Sixth Order PECE Algorithms With Improved Stability Properties, SIAM J. Num. Anal. 8, 2, 244

Seinfeld, J.H., Lapidus, L. and Hwang, M. 1970: Review Of Numerical Techniques For Stiff Ordinary Differential Equations, I.E.C. Fund. 9, 2, 266

Shannon, P.H. 1971: Numerical Integration Of Stiff, Sensitive And Multivalued Equations, M.A.Sc. Thesis, Dept. Of Chem. Eng., Univ. Of Houston

Shell, D.L. 1958: General Electric Co., Tech. Information Ser. No. DF58AGT679, General Electric Co., Cincinnati

Shern, D.R. and Petty, A.W. 1970: Engineering Development Division Memorandum Report "FLEX" - Digital Simulation On The 1130 Computer, Procter And Gamble, Cincinnati, May 4

Shichman, H. 1969: The Design Of The Integration System For A Nonlinear Circuit Analysis Program (Abstract), Digest: Joint Conf. On Math. And Computer Aids To Design, P. 394

Sigurdsson, S.T. 1970: Study Of Numerical Methods For Integration Of Stiff Systems Of Ordinary Differential Equations, M.Sc. Thesis, Dept. Of Math., Univ. Of Dundee

Sloate, H. 1970: An Implicit Formula For The Integration Of Stiff Network Equations, Rept. TIS-70ELS2, General Electric Co., Syracuse, N.Y.

Sloate, H.M. 1971: Simultaneous Implicit Formulas For The Solution Of Stiff Systems Of Differential Equations, Ph.D. Thesis, Dept. Of Elect. Eng., Syracuse, Univ.

Sloate, H.M. and Bickart, T.A. 1973: A-Stable Composite Multistep Methods, J. A.C.M. 20, 7

Snider, A.D. and Fleming, G.C. 1974: Approximation By Aliasing With Application To "Certaine" Stiff Differential Equations, Math. Comp. 28, 126, 465

Snow, R.H. 1966: A Chemical Kinetics Computer Program For Homogeneous And Free-Radical Systems Of Reactions, J. Phys. Chem. 70, 2780

Snow, R.H. 1967: Reply To Comments On The Paper, "A Chemical Kinetics Program For Homogeneous And Free-Radical Systems Of Reactions", J. Phys. Chem. 71, 4, 1162

Spicer, J.A. 1968: A Predictor-Corrector Algorithm Using Exact Stability, Conf. Record Of Second Asilomar Conf. On Circuits & Systems, Pacific Grove, Calif., P. 430

Spicer, J.A. 1969: Numerical Integration Of Nonlinear Flight Equations Using The Modified Exact Stability Algorithm (Abstract), Digest: Joint Conf. On Math. And Computer Aids To Design, P. 337

SSP 1970: System/360 Scientific Subroutine Package Version III, Programmers Manual, 5th Ed., GH20-0205-4, I.B.M., White Plains, New York

Stanton, K.N. and Talukdar, S. N. 1970: New Integration Algorithms For Transient Stability Studies, I.E.E.E. Trans. Power App. Sys. PAS-89, 5, 985

Steeper, D.E. 1970: Numerical Integration Of Stiff Differential Equations, Rept. DF-70-IE-111, General Electric Co., Syracuse, N.Y.

Stineman, R.W. 1965: Digital Time-Domain Analysis Of Systems With Widely Separated Poles, J. A.C.M. 12, 2, 286

Treanor, C.E. 1966: A Method For The Numerical Integration Of Coupled First-Order Differential Equations With Greatly Different Time Constants, Math. Comp. 20, 39

Utsumi, T. 1969: Stone And Webster All Purpose Simulator And Optimizer (SWAPSO) And Its Applications, Paper Presented At The Conference On Applications Of Continuous Systems Simulations Languages, San Francisco

Varga, R.S. 1962: Matrix Iterative Analysis, Prentice-Hall

Walters, C.M. 1971: Numerical Integration Of Stiff Ordinary Differential Equations, AFWL-TN-71-6,NTIS

Walters, C.M. 1972: A Discussion Of Gears Implementation Of Adams And Stiff Methods For Solving Ordinary Differential Equations, AFWL-TR-72-170, NTIS

Watts, H.A. and Shampine, L.F. 1972: A-Stable Block Implicit One-Step Methods, BIT 12, 252

Weaver, R.E. 1974: Professor, Dept. Of Chem. Eng., Tulane Univ., New Orleans, private communication, Sept. 24

Westlake, J.R. 1968: A Handbook Of Numerical Matrix Inversion And Solution Of Linear Equations, Wiley

Widlund, O.B. 1967: A Note On Unconditionally Stable Linear Multistep Methods, BIT 7, 65

Williams, J. and Hoog, F. 1974: A Class Of A-Stable Advanced Multistep Methods, Math. Comp. 28, 125, 163

Williams, T.J. and Otto, R.E. 1960: A Generalized Chemical Processing Model For The Investigation Of Computer Control, A.I.E.E. Trans., Part 1, 79; 458

Williams, T.J. 1961: Systems Engineering For The Process Industries, McGraw-Hill

Willoughby, R.A. (ed.) 1969: Sparse Matrix Proceedings, I.B.M., Thomas J. Watson Research Centre, Yorktown Heights, New York

Young, D.M. 1971: Iterative Solution Of Large Linear Systems, Academic Press

Zavorin, A.I. and Khesina, I.Y. 1974: Some Numerical Methods For The Solution Of Stiff Systems Of Ordinary Differential Equations, U.S.S.R. Computational Mathematics And Mathematical Physics 13, 1, 89

Zein, D.A.R. and Hakimi, S.L. 1971: A Computer Approach For Analysis Of Nonlinear Circuits, Proc. Ninth Annual Allerton Conf. On Circuit and System Theory, P. 455