GEORGIA INSTITUTE OF TECHNOLOGY
Engineering Experiment Station
Atlanta, Georgia

FINAL REPORT

PROJECT A-740

STUDY OF METHODS FOR THE NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS

By

H. L. DURHAM, JR., O. B. FRANCIS, JR., L. J. GALLAHER,
H. G. HALE, JR., AND I. E. PERLIN

CONTRACT NAS8-11129

9 NOVEMBER 1963 to 8 NOVEMBER 1964

Performed for
GEORGE C. MARSHALL SPACE FLIGHT CENTER
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
HUNTSVILLE, ALABAMA

TABLE OF CONTENTS

		Page
I.	INTRODUCTION	1
II.	ABSTRACT	3
III.	THE RESTRICTED THREE BODY PROBLEM	5
IV.	INTEGRATION METHODS	15
	A. The Lie Series Method of Groebner	15
	1. Description of the Method	15
	1.1. Evaluation of the Integrals	18
	1.2. Error Estimates	19
	1.3. Step Size Control	20
	2. The Lie Series Computer Program	21
	2.1. Closing an Orbit	21
	2.2. Searching for Improved Initial Conditions	22
	2.3. Data Output	22
	2.4. Data Input	23
	3. The Flow Diagram for the Lie Series Method	27
	4. The Program Listing for the Lie Series Method	27
	B. Cowell's Method	48
	1. General	48
	2. The First Order Case	49
	2.1. The General Algorithm	49
	2.2. Evaluation of the Coefficients	53
	3. The Second Order Case	56
	3.1. The General Algorithm	56
	3.2. Evaluation of the Coefficients	57
	4. The Starting Table of Differences	60
	4.1. The Modified Cowell Method	62

(Continued)

TABLE OF CONTENTS (Continued)

Page

5.1. Predictor-Corrector Formulas for the First Order Case 62
5.2. Predictor-Corrector Formulas for the Second Order Case 65
5.3. Mid-Range Formulas 67

6. A General Description of an Application of Cowell's Method to the Simplest Second Order Initial Value Problem. 71

6.1. The Starting Procedure and Initial Step Size Determination. 87
6.2. Intermediate Step Size Control 90

7. Predictor-Corrector Formulas (Expanded Form) 91

7.1. First Order Case-Predictor 91
7.2. First Order Case-Corrector 97
7.3. Second Order Case-Predictor. 101
7.4. Second Order Case-Corrector. 102

8. The Computer Program 104

8.1. The Initial Value Problem. 104
8.2. Operating Instructions 105
8.3. Error Control. 108

9. Program Listing for Cowell's Method. 110

C. The Adams Method 153

1. Description of the Method. 153

2. The Adams Computer Program 155

2.1. Starting Procedure 155
2.2. Orders Used. 155
2.3. Error Estimates and Step Size Control. 155
2.4. Reducing the Step Size 157
2.5. Closing an Orbit 157
2.6. Searching for Improved Initial Conditions. 158
2.7. Data Output. 158
2.8. Data Input 159

3. Flow Diagram for the Adams Method. 165

4. The Program Listing for the Adams Method 165

D. The Method of Runge-Kutta-Fehlberg 197

(Continued)

TABLE OF CONTENTS (Continued)

Page

1. Introduction 197

2. Theory 197

3. The Runge-Kutta-Fehlberg Computer Program. 203

3.1. Data Input 205

4. Flow Diagram for the Runge-Kutta-Fehlberg Method 210

5. The Program Listing for the Runge-Kutta-Fehlberg Method. . 210

E. The Method of Runge-Kutta-Shanks 235

1. Introduction 235

2. Description of the Method. 236

3. The Computer Program for the Shanks Method 237

3.1. Error Estimates and Step Size Control. 237
3.2. Searching for Improved Initial Conditions. 237
3.3. Closing an Orbit 239
3.4. Data Input 239

4. Flow Diagram for the Runge-Kutta-Shanks Method 241

5. Program Listing for the Runge-Kutta-Shanks Method 244

V. RESULTS AND CONCLUSIONS. 267

A. Results. 267

B. Conclusions. 271

1. The Lie Series Method. 271

2. The Adams Method 272

3. The Cowell Method 273

4. The Runge-Kutta-Fehlberg Method. 274

5. The Runge-Kutta-Shanks Method. 274

(Continued)

TABLE OF CONTENTS (Concluded)

Page

6. Comparisons of the Various Methods 275

7. General Discussion 275

C. Recommendations. 276

D. Summary. 277

VI. REFERENCES AND LITERATURE CITED. 279

LIST OF TABLES

		Page
I.	THE γ COEFFICIENTS (RATIONAL FORM)	56
II.	THE γ^* COEFFICIENTS (RATIONAL FORM)	58
III.	GAMMA COEFFICIENTS (HIGH PRECISION).	59
IV.	PREDICTOR-CORRECTOR AND MID RANGE FORMULA COEFFICIENTS	72
V.	TABLE OF ORBIT RUNS OF LOW ORDER (5-9) AND ACCURACY (10^{-12}). . . .	268
VI.	TABLE OF ORBIT RUNS OF INTERMEDIATE ORDER (12-13) AND ACCURACY (10^{-14}).	269
VII.	TABLE OF ORBIT RUNS OF HIGH ORDER (14-16) AND ACCURACY (10^{-16}) . .	270

GEORGE C. MARSHALL SPACE FLIGHT CENTER
HUNTSVILLE, ALABAMA

Memorandum

THRU Mr. Robert L. Wesson, R-COMP-M

TO Mr. Audie E. Anderson, R-COMP-RD

Mr. McMurrey, 842-2378
DATE March 4,1965

FROM Chief, Contract Administration Section, PR-EC

N65-20106

SUBJECT Review of [redacted]/Final Contractor Report, NAS8-11129 - Georgia Institute of Technology

1. The attached Document Release Form (Facility Test Form 545) is forwarded for action by your office. Please review subject report and complete the Form 545, to indicate your recommendations as to the availability of the report for use by qualified recipients in the Aerospace Field. (See explanation on reverse of green copy).

2. The completed Form 545 will be forwarded through the Space System Information Branch (MS-I) for transmittal to NASA Headquarters, Office of Scientific and Technical Information and subsequent listing of the report in the NASA Scientific and Technical Aerospace Reports (STAR) Announcement Journal.

3. Questions concerning security classification will be referred to the Security Branch, MS-S. Questions concerning proprietary data will be resolved with the Associate Chief Counsel for patent matters CC-P before the form is returned.

4. Your expeditious handling of this request will be appreciated.

5. If distribution should be limited or not made, indicate below your recommendation and forward the form to MS-I.

Joseph D. Hinesley

COMMENT 1

TO MS-I DATE

FROM

The completed Form 545 containing recommendation is forwarded for action by your office.

Contracting Officer's Representative

MSFC - Form 488 (August 1960)

LIST OF FIGURES

		Page
1.	Orbit Number 1	9
2.	Orbit Number 2	11
3.	Orbit Number 3	13
4.	The Flow Diagram for the Lie Series Method	26
5.	Starting Table of Differences for the Cowell Method	61
6.	The Flow Diagram for the Cowell Method	83
7.	The Flow Diagram for the Initial Step Size Determination	107
8.	The Flow Diagram for the Adams Method	162
9.	The Flow Diagram for the Runge-Kutta-Fehlberg Method	209
10.	The Flow Diagram for the Runge-Kutta-Shanks Method	242

I. INTRODUCTION

The object of this study is to test a series of methods for integrating nonlinear coupled differential equations. By applying each method to the same set of equations and by programming all on the same computer (the Burroughs B-5000) a comparison in terms of speed and accuracy of the different methods is obtained.

The set of differential equations chosen for the test are the equations of motion of an earth-moon satellite, using the simplifying assumptions that the earth to moon distance is constant and that the satellite remains in the plane of the lunar orbit. The advantage of these particular equations and conditions is that periodic orbits are known for this system and a check for periodicity will give a guage for the accuracy of a method.

The methods to be used are the following:

(A) the single step Lie Series method of Groebner

(B) the predictor-corrector multistep method of Cowell

(C) the predictor-corrector multistep method of Adams

(D) the single step method of Runge, Kutta, and Fehlberg

(E) the single step method of Runge, Kutta, and Shanks

Each of these methods is described in detail in separate sections.

II. ABSTRACT

This study was an examination of various methods for integrating non-linear coupled differential equations. The methods used were the following:

A. the single step Lie Series method,

B. the multistep Cowell method,

C. the multistep Adams method,

D. the single step Runge-Kutta-Fehlberg method,

E. the single step Runge-Kutta-Shanks method.

Each of the methods is discussed in detail in this report.

The above methods were applied to the restricted three body problem. In particular, Arenstorf orbits of the restricted three body problem were used. The error in the methods was checked by noting the degree by which the initial conditions failed to be reproduced at the end of one complete period.

Programs for each method were written in double precision floating point arithmetic (23 decimal places) in Extended Algol for the B5000. A series of runs were made on three different Arenstorf orbits at various orders from 7 to 16 and accuracies from 10^{-12} to 10^{-16}. These results are presented in Tables V, VI, and VII.

The conclusions reached were that each of the methods, except that of Cowell, could be considered effective, but the methods of Runge-Kutta-Shanks and Runge-Kutta-Fehlberg were the best. At the highest accuracies and orders, where Runge-Kutta-Shanks formulas are not available, the Runge-Kutta-Fehlberg method was superior.

III. THE RESTRICTED THREE-BODY PROBLEM

The various methods are applied to a particular problem known as the restricted three-body problem. Here three masses are assumed to attract each other by an inverse square force but one of the masses is considered small enough so that its influence on the motion of the other two can be neglected. It is also assumed that the motion takes place in a plane and that the distance between the two massive bodies remains fixed. This problem is of special interest since Arenstorf [1] showed the existence of periodic orbits for this system. Many of these orbits are known and can be used as a check on the accuracy of the integration method. In the rotating coordinate system in which the two massive bodies appear to be at rest, the equations of motion for the restricted three-body problem are

$$\ddot{x} = x + 2\dot{y} - \mu' \frac{(x+\mu)}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{(x-\mu')}{((x-\mu')^2+y^2)^{3/2}} ,$$

$$\ddot{y} = y - 2\dot{x} - \mu' \frac{y}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{y}{((x-\mu')^2+y^2)^{3/2}} .$$

Here the two massive bodies are located on the x axis with the center of mass of the system at the origin, μ is the ratio of the mass of the body located on the positive x axis to the mass of the entire system, and μ' is the ratio of the mass of the body located on the negative x axis to the mass of the entire system ($\mu+\mu'=1$). The units of distance here are chosen so that the distance between the two massive bodies is unity, and the unit of time is chosen so that the angular velocity of the rotating reference frame is unity (period $= 2\pi$).

In this project μ was selected as about 0.012, the approximate value for the earth-moon system. The three orbits that were used to test the various methods are pictured in Figures 1, 2, and 3, and will be referred to as orbits 1, 2, and 3, respectively. Orbit 1 is an example of an orbit that comes relatively close to the earth but not close to the moon, while orbits 2 and 3 both come close to the moon but not so close to the earth.

The initial conditions for the Arenstorf orbits are obtained by trial and error correcting of half orbit runs. The periodic orbits are known to be symmetric with respect to the x-axis. Starting with initial conditions of $x = x_0$, $y = 0$ and $\dot{x} = 0$, $\dot{y}_0$ is varied so as to make $\dot{x} \approx 0$ (reduce $|\dot{x}|$ below some preassigned value) at some particular x-axis crossing.

Both the Adams and the Shanks methods were used to obtain initial conditions for each orbit. The two methods agreed with each other on what the initial conditions should be in each case to 19 significant figures. An average of the initial conditions obtained by the two methods was then used and is probably good to 20 significant figures for each orbit. It should be noted that even if the starting values are correct to 20 significant figures the error at the end of a complete orbit may be larger than this; for example, in orbits 2 and 3 a change of 10^{-19} in the initial values produces a change of 10^{-17} in the final values. In any case the initial conditions were at least good enough to allow a return to the starting conditions to within better than 5×10^{-17} for orbits 2 and 3, and 5×10^{-19} for orbit 1, independent of the method used for the integration.

Figure 1. Orbit Number 1.

The initial conditions and parameters for this orbit are:

$$x_0 = 1.2 \ ,$$

$$y_0 = 0 \ ,$$

$$\dot{x}_0 = 0 \ ,$$

$$\dot{y}_0 = -\ 1.04935750983031990726 \ ,$$

$$\mu = 0.012128562765312310491206 8 \ ,$$

$$\text{period} = 6.192169331319639706 74 \ .$$

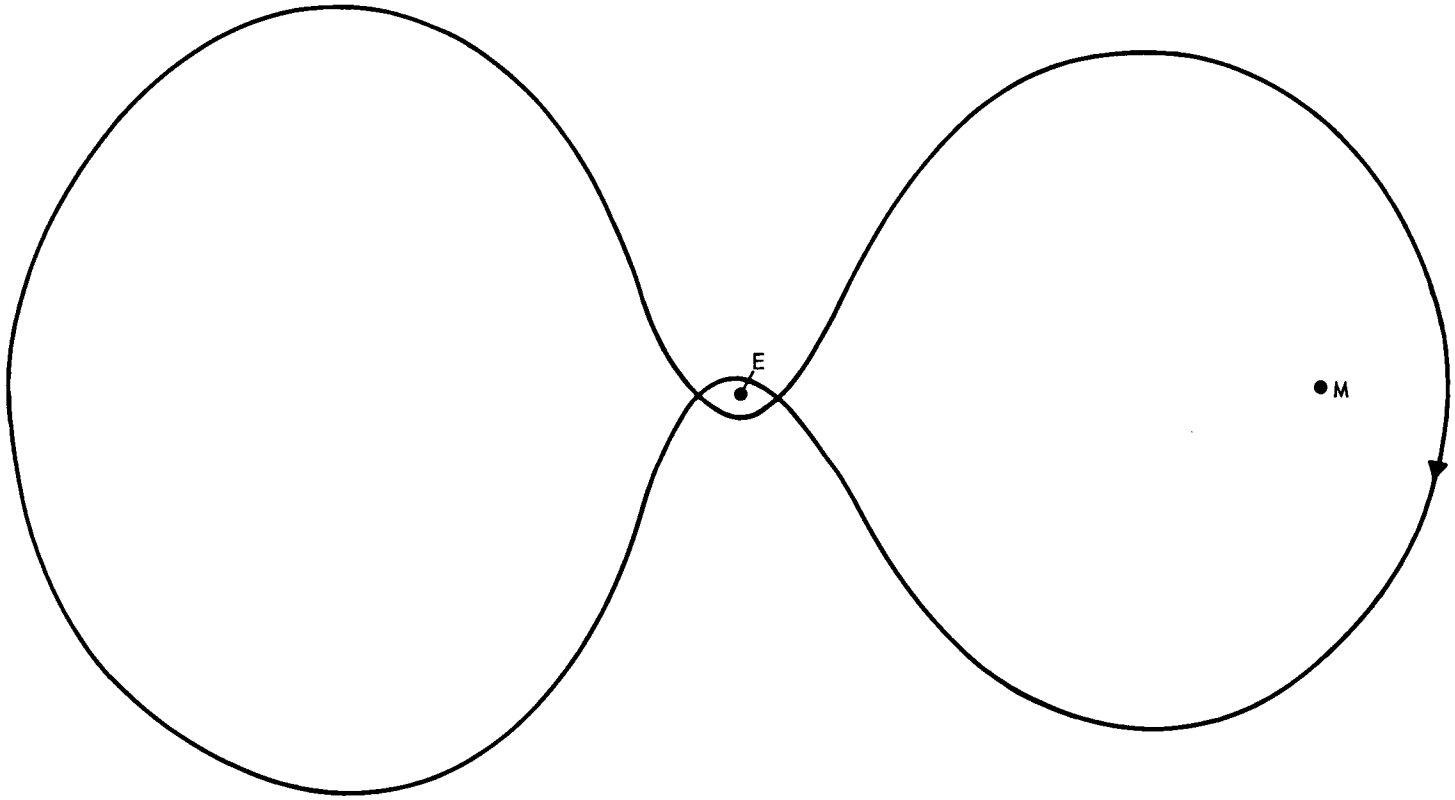

Figure 1. Orbit Number 1.

Figure 2. Orbit Number 2.

The initial conditions and parameters for this orbit are:

$$x_0 = 0.994 \ ,$$

$$y_0 = 0 \ ,$$

$$\dot{x}_0 = 0 \ ,$$

$$\dot{y}_0 = -\ 2.03173262955733683566 \ ,$$

$$\mu = 0.012277471 \ ,$$

$$\text{period} = 11.1243403372660851350 70 \ .$$

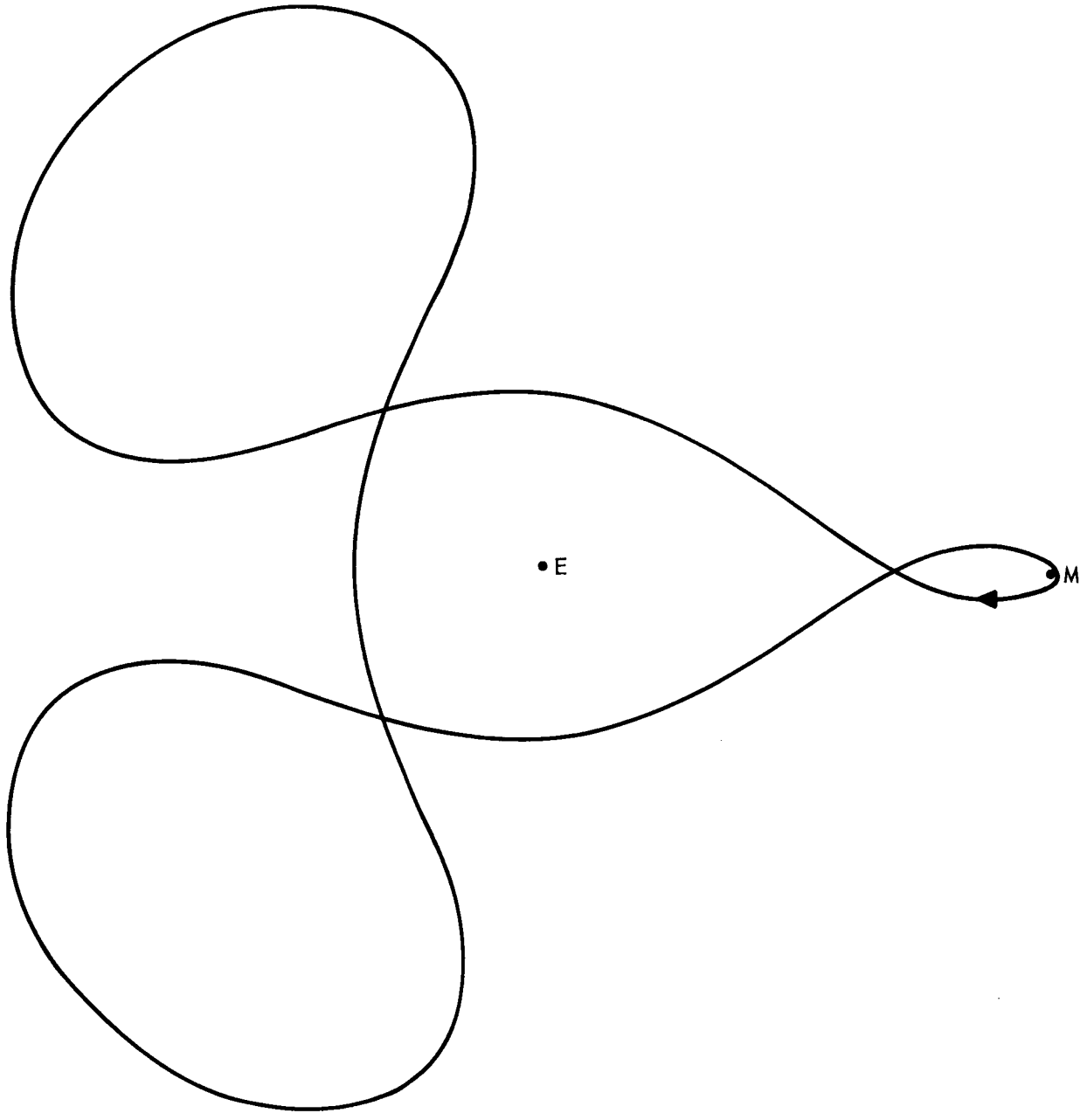

Figure 2. Orbit Number 2.

Figure 3. Orbit Number 3.

The initial conditions and parameters for this orbit are:

$$x_0 = 0.994 \ ,$$

$$y_0 = 0 \ ,$$

$$\dot{x}_0 = 0 \ ,$$

$$\dot{y}_0 = -\ 2.11389879669450266823 \ ,$$

$$\mu = 0.012277471 \ ,$$

$$\text{period} = 5.4367954392601899689794 5 \ .$$

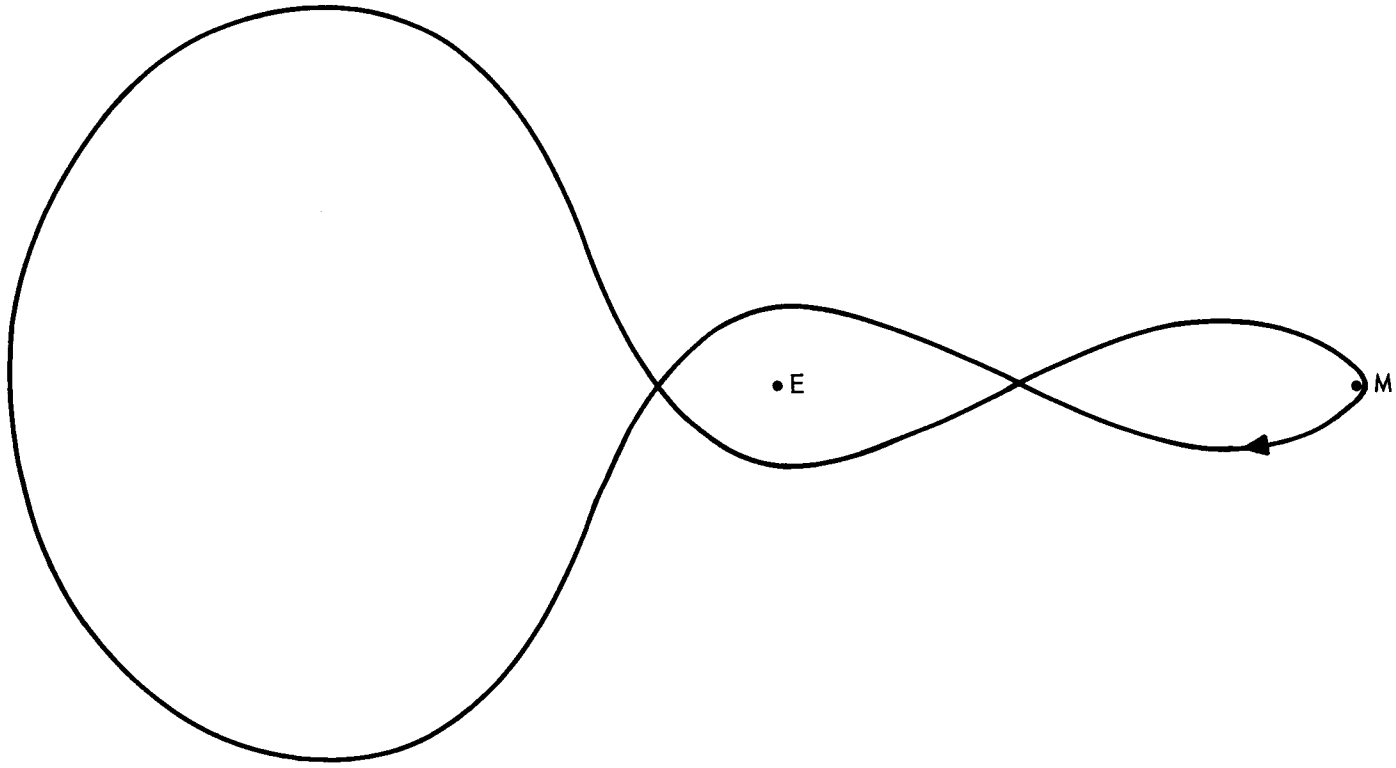

Figure 3. Orbit Number 3.

IV. INTEGRATION METHODS

A. The Lie Series Method of Groebner

1. Description of the Method

The differential equations of the restricted three-body problem in the rotating coordinate system are

$$\ddot{x} = x + 2\dot{y} - \mu' \frac{(x+\mu)}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{(x-\mu')}{((x-\mu')^2+y^2)^{3/2}} ,$$

$$\ddot{y} = y - 2\dot{x} - \mu' \frac{y}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{y}{((x-\mu')^2+y^2)^{3/2}} ,$$

where $\mu' = 1 - \mu$.

The initial conditions will be written as

$$x(0) \equiv x_0 \qquad \dot{x}(0) \equiv \dot{x}_0 ,$$

$$y(0) \equiv y_0 \qquad \dot{y}(0) \equiv \dot{y}_0 .$$

It is convenient to introduce the following vector notation:

$$\vec{x} \equiv \begin{pmatrix} x \\ y \end{pmatrix} \qquad \vec{u} \equiv \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \qquad \vec{\theta} \equiv \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} ,$$

$$\vec{e} \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \vec{u}^{+} \equiv \begin{pmatrix} \dot{y} \\ -\dot{x} \end{pmatrix} \qquad \vec{\theta}^{+} \equiv \begin{pmatrix} \ddot{y} \\ -\ddot{x} \end{pmatrix} ,$$

$$\vec{a} \equiv \vec{x} + \mu\vec{e} \qquad \vec{b} \equiv \vec{x} - \mu'\vec{e} ,$$

$$a \equiv |\vec{a}| \qquad b \equiv |\vec{b}| .$$

The equations of motion are then written

$$\vec{\theta} = \vec{x} + 2\vec{u}^{+} - \mu' \frac{\vec{a}}{a^3} - \mu \frac{\vec{b}}{b^3} ,$$

with initial conditions

$$\vec{x}(0) \equiv \vec{x}_0 \quad \text{and} \quad \vec{u}(0) \equiv \vec{u}_0 .$$

To express the Lie Series it is further convenient to introduce the operations

$$D_1 \equiv \vec{u} \cdot \frac{\partial}{\partial \vec{x}} + \vec{\phi} \cdot \frac{\partial}{\partial \vec{u}} + \frac{\partial}{\partial t} ,$$

$$D_2 \equiv \vec{\delta} \cdot \frac{\partial}{\partial \vec{u}} ,$$

$$D \equiv D_1 + D_2 ,$$

where

$$\vec{\delta} \equiv \vec{\theta} - \vec{\phi}$$

and $\vec{\phi}$ is yet to be chosen. The derivatives of interest are of the form $D^{\alpha} \vec{x}$ and $D_2 D^{\alpha} \vec{x}$ which for α from 0 to 3 are:

$$D^0 \vec{x} = \vec{x} ,$$

$$D^1 \vec{x} = \vec{u} ,$$

$$D^2 \vec{x} = \vec{\theta} ,$$

$$D^3 \vec{x} = \vec{u} + 2\vec{\theta}^{+} - \frac{\mu'}{a^3} (\vec{u} - 3\, \hat{a}.\vec{u}\, \hat{a}) + \frac{\mu}{b^3} (\vec{u} - 3\, \hat{b}.\vec{u}\, \hat{b}) ,$$

and

$$D_2 D^0 \vec{x} = 0 ,$$

$$D_2 D^1 \vec{x} = \vec{\delta} ,$$

$$D_2 D^2 \vec{x} = 2\vec{\delta}^+ ,$$

$$D_2 D^3 \vec{x} = -3\vec{\delta} - \frac{\mu'}{a^3} [\vec{\delta} - 3\,\hat{a}.\vec{\delta}\,\hat{a}]$$

$$- \frac{\mu}{b^3} [\vec{\delta} - 3\,\hat{b}.\vec{\delta}\,\hat{a}] \equiv \vec{\eta} ,$$

where

$$\vec{\delta}^+ = \begin{pmatrix} \delta_2 \\ -\delta_1^2 \end{pmatrix}, \quad \hat{a} = \frac{\vec{a}}{a} , \quad \hat{b} = \frac{\vec{b}}{b} .$$

The Lie Series solution can then be written

$$\vec{x}^*(t') = \vec{x}_a(t') + \sum_{\alpha=1} \int_{t_0}^{t'} \frac{(t'-\tau)^\alpha}{\alpha!} \left[D_2 D^\alpha \vec{x}_a(t) \right]_{t=\tau} d\tau ,$$

$$\vec{u}^*(t') = \vec{u}_a(t') + \sum_{\alpha=1} \int_{t_0}^{t'} \frac{(t'-\tau)^{\alpha-1}}{(\alpha-1)!} \left[D_2 D^\alpha \vec{x}_a(t) \right]_{t=\tau} d\tau ,$$

where

$$\vec{x}_a(t) = \vec{x}(t_0) + \sum_{\alpha=1} \frac{(t-t_0)^\alpha}{\alpha!} \left[D^\alpha \vec{x}(t) \right]_{t=t_0} ,$$

$$\vec{u}_a(t) = \vec{u}(t_0) + \sum_{\alpha=2} \frac{(t-t_0)^{\alpha-1}}{(\alpha-1)!} \left[D^\alpha \vec{x}(t) \right]_{t=t_0} .$$

Written out explicitly for α running up to 3 this gives

$$\vec{x}^*(t') = \vec{x}_a(t') + \int_{t_0}^{t'} \left[(t'-\tau)\, \vec{\delta}_a(\tau) + \frac{(t'-\tau)^2}{2!}\, 2\vec{\delta}_a^{\,+}(\tau) + \frac{(t'-\tau)^3}{3!}\, \eta_a(\tau) \right] d\tau \ ,$$

$$u^*(t') = \vec{u}_a(t') + \int_{t_0}^{t'} \left[\vec{\delta}_a(\tau) + (t'-\tau)\, 2\vec{\delta}_a^{\,+}(\tau) + \frac{(t'-\tau)^2}{2!}\, \vec{\eta}_a(\tau) \right] d\tau \ ,$$

$$\vec{x}_a(t) = \vec{x}(t_0) + (t-t_0)\, \dot{\vec{u}}(t_0) + \frac{(t-t_0)^2}{2!} \left[D^2 \vec{x}(t) \right]_{t=t_0} + \frac{(t-t_0)^3}{3!} \left[D^3 \vec{x}(t) \right]_{t=t_0} \ ,$$

$$\vec{u}_a(t) = \vec{u}(t_0) + (t-t_0) \left[D^2 \vec{x}(t) \right]_{t=t_0} + \frac{(t-t_0)^2}{2!} \left[D^3 \vec{x}(t) \right]_{t=t_0}$$

It should be noted here that $\vec{\delta}_a(\tau)$ and $\vec{\eta}_a(\tau)$ are $\vec{\delta}$ and $\vec{\eta}$ evaluated at $\vec{x}_a(\tau)$.

As yet $\vec{\phi}$ is undefined; it is to be chosen so as to give the best possible first approximation to $\vec{\theta}$. In this work $\vec{\phi}$ was chosen as

$$\vec{\phi}(t') = \left[D^2 \vec{x}(t) \right]_{t=t_0} + (t'-t_0) \left[D^3 \vec{x}(t) \right]_{t=t_0}$$

1.1. Evaluation of the Integrals

The evaluation of the integrals in the Lie Series was accomplished by approximating

$$\int_{t_0}^{t} \frac{(t-\tau)^{\alpha}}{\alpha!} \; g_{\alpha}(\tau) \; d\tau$$

by

$$\frac{(t-t_0)^{\alpha+1}}{(\alpha+1)!} \sum_{k=0}^{\sigma} C^{\sigma}{}_{k\alpha} \; g_{\alpha}\left(t_0 + k\,(t-t_0)/\sigma\right) \quad ,$$

where

$$C^{\sigma}{}_{k\alpha} = \sum_{\nu=k}^{\sigma} \frac{\nu!}{(\nu-k)!\,k!} \; A^{\sigma}{}_{\nu\alpha} \; (-1)^{\nu+k} \quad ,$$

and the $A^{\sigma}{}_{\nu\alpha}$ are given by

$$A^{\sigma}_{0\alpha} = 1 \quad ,$$

$$A^{\sigma}_{1\alpha} = \frac{\sigma}{(\alpha+2)} \quad ,$$

$$A^{\sigma}_{2\alpha} = \frac{\sigma}{2} \; \frac{2\sigma - (\alpha+3)}{(\alpha+2)(\alpha+3)} \quad ,$$

$$A^{\sigma}_{3\alpha} = \frac{\sigma}{3} \; \frac{3\sigma^2 - 3\sigma(\alpha+4) + (\alpha+3)(\alpha+4)}{(\alpha+2)(\alpha+3)(\alpha+4)} \quad ,$$

$$A^{\sigma}_{4\alpha} = \frac{\sigma}{12} \; \frac{12\sigma^3 - 18\sigma^2(\alpha+5) + 11\sigma(\alpha+4)(\alpha+5) - 3(\alpha+3)(\alpha+4)(\alpha+5)}{(\alpha+2)(\alpha+3)(\alpha+4)(\alpha+5)} \quad .$$

This is equivalent to a Newton-Cotes integration of degree σ. For example, when $\sigma = 2$ it is equivalent to Simpson's rule.

1.2. Error Estimates

The estimate of the error in $\vec{\theta}$ is given by

$$\vec{P}(t) = \vec{\theta}^{\,*}(t) - \vec{\theta}_a(t) - \int_{t_0}^{t} \sum_{\alpha=2} \frac{(t-\tau)^{\alpha-2}}{(\alpha-2)!} \left[D_2 \, D^{\alpha} \vec{x}_a(t) \right]_{t=\tau} d\tau \ ,$$

or

$$\vec{P}(t) = \vec{\theta}^{\,*}(t) - \vec{\theta}(t) - \int_{t_0}^{t} \left[2\vec{\delta}^{\,+}_a(\tau) + (t-\tau)\, \vec{\eta}_a(t) \right] d\tau \ ,$$

where again $\vec{\theta}_a(\tau)$, $\vec{\delta}_a^{\,+}(\tau)$ and $\vec{\eta}_a(\tau)$ are to be evaluated at $\vec{x}_a(\tau)$. The estimated error in the velocity and position are respectively

$$\vec{q}(t) = \frac{(t-t_0)}{4} \vec{P}(t) \text{ and } \vec{p}(t) = \frac{(t-t_0)^2}{20} \vec{P}(t) \ .$$

The integrals in the error terms are carried out in the same way as described in the previous paragraph on integration.

1.3. Step Size Control

The step size control is based on the set of constants γ, γ_1, γ_2, γ^*, γ_k and γ_{max}.

At the end of a step no change is made in Δt if

$$p \leq \gamma_1 \quad \text{and} \quad q \leq \gamma_2$$

and either $p \geq \gamma$ or $q \geq \gamma$.

Otherwise the step size is changed by the relation

$$\Delta t_{next} = \frac{\Delta t_{last}}{4} \left(3 + \frac{\gamma^*}{R+\gamma} \right) \ .$$

In the above criterion

$$p = |\vec{p}|, \quad q = |\vec{q}| \ ,$$

$$\gamma = (\gamma_1 + \gamma_2)/2 \ ,$$

$$\gamma^* = \gamma + \gamma_k \ ,$$

$$\gamma_k = \begin{cases} \gamma_1 & \text{if } p \geq q \\ \gamma_2 & \text{if } p < q \end{cases} \ ,$$

and R = maximum of p and q. This results in the maximum increase or decrease in Δt at any step being $\frac{3}{2}$ or $\frac{3}{4}$ respectively of the previous Δt.

The constants γ_1 and γ_2 are selected on the first step by halving the step size until $|\vec{P}| < \gamma_{max}$ and assigning $\gamma_1 = p$ and $\gamma_2 = q$. The constant γ_{max} is an input constant.

While there is some theoretical foundation for this system of step size control it is to a large extent empirical.

2. The Lie Series Computer Program

The program itself is written in double precision (23 decimal places) arithmetic for the Burroughs B-5000 in a language called "Extended Algol for the B-5000." This language is close to Algol-60 except that all double precision arithmetic is written in Polish prefix notation. There are no unusual hardware requirements. The program will run on a minimum B-5000 system.

The program assumes that all orbits will start on the x-axis (axis of symmetry for the orbit) with the x component of velocity zero, and will return reasonably close to the starting point.

2.1. Closing an Orbit

The orbit is completed by first running to the end of the period (as given on input data card no. 7) and then interpolating to a time that would

give y = 0. The error in closing the orbit can then be taken as either the error in x, y, $\dot{x}$, and $\dot{y}$ or the error in x, $\dot{x}$, $\dot{y}$ and the period.

2.2. Searching for Improved Initial Conditions

At a time close to the half period (as given by the input data) a search is made for a crossing of the x axis in a direction specified by data input card no. 4. The time interval is then reduced until a value of y is found such that $|y| < \gamma_1/16$. The value of $\dot{x}$ at this point and the value of $\dot{x}$ on the previous half orbit are used together with the initial $\dot{y}$ of this and the previous orbit to do a linear interpolation for a new initial $\dot{y}$ that will further reduce the magnitude of $\dot{x}$. Half orbits are run until $|\dot{x}| < 16\gamma_2$.

2.3. Data Output

The following is a description of the output:

The first line printed is the information on the first six cards of the data input together with identification of each item. Here σ is referred to as SIGMA, γ_{max} as GMAX, and Δt, the initial trial step size as DT.

The orbit information is printed at every N steps, where N is indicated on the data input card no. 5, including the zeroth step (initial values). The information printed out is:

the step number (labeled STEP),

the processor time in seconds (labeled TIME),

the estimated error in $\vec{\theta}$ (labeled P),

Δt for the step just taken (labeled DT),

the x position coordinate (labeled X),

the y position coordinate (labeled Y),

the time, t, for this position (labeled T),

the velocity, $\dot{x}$ (labeled U),

the velocity, $\dot{y}$ (labeled V).

At the final step (at the end of the time interval equal to the period specified on input data card no. 7) the orbit information is also printed. An estimate is then made of the Δt which would make y = zero ($\Delta t = - y/\dot{y}$), and the interpolated value of $\dot{x}$ calculated ($\dot{x}_{int} = \dot{x}_{final} + \Delta t\ \ddot{x}$). The errors for the orbit closure are then printed out in this order:

the differences between the initial and final value of x (labeled DX),

the value of y at the end of the period given on input data card no. 7 (labeled DY),

the value of $\dot{x}$ at the end of the period given on input data card no. 7 (labeled DU),

the difference between the initial and final value of $\dot{y}$ (labeled DV),

the time interval necessary to interpolate y to zero (labeled DT),

the interpolated value of $\dot{x}$ at the time when y is zero (labeled DU).

If a search for improved initial conditions is called for by the input data this additional information is printed:

the orbit information at half orbit,

the new period based on this half orbit (labeled PERIOD),

the new value of the initial $\dot{y}$ (labeled V NEW),

the difference between this starting value of $\dot{y}$ and that of the last half orbit (labeled DV),

the difference between this ending value of $\dot{x}$ and that of the last half orbit (labeled DU).

2.4. Data Input

The following is a description of the data input cards and an explanation of the function of the information on each card.

Card No. 1:

The value of σ to be used in the integration scheme. The value of σ can vary from 1 to 4 and gives the polynomial degree of the numerical integration. For example, $\sigma = 1$ is equivalent to trapezoid integration, $\sigma = 2$ Simpson's rule, etc. Note that $\sigma = 2$ and $\sigma = 3$ have the same order truncation error. This number must be a free field integer followed by a comma.

Card No. 2:

The value of γ_{max}, the error to be associated with $\vec{\theta}$. This number must be a free field single precision real number followed by a comma.

Card No. 3:

A "YES" or a " NO" to indicate if a search is to be made for improved initial conditions or not. This entry must be a three character free field literal followed by a comma.

Card No. 4:

A +1 or a -1 indicating the direction of the x axis crossing at half orbit. If the half orbit crossing is in the +y direction ($\dot{y}$ positive) then +1, if in the -y direction then -1. This information is used in searching for improved initial conditions, but must also be present in the input data even if card no. 3 is " NO". This number must be free field integer followed by a comma.

Card No. 5:

This card contains an integer to indicate how often orbit information is to be printed out. A 1 causes a print at every step, a 2 causes print every other step, a 3 causes a print every third step, etc. Regardless of the value of this number a print out will take place at the beginning and end of every orbit or at the end of every half orbit if a search for improved initial conditions is being made. This number must be a free field integer followed by a comma.

Card No. 6:

An initial trial value for Δt, the time step size. If this trial value of Δt produces too large an error (i.e. $|\vec{P}| > \gamma_{max}$), the step size will be halved until $|\vec{P}| < \gamma_{max}$ before the first step is accepted. This number must be a free field single precision real number followed by a comma.

Card No. 7:

This card contains the period of the orbit to be run. This number must be a free field double precision real number not followed by a comma.

Card No. 8:

This card contains the value of μ. This number must be a free field double precision real number not followed by a comma.

Card No. 9:

This card contains the initial value of the x coordinate. This number must be a free field double precision real number not followed by a comma.

Card No. 10:

This card contains the initial value of the y coordinate. If zero it must be so punched. This number must be a free field double precision real number not followed by a comma.

Card No. 11:

This card contains the initial value of $\dot{x}$ or U, the x component of the velocity. If zero it must be so punched. This number must be a free field double precision real number not followed by a comma.

Card No. 12:

This card contains the initial value of $\dot{y}$ or V, the y component of the velocity. This number must be a free field double precision real number not followed by a comma.

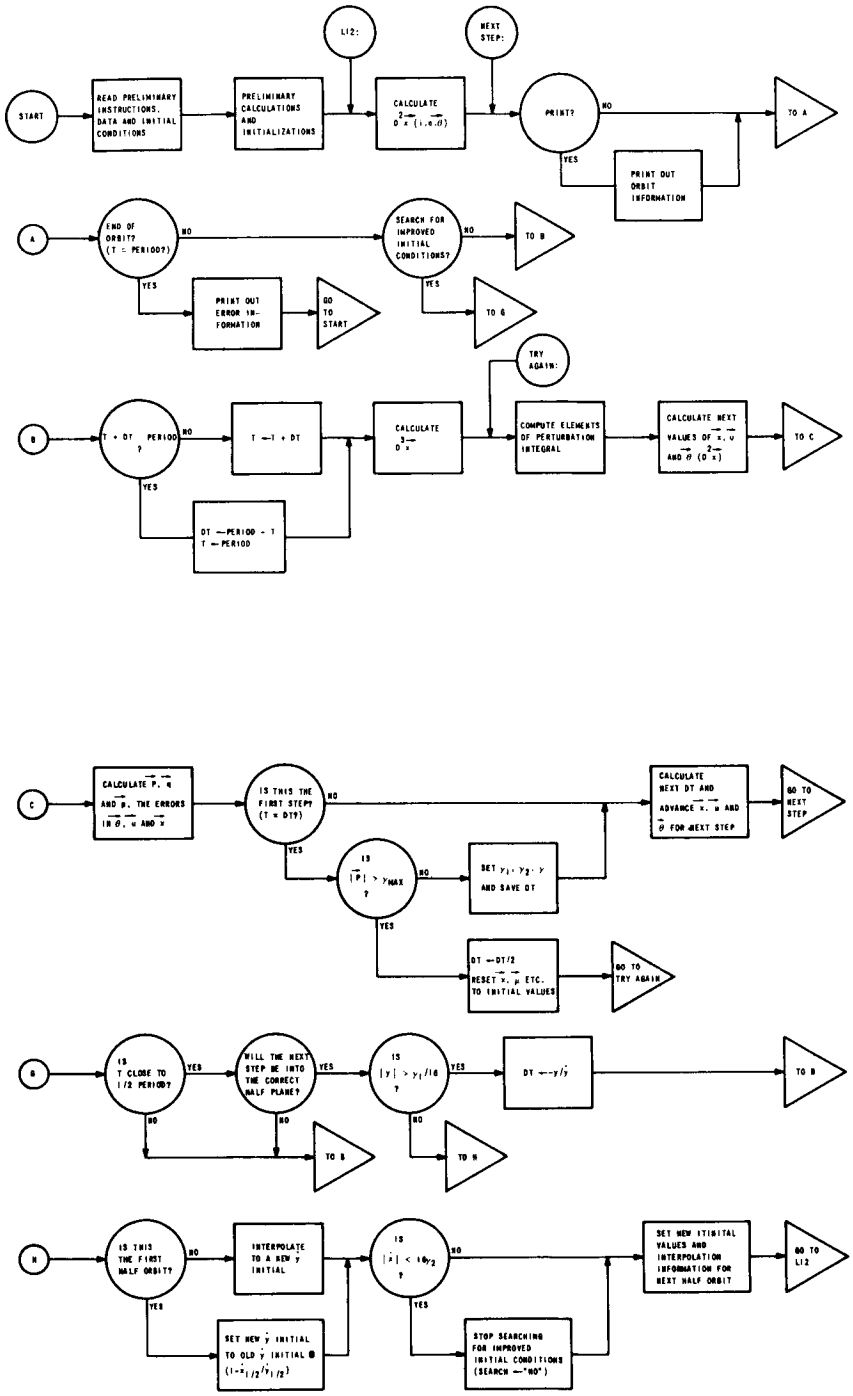

Figure 4. The Flow Diagram for the Lie Series Method.

Comments may be put on the data cards by observing the following rules. For free field integers and single precision real numbers a slash, /, indicates that the remainder of the card is to be ignored. For double precision real numbers an asterisk, *, indicates that the remainder of the card is to be ignored. Comments can then be written on the data cards following the slash or the asterisk.

3. The Flow Diagram for the Lie Series Method

Figure 4 gives a detailed flow diagram of the computer program for the Lie Series method. The symbols and terminology used in the flow diagram correspond to those in the above discussion and in the program listing.

4. The Program Listing for the Lie Series Method

The following 18 pages list the program for the Lie Series Method.

```
 LABEL   000000000XXXXXX0010000001                                          00000000
                                                                            00001000
BEGIN                                                                       00002000
  COMMENT GALLAHER L J ;                                                    00003000
  FILE IN FILEIN (3,10);                                                    00004000
  FILE OUT FILEOUT 4(3,15);                                                 00005000
  COMMENT HERE PUT IN DPOP AND DREAD;                                       00006000
  DEFINE CARD =FILEIN #,PRINT =FILEOUT #;                                   00007000
  LABEL LAST;                                                               00008000
  DEFINE F =FILEOUT #;                                                      00009000
  ALPHA ARRAY A[0:77];                                                      00010000
  INTEGER EVER,NEVER ;                                                      00011000
  INTEGER ARRAY NOTUSE1[0:80],NOTUSE2[0:9];                                 00012000
  REAL ARRAY NOTUSE3[0:68],NOTUSE4[0:68];                                   00013000
  PROCEDURE DREAD(AH,AL);                                                   00014000
  REAL AH,AL ;                                                              00015000
                                                                            00016000
  BEGIN                                                                     00017000
    INTEGER K,T1 ;                                                          00018000
    REAL TH,TL ;                                                            00019000
    BOOLEAN DF,MF ;                                                         00020000
    LABEL ER,L1,L2,L3,L4,L5,L6,L7 ;                                         00021000
    FORMAT OUT FR(/"DATA CARD ERROR",X105);                                 00022000
    STREAM PROCEDURE EC(S,D);                                               00023000
                                                                            00024000
    BEGIN                                                                   00025000
      SI :=S ;                                                              00026000
      DI :=D ;                                                              00027000
      DS :=8 LIT "0";                                                       00028000
      2(40(DS :=7 LIT "0";                                                  00029000
      DS :=1 CHR));                                                         00030000
                                                                            00031000
    END ;                                                                   00032000
    PROCEDURE GN ;                                                          00033000
                                                                            00034000
    BEGIN                                                                   00035000
      LABEL L1 ;                                                            00036000
      IF NEVER ≤80 THEN GO TO L1 ELSE IF NEVER >81 THEN                     00037000
      BEGIN                                                                 00038000
        READ (CARD,10,NOTUSE2[*])[LAST];
```

```
    EC(NOTUSE2,NOTUSE1);                                          00039000
    NEVER :=1 ;                                                   00040000
    L1:EVER :=NOTUSE1[NEVER];                                     00041000
    IF EVER ="*"THEN                                              00042000
    BEGIN                                                         00043000
      NEVER :=81 ;                                                00044000
      EVER :=" "END ELSE                                          00045000
    END ELSE EVER :=" ";                                          00046000
    NEVER :=NEVER +1 ;                                            00047000
                                                                  00048000
  END ;                                                           00049000
  L1:GN ;                                                         00050000
  IF EVER =" "THEN GO TO L1 ;                                     00051000
  DF :=FALSE ;                                                    00052000
  K :=0 ;                                                         00053000
  TH :=0 ;                                                        00054000
  TL :=0 ;                                                        00055000
  IF EVER ="-"THEN MF :=TRUE ELSE                                 00056000
  BEGIN                                                           00057000
    MF :=FALSE ;                                                  00058000
    IF EVER ≠"+"THEN GO TO L3 ELSE                                00059000
  END ;                                                           00060000
  L2:GN ;                                                         00061000
  L3:IF EVER <10 THEN                                             00062000
  BEGIN                                                           00063000
    DF :=TRUE ;                                                   00064000
    DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL);                        00065000
    GO TO L2                                                      00066000
  END ;                                                           00067000
  IF EVER =".."THEN                                               00068000
  BEGIN                                                           00069000
    L4:GN ;                                                       00070000
    IF EVER <10 THEN                                              00071000
    BEGIN                                                         00072000
      K :=K -1 ;                                                  00073000
      DF :=TRUE ;                                                 00074000
      DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL);                      00075000
      GO TO L4                                                    00076000
    END ELSE                                                      00077000
  END ;                                                           00078000
```

```
    IF MF THEN DOUBLE(,0,TH,TL,-,:=,TH,TL);                                  00079000
    IF NOT DF THEN                                                           00080000
    BEGIN                                                                    00081000
      ER:WRITE(PRINT,FR);                                                    00082000
      GO TO LAST                                                             00083000
    END ;                                                                    00084000
    IF EVER =" "THEN GO TO L7 ;                                              00085000
    IF EVER ≠"@"THEN GO TO ER ;                                              00086000
    GN ;                                                                     00087000
    T1 :=0 ;                                                                 00088000
    DF :=FALSE ;                                                             00089000
    IF EVER ="-"THEN MF :=TRUE ELSE                                          00090000
    BEGIN                                                                    00091000
      MF :=FALSE ;                                                           00092000
      IF EVER ≠"+"THEN GO TO L6 ELSE                                         00093000
    END ;                                                                    00094000
    L5:GN ;                                                                  00095000
    L6:IF EVER <10 THEN                                                      00096000
    BEGIN                                                                    00097000
      DF :=TRUE ;                                                            00098000
      T1 :=T1 ×10 +EVER ;                                                    00099000
      GO TO L5                                                               00100000
    END ;                                                                    00101000
    IF NOT DF THEN GO TO ER ;                                                00102000
    IF EVER ≠" "THEN GO TO ER ;                                              00103000
    IF MF THEN K :=K -T1 ELSE K :=K +T1 ;                                    00104000
    L7:IF K <0 THEN DOUBLE(TH,TL,NOTUSE3[-K],NOTUSE4[-K],/,:=,AH,AL          00105000
    )ELSE DOUBLE(TH,TL,NOTUSE3[K],NOTUSE4[K],×,:=,AH,AL);                    00106000
                                                                             00107000
END ;                                                                        00108000
PROCEDURE DPOP(A1,A2,ALF );                                                  00109000
VALUE A1,A2 ;                                                                00110000
REAL A1,A2 ;                                                                 00111000
ALPHA ARRAY ALF[0];                                                          00112000
                                                                             00113000
BEGIN                                                                        00114000
  STREAM PROCEDURE PRINT(X,Y,N,F,A,ALF);                                     00115000
  VALUE X,Y,N ;                                                              00116000
                                                                             00117000
  BEGIN                                                                      00118000
```

```
LOCAL V1,V11,V2,V22,V3,V4,ST,RP ;                                    00119000
DI ←LOC V22;                                                         00120000
SI ←LOC X ;                                                          00121000
SI ←SI+1 ;                                                           00122000
SKIP 3 SB ;                                                          00123000
13(DS ←3 RESET ;                                                     00124000
3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                00125000
SKIP 1 SB));                                                         00126000
SI ←LOC Y ;                                                          00127000
SI ←SI+1 ;                                                           00128000
SKIP 3 SB ;                                                          00129000
13(DS ←3 RESET ;                                                     00130000
3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                00131000
SKIP 1 SB));                                                         00132000
DI ←LOC RP ;                                                         00133000
DS ←8 LIT"00000000";                                                 00134000
DI ←LOC ST ;                                                         00135000
SI ←F ;                                                              00136000
DS ←WDS ;                                                            00137000
DI ←ST ;                                                             00138000
SI ←LOC X ;                                                          00139000
SKIP 1 SB ;                                                          00140000
IF SB THEN DS ←8 LIT"     -0."ELSE DS ←8 LIT"      0.";              00141000
3(DS ←8 LIT"00000000");                                              00142000
11(DS ←8 LIT"         ");                                            00143000
SI ←A ;                                                              00144000
V4 ←SI ;                                                             00145000
SI ←LOC V22;                                                         00146000
V3 ←SI ;                                                             00147000
26(DI ←LOC RP ;                                                      00148000
DI ←DI +7 ;                                                          00149000
DS ←CHR ;                                                            00150000
SI ←V4 ;                                                             00151000
DI ←ST ;                                                             00152000
DI ←DI+8 ;                                                           00153000
RP(DS ←24 ADD ;                                                      00154000
DI ←DI -24 ;                                                         00155000
SI ←SI -24);                                                         00156000
SI ←SI +24 ;                                                         00157000
V4 ←SI ;                                                             00158000
```

```
    SI ←V3 ;                                                    00159000
    SI ←SI+1 ;                                                  00160000
    V3 ←SI);                                                    00161000
    DI ←DI +24 ;                                                00162000
    SI ←LOC N ;                                                 00163000
    SKIP 1 SB ;                                                 00164000
    IF SB THEN DS ←2 LIT "@-"ELSE DS ←2 LIT "@+";               00165000
    SI ←LOC N ;                                                 00166000
    DS ←2 DEC ;                                                 00167000
    DI ←DI-1 ;                                                  00168000
    2(DS ←RESET);                                               00169000
    DI ←ALF ;                                                   00170000
    SI ←ST ;                                                    00171000
    SI ←SI +5 ;                                                 00172000
    DI ←DI +2 ;                                                 00173000
    DS ←CHR ;                                                   00174000
    SI ←SI +2 ;                                                 00175000
    DS ←CHR ;                                                   00176000
    DS ←1 LIT".";                                               00177000
    DS ←3 CHR ;                                                 00178000
    4(DI ←DI +2 ;                                               00179000
    DS ←6 CHR);                                                 00180000
                                                                00181000
END ;                                                           00182000
STREAM PROCEDURE SHIFT(A,B,C);                                  00183000
VALUE A,B ;                                                     00184000
                                                                00185000
BEGIN                                                           00186000
    SI ←LOC A ;                                                 00187000
    DI ←C ;                                                     00188000
    DS ←CHR ;                                                   00189000
    SKIP 3 SB ;                                                 00190000
    DI ←DI +1 ;                                                 00191000
    36(IF SB THEN DS ←SET ELSE DS ←RESET ;                      00192000
    SKIP 1 SB);                                                 00193000
    SKIP 9 DB ;                                                 00194000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                       00195000
    SKIP 1 SB);                                                 00196000
    SI ←LOC B ;                                                 00197000
    SKIP 9 SB ;                                                 00198000
```

```
    36(IF SB THEN DS ←SET ELSE DS ←RESET );                                  00199000
    SKIP 1 SB)};                                                             00200000
                                                                             00201000
  END };                                                                     00202000
  INTEGER Y,I };                                                             00203000
  REAL T1,T2,HT ,LT};                                                        00204000
  ARRAY T[0:1]};                                                             00205000
  LABEL L1,L2,L3 };                                                          00206000
  IF A1 =0 THEN                                                              00207000
  BEGIN                                                                      00208000
    T1 ←T2 ←0 };                                                             00209000
    GO TO L2                                                                 00210000
  END };                                                                     00211000
  HT ←1 };                                                                   00212000
  LT ←0 };                                                                   00213000
  Y ←0.90309 ×(IF BOOLEAN(A1.[2:1])THEN 13 -A1.[3:6]ELSE 13 +A1.[3:600214000
  ])};                                                                       00215000
  FOR I ←1 STEP 1 UNTIL ABS(Y)DO DOUBLE(HT,LT,,10,×,←,HT,LT)};               00216000
  IF Y>0 THEN DOUBLE(A1,A2,HT,LT,/,←,T1,T2)ELSE DOUBLE(A1,A2,HT,LT,×00217000
  ,←,T1,T2)};                                                                00218000
  Y ←Y-1 };                                                                  00219000
  L1:IF T1.[3:6]<13 THEN                                                     00220000
  BEGIN                                                                      00221000
    DOUBLE(T1,T2,,10.0,/,←,T1,T2)};                                          00222000
    Y ←Y+1                                                                   00223000
  END ELSE IF T1.[3:6]>13 THEN                                               00224000
  BEGIN                                                                      00225000
    DOUBLE(T1,T2,,10.0,×,←,T1,T2)};                                          00226000
    Y ←Y-1                                                                   00227000
  END ELSE GO TO L2 };                                                       00228000
  IF T1.[3:6]=14 THEN                                                        00229000
  BEGIN                                                                      00230000
    SHIFT(T1,T2,T)};                                                         00231000
    PRINT(T[0],T[1],Y,F,A,ALF)};                                             00232000
    GO TO L3                                                                 00233000
  END ELSE GO TO L1 };                                                       00234000
  L2:PRINT(T1,T2,Y,F,A,ALF )};                                               00235000
  L3:                                                                        00236000
END };                                                                       00237000
FILL A[*]WITH "12500000","00000000","00000000","01562500","00000000"00238000
```

```
    ,"00000000","00195312",,"50000000",,"00000000",,"00024414","06250000","00239000
00000000","00003051",,"75781250","00000000",,"00000381","46972656","25000000240000
00","00000047",,"68371582","03125000",,"00000005",,"96046447","75390625",,"000241000
0000000",,"74505805","96923828","00000000","09313225","74615479","000000000242000
0","01164153","21826935","00000000","00145519","15228367","00000000",,"0000243000
018189","89403546","00000000","00002273","73675443","00000000","000028400244000
","21709430","00000000",,"00000035","52713679","00000000",,"00000004","44000245000
89210","00000000","00000000","55511151","00000000",,"00000000",,"06938894"00246000
    ,"00000000","00000000","00867362","00000000",,"00000000","00108420","00247000
00000000","00000000",,"00013553","00000000","00000000","00001694","00000000248000
00","00000000","0 000212","00000000","00000000","00000026","00000000",,"000249000
0000000","00000003"};                                                          00250000
                                                                               00251000
    BEGIN                                                                      00252000
      INTEGER I ;                                                              00253000
      DOUBLE(,1,:=,NOTUSE3[0],NOTUSE4[0]);                                     00254000
      FOR I :=1 STEP 1 UNTIL 33 DO DOUBLE(,10,NOTUSE3[I -1],,NOTUSE4[I -100255000
      ],x,:=,NOTUSE3[I],,NOTUSE4[I]);                                          00256000
      FOR I :=1 STEP 1 UNTIL 35 DO DOUBLE(NOTUSE3[33],,NOTUSE4[33],,          00257000
      NOTUSE3[I],,NOTUSE4[I],x,:=,NOTUSE3[I +33],,NOTUSE4[I +33]);            00258000
      NEVER :=82 ;                                                             00259000
                                                                               00260000
    END ;                                                                      00261000
                                                                               00262000
    BEGIN                                                                      00263000
      PROCEDURE MON(A,H,,L);                                                   00264000
      REAL H,,L;                                                               00265000
      ALPHA A;                                                                 00266000
                                                                               00267000
      BEGIN                                                                    00268000
        ALPHA ARRAY ALEPH[0:4];                                                00269000
        INTEGER J;                                                             00270000
        FORMAT FM(A6,,X5,,5A6);                                                00271000
        DPOP(H,,L,,ALEPH);                                                     00272000
        WRITE(FILEOUT,,FM,,A,,FOR J :=0 STEP 1 UNTIL 4 DO ALEPH[J]);           00273000
                                                                               00274000
      END;                                                                     00275000
      INTEGER KA,,SIGMA,,PRFLAG;                                               00276000
      REAL GAM,,GAM1,,GAM2,,GAMK,,GMAX,,T2,,HPERIOD,,HDT,,HMU,,HXIN,,HYIN,,HUIN,,  00277000
      HVIN,,HMP,,LPERIOD,,LDT,,LMU,,LXIN,,LYIN,,LUIN,,LVIN,,LMP,,GAMSQ,,GAM1SQ,,  00278000
```

```
GAM2SQ,GMAXSQ};                                                          00279000
ALPHA SEARCH};                                                           00280000
COMMENT DEFINITIONS FOR BLOCK 1;                                         00281000
DEFINE XIN =HXIN ,LXIN #,PERIOD =HPERIOD ,LPERIOD #,YIN =HYIN ,          00282000
LYIN #,DT =HDT ,LDT #,UIN =HUIN ,LUIN #,MU =HMU ,LMU #,VIN =HVIN         00283000
,LVIN #,MP =HMP ,LMP #}                                                  00284000
LABEL START,FINISH};                                                     00285000
START:                                                                   00286000
BEGIN                                                                    00287000
  COMMENT READ PRELIMINARY DATA AND INITIAL CONDITIONS};                 00288000
  FORMAT FMPRELIM(//,"SIGMA =",I2,"   GMAX =",E10.2                      00289000
  ,"   SEARCH: ",A3,I3,"   PRINT EVERY",I5,"  STEPS    DT =",E13.500290000
  ,/)};                                                                  00291000
  LIST LSTPRELIM (SIGMA,GMAX,SEARCH,KA,PRFLAG,HDT)};                     00292000
  T2 :=TIME(2)};                                                         00293000
  LDT :=0;                                                               00294000
  READ (FILEIN,/,LSTPRELIM)[FINISH];                                     00295000
  WRITE(FILEOUT,FMPRELIM,LSTPRELIM)};                                    00296000
  DREAD(PERIOD)};                                                        00297000
  DREAD(MU)};                                                            00298000
  DREAD(XIN)};                                                           00299000
  DREAD(YIN)};                                                           00300000
  DREAD(UIN)};                                                           00301000
  DREAD(VIN)};                                                           00302000
                                                                         00303000
END};                                                                    00304000
                                                                         00305000
BEGIN                                                                    00306000
  COMMENT DECLARATIONS FOR BLOCK 2};                                     00307000
  INTEGER A,J,STEPCOUNT};                                                00308000
  REAL HDTP,HT,HTAU,HTAU2,HTAU3,R,HTEMP,LDTP,LT,LTAU,LTAU2,LTAU3,        00309000
  LTEMP,HSUMX1,HSUMU1,HNEXTX1,HNEXTU1,HNEXTTHETA1,LSUMX1,LSUMU1,         00310000
  LNEXTX1,LNEXTU1,LNEXTTHETA1,HSUMX2,HSUMU2,HNEXTX2,HNEXTU2,             00311000
  HNEXTTHETA2,LSUMX2,LSUMU2,LNEXTX2,LNEXTU2,LNEXTTHETA2,HA1,HB1,         00312000
  HAU1,HBU1,HTEMPX1,HTEMPU1,HALPH,HALPHSQ,HALPHCUBE,LA1,LB1,LAU1,        00313000
  LBU1,LTEMPX1,LTEMPU1,LALPH,LALPHSQ,LALPHCUBE,HA2,HB2,HAU2,HBU2,        00314000
  HTEMPX2,HTEMPU2,HBETA,HBETASQ,HBETACUBE,LA2,LB2,LAU2,LBU2,             00315000
  LTEMPX2,LTEMPU2,LBETA,LBETASQ,LBETACUBE,HMPDIVALPHCUBE,HAUDOTU3        00316000
  ,HAUDOTDELTA3,HCAP1,P1,Q1,PSQ,HD3X1,LMPDIVALPHCUBE,LAUDOTU3,           00317000
  LAUDOTDELTA3,LCAP1,LD3X1,HMUDIVBETACUBE,HBUDOTU3,HBUDOTDELTA3,         00318000
```

```
HCAP2,P2,Q2,QSQ,HD3X2,LMUDIVBETACUBE,LBUDOTU3,LBUDOTDELTA3,LCAP200319000
,LD3X2;                                                          00320000
ARRAY HCS,LCS[0:SIGMA,0:3],HG1,LG1,HG2,LG2[0:3,0:SIGMA],HX1A,    00321000
HU1A,HTHETA1A,HO1A,HDELTA1A,HETA1A,LX1A,LU1A,LTHETA1A,LO1A,      00322000
LDELTA1A,LETA1A,HX2A,HU2A,HTHETA2A,HO2A,HDELTA2A,HETA2A,LX2A,    00323000
LU2A,LTHETA2A,LO2A,LDELTA2A,LETA2A[0:SIGMA];                     00324000
LABEL NEXTSTEP,TRYAGAIN;                                         00325000
COMMENT DEFINITIONS,FOR BLOCK 2;                                 00326000
DEFINE MPDIVALPHCUBE=HMPDIVALPHCUBE,LMPDIVALPHCUBE#,SUMX1=HSUMX100327000
,LSUMX1#,MUDIVBETACUBE=HMUDIVBETACUBE,LMUDIVBETACUBE#,SUMX2     00328000
=HSUMX2,LSUMX2#,AUDOTU3 =HAUDOTU3 ,LAUDOTU3 #,SUMU1=HSUMU1,      00329000
LSUMU1#,BUDOTU3 =HBUDOTU3 ,LBUDOTU3 #,SUMU2=HSUMU2,LSUMU2#,      00330000
AUDOTDELTA3 =HAUDOTDELTA3 ,LAUDOTDELTA3 #,DTP =HDTP ,LDTP #,     00331000
BUDOTDELTA3 =HBUDOTDELTA3 ,LBUDOTDELTA3 #,T =HT ,LT #,TEMPX1     00332000
=HTEMPX1 ,LTEMPX1 #,TAU =HTAU ,LTAU #,TEMPX2 =HTEMPX2 ,LTEMPX2 #00333000
,TAU2 =HTAU2 ,LTAU2 #,TEMPU1 =HTEMPU1 ,LTEMPU1 #,TAU3 =HTAU3 ,   00334000
LTAU3 #,TEMPU2 =HTEMPU2 ,LTEMPU2 #,ALPHSQ =HALPHSQ ,LALPHSQ #,A100335000
=HA1 ,LA1 #,BETASQ =HBETASQ ,LBETASQ #,A2 =HA2 ,LA2 #,ALPHCUBE   00336000
=HALPHCUBE ,LALPHCUBE #,B1 =HB1 ,LB1 #,BETACUBE =HBETACUBE ,     00337000
LBETACUBE #,B2 =HB2 ,LB2 #,NEXTX1 =HNEXTX1 ,LNEXTX1 #,AU1 =HAU1 00338000
,LAU1 #,NEXTX2 =HNEXTX2 ,LNEXTX2 #,AU2 =HAU2 ,LAU2 #,NEXTU1      00339000
=HNEXTU1 ,LNEXTU1 #,BU1 =HBU1 ,LBU1 #,NEXTU2 =HNEXTU2 ,LNEXTU2 #00340000
,BU2 =HBU2 ,LBU2 #,NEXTTHETA1 =HNEXTTHETA1 ,LNEXTTHETA1 #,ALPH   00341000
=HALPH ,LALPH #,NEXTTHETA2 =HNEXTTHETA2 ,LNEXTTHETA2 #,BETA      00342000
=HBETA ,LBETA #,CAP1 =HCAP1 ,LCAP1 #,D3X1 =HD3X1 ,LD3X1 #,CAP2   00343000
=HCAP2 ,LCAP2 #,D3X2 =HD3X2 ,LD3X2 #;                            00344000
DEFINE CSJA =HCS[J,A],LCS[J,A]#,X1AJ =HX1A[J],LX1A[J]#,G1AJ =HG100345000
[A,J],LG1[A,J]#,X2AJ =HX2A[J],LX2A[J]#,G2AJ =HG2[A,J],LG2[A,J]# 00346000
,U1AJ =HU1A[J],LU1A[J]#,DELTA1AJ =HDELTA1A[J],LDELTA1A[J]#,U2AJ 00347000
=HU2A[J],LU2A[J]#,DELTA2AJ =HDELTA2A[J],LDELTA2A[J]#,O1AJ =HO1A 00348000
[J],LO1A[J]#,CSJA1 =HCS[J,A-1],LCS[J,A-1]#,O2AJ =HO2A[J],LO2A[J]00349000
#,THETA1AJ =HTHETA1A[J],LTHETA1A[J]#,ETA1AJ =HETA1A[J],LETA1A[J]00350000
#,THETA2AJ =HTHETA2A[J],LTHETA2A[J]#,ETA2AJ =HETA2A[J],LETA2A[J]00351000
#,DELTA1A0 =HDELTA1A[0],LDELTA1A[0]#,X1A0 =HX1A[0],LX1A[0]#,     00352000
DELTA2A0 =HDELTA2A[0],LDELTA2A[0]#,X2A0 =HX2A[0],LX2A[0]#,       00353000
THETA1A0 =HTHETA1A[0],LTHETA1A[0]#,U1A0 =HU1A[0],LU1A[0]#,       00354000
THETA2A0 =HTHETA2A[0],LTHETA2A[0]#,U2A0 =HU2A[0],LU2A[0]#,ETA1A000355000
=HETA1A[0],LETA1A[0]#,O1A0 =HO1A[0],LO1A[0]#,ETA2A0 =HETA2A[0],  00356000
LETA2A[0]#,O2A0 =HO2A[0],LO2A[0]#,TEMP =HTEMP ,LTEMP #;          00357000
DEFINE X =X1A0 #,U =U1A0 #,Y =X2A0 #,V =U2A0 #,X1 =X #,U1 =U #,  00358000
```

```
X2 =Y #,U2 =V #,THETA1 =THETA1A0 #,US1 =U2 #,THETA2 =THETA2A0 # 00359000
,US2 =-U1 #,D2X1 =THETA1 #,US1AJ =U2AJ #,D2X2 =THETA2 #,US2AJ = 00360000
-U1AJ #,THS1 =THETA2 #,NEXTUS1 =NEXTU2 #,THS2 =-THETA1 #,NEXTUS200361000
=-NEXTU1 #,01 =01A0 #,DELS1 =DELTA2 #,02 =02A0 #,DELS2 =-DELTA1 00362000
#,DELTA1 =DELTA1A0 #,DELS1AJ =DELTA2AJ #,DELTA2 =DELTA2A0 #,    00363000
DELS2AJ =-DELTA1AJ #,ETA1 =ETA1A0 #,D =DOUBLE #,ETA2 =ETA2A0 #; 00364000
DEFINE CALCABALPHBETAETC =                                      00365000
BEGIN                                                           00366000
  D(TEMPX1,MU,+,←,A1);                                          00367000
  D(TEMPX2,←,A2);                                               00368000
  D(TEMPX1,MP,-,←,B1);                                          00369000
  D(TEMPX2,←,B2);                                               00370000
  D(A1,A1,×,A2,A2,×,+,←,ALPHSQ);                                00371000
  D(B1,B1,×,B2,B2,×,+,←,BETASQ);                                00372000
  D(SQRT(HALPHSQ),0,←,ALPH);                                    00373000
  D(ALPHSQ,ALPH,/,ALPH,+,,2,/,←,ALPH);                          00374000
  D(SQRT(HBETASQ),0,←,BETA);                                    00375000
  D(BETASQ,BETA,/,BETA,+,,2,/,←,BETA);                          00376000
  D(A1,ALPH,/,←,AU1);                                           00377000
  D(A2,ALPH,/,←,AU2);                                           00378000
  D(B1,BETA,/,←,BU1);                                           00379000
  D(B2,BETA,/,←,BU2);                                           00380000
  D(ALPH,ALPHSQ,×,←,ALPHCUBE);                                  00381000
  D(BETA,BETASQ,×,←,BETACUBE);                                  00382000
  D(MP,ALPHCUBE,/,←,MPDIVALPHCUBE);                             00383000
  D(MU,BETACUBE,/,←,MUDIVBETACUBE);                             00384000
                                                                00385000
END #;                                                          00386000
FORMAT FMOUT1("STEP =",I5,"   TIME =",F8.2," SEC   P =",E10.2 );00387000
LIST LSTOUT1(STEPCOUNT,(TIME(2)-T2)/60,SQRT(HCAP1*2+HCAP2*2));  00388000
FORMAT FMOUT2("DT = ",5A6,"   X = ",5A6,"   Y = ",5A6);         00389000
FORMAT FMOUT3(" T = ",5A6,"   U = ",5A6,"   V = ",5A6,/);       00390000
ALPHA ARRAY ALEPH1,ALEPH2,ALEPH3[0:4];                          00391000
DEFINE PRINT =                                                  00392000
BEGIN                                                           00393000
  WRITE(FILEOUT,FMOUT1,LSTOUT1);                                00394000
  DPOP(DTP,ALEPH1);                                             00395000
  DPOP(X,ALEPH2);                                               00396000
  DPOP(Y,ALEPH3);                                               00397000
  WRITE(FILEOUT,FMOUT2,LSTOUT2);                                00398000
```

```
        DPOP(T,ALEPH1);                                                    00399000
        DPOP(U,ALEPH2);                                                    00400000
        DPOP(V,ALEPH3);                                                    00401000
        WRITE(FILEOUT,FMOUT3,LSTOUT2);                                     00402000
                                                                           00403000
      END #;                                                               00404000
      LIST LSTOUT2(FOR J :=0 STEP 1 UNTIL 4 DO ALEPH1[J],FOR J :=0         00405000
      STEP 1 UNTIL 4 DO ALEPH2[J],FOR J :=0 STEP 1 UNTIL 4 DO ALEPH3[J00406000
      ]);                                                                  00407000
      REAL HVINLAST,HDY,HVINNEW,HUHALVELAST,HDU,HDTIN,LVINLAST,LDY,        00408000
      LVINNEW,LUHALVELAST,LDU,LDTIN;                                       00409000
      DEFINE VINTHIS =VIN #,VINLAST =HVINLAST ,LVINLAST #,DY =HDY ,LDY00410000
      #,UHALVELAST =HUHALVELAST,LUHALVELAST #,DU =HDU ,LDU #,VINNEW        00411000
      =HVINNEW ,LVINNEW #,DTIN =HDTIN,LDTIN #;                             00412000
      LABEL L12;                                                           00413000
      FORMAT FMHALVE("PERIOD =",5A6,"    V NEW =",5A6,/,"DV =",E19.11,"00414000
DU ="E19.11,//);                                                           00415000
      LIST LSTHALVE(FOR J :=0 STEP 1 UNTIL 4 DO ALEPH1[J],FOR J :=0        00416000
      STEP 1 UNTIL 4 DO ALEPH2[J],HDY,HDU );                               00417000
      DEFINE SEARCHHALVEPOINT =                                            00418000
      BEGIN                                                                00419000
        IF ABS(HX2A[0])>GAM1/16 THEN HDT :=-HX2A[0]/HU2A[0]ELSE            00420000
        BEGIN                                                              00421000
          IF HVINLAST ≠0 THEN                                              00422000
          BEGIN                                                            00423000
            D(VINTHIS,VINLAST,-,←,DY);                                     00424000
            D(U,UHALVELAST,-,←,DU);                                        00425000
            D(-UHALVELAST,DY,×,DU,/,VINLAST,+,←,VINNEW);                   00426000
                                                                           00427000
          END ELSE D(VINTHIS,, 1,U,V,/,- ,×,←,VINNEW);                     00428000
          PRINT;                                                           00429000
          D(T,T,+,←,PERIOD);                                               00430000
          DPOP(PERIOD,ALEPH1);                                             00431000
          DPOP(VINNEW,ALEPH2);                                             00432000
          WRITE(FILEOUT,FMHALVE,LSTHALVE);                                 00433000
          D(U,←,UHALVELAST);                                               00434000
          IF ABS(HU1A[0])≤GAM2×16 THEN SEARCH :=" NO";                     00435000
          D(VINTHIS,←,VINLAST);                                            00436000
          D(VINNEW,←,VINTHIS);                                             00437000
          D(VINTHIS,←,V);                                                  00438000
```

```
      D(UIN,←,U);                                                          00439000
      D(YIN,←,Y);                                                          00440000
      D(XIN,←,X);                                                          00441000
      D(DTIN,←,DTP);                                                       00442000
      D(DTP,←,DT);                                                         00443000
      D(,0,←,T);                                                           00444000
      STEPCOUNT :=-1;                                                      00445000
      GO TO L12;                                                           00446000
                                                                           00447000
    END;                                                                   00448000
                                                                           00449000
  END #;                                                                   00450000
  COMMENT END OF DECLARATIONS BLOCK 2,START PRELIMINARY                    00451000
  CALCULATIONS;                                                            00452000
                                                                           00453000
  BEGIN                                                                    00454000
    INTEGER S,NU;                                                          00455000
    ARRAY FACT[0:SIGMA],HASIG,LASIG[0:SIGMA,0:3];                          00456000
    DEFINE ASIGNUA =HASIG[NU,A],LASIG[NU,A]#;                              00457000
    STEPCOUNT :=-1;                                                        00458000
    GMAXSQ :=GMAX*2;                                                       00459000
    D(,1,MU,=,←,MP);                                                       00460000
    D(,0,←,CAP1);                                                          00461000
    D(,0,←,CAP2);                                                          00462000
    D(,0,←,T);                                                             00463000
    D(DT,←,DTP);                                                           00464000
    D(XIN,←,X);                                                            00465000
    D(YIN,←,Y);                                                            00466000
    D(UIN,←,U);                                                            00467000
    D(VIN,←,V);                                                            00468000
    COMMENT HERE COMPUTE CS[J,A]ARRAY;                                     00469000
    S :=SIGMA;                                                             00470000
    FOR A :=3,2,1,0 DO                                                     00471000
    BEGIN                                                                  00472000
      NU :=0;                                                              00473000
      D(,1,←,ASIGNUA);                                                     00474000
      NU :=1;                                                              00475000
      D(S,0,A+2,0,/,←,ASIGNUA);                                            00476000
      NU :=2;                                                              00477000
      IF S≥2 THEN D(S×(2×S-(A+3)),0,(A+2)×(A+3)×2,0,/,←,ASIGNUA);          00478000
```

```
      NU :=3;                                                          00479000
      IF S≥3 THEN D(S×(3×S*2-3×S×(A+4)+(A+3)×(A+4)),0,(A+2)×(A+3)×00480000
      (A+4)×3,0,/,←,ASIGNUA);                                          00481000
      NU :=4;                                                          00482000
      IF S≥4 THEN D(S×(12×S*3-18×S*2×(A+5)+11×S×(A+4)×(A+5)-3×(A+300483000
      )×(A+4)×(A+5)),0,(A+2)×(A+3)×(A+4)×(A+5)×12,0,/,←,ASIGNUA); 00484000
                                                                       00485000
    END;                                                               00486000
    FACT[0]:=1;                                                        00487000
    FOR J :=1 STEP 1 UNTIL SIGMA DO FACT[J]:=J×FACT[J-1];              00488000
    FOR A :=3,2,1,0 DO FOR J :=0 STEP 1 UNTIL SIGMA DO                 00489000
    BEGIN                                                              00490000
      D(,0,←,TEMP);                                                    00491000
      FOR NU :=J STEP 1 UNTIL SIGMA DO D(FACT[NU]×(-1)*NU,0,FACT       00492000
      [NU-J],0,/,ASIGNUA,×,TEMP,+,←,TEMP);                             00493000
      D((-1)*J,0,TEMP,×,FACT[J],0,/,←,CSJA);                           00494000
                                                                       00495000
    END;                                                               00496000
                                                                       00497000
  END;                                                                 00498000
  L12:COMMENT CALCULATE D2X INITIAL;                                   00499000
                                                                       00500000
  BEGIN                                                                00501000
    D(X1,←,TEMPX1);                                                    00502000
    D(X2,←,TEMPX2);                                                    00503000
    CALCABALPHBETAETC;                                                 00504000
    D(X1,US1,+,US1,+,MPDIVALPHCUBE,A1,×,-,MUDIVBETACUBE,B1,×,-,←,      00505000
    D2X1);                                                             00506000
    D(X2,US2,+,US2,+,MPDIVALPHCUBE,A2,×,-,MUDIVBETACUBE,B2,×,-,←,      00507000
    D2X2);                                                             00508000
    NEXTSTEP:D(T,PERIOD,-,←,TEMP);                                     00509000
    IF (STEPCOUNT :=STEPCOUNT+1)MOD PRFLAG =0 OR HTEMP =0 THEN         00510000
    PRINT;                                                             00511000
    IF HTEMP =0 THEN                                                   00512000
    BEGIN                                                              00513000
      FORMAT FMEND(//,"DX =",E10.2,X10,"DY =",E10.2,X10,"DU =",        00514000
      E10.2,X10,"DV =",E10.2 /X24,"DT =",E10.2,X10,"DU =",E10.2);      00515000
      D(X,XIN,-,←,TEMPX1);                                             00516000
      D(Y,YIN,-,←,TEMPX2);                                             00517000
      D(U,UIN,-,←,TEMPU1);                                             00518000
```

```
      D(V,VIN,-,←,TEMPU2);                                                    00519000
      D(-Y,V,×,←,DT);                                                         00520000
      D(DT,THETA1,×,U,+,←,TEMP);                                              00521000
      WRITE(FILEOUT,FMEND,HTEMPX1,HTEMPX2,HTEMPU1,HTEMPU2,HDT,                00522000
      HTEMP);                                                                 00523000
      GO TO START;                                                            00524000
                                                                              00525000
    END;                                                                      00526000
    IF SEARCH ="YES"THEN                                                      00527000
    BEGIN                                                                     00528000
      D(ABS(HDT),LDT,U2,×,X2,+,KA,0,×,←,TEMPX2);                              00529000
      IF ABS(HT -HPERIOD/2)<.9 +ABS(HDT)AND HTEMPX2>0 THEN                    00530000
      SEARCHHALVEPOINT;                                                       00531000
                                                                              00532000
    END;                                                                      00533000
    D(T,DT,+,PERIOD,-,←,TEMP );                                               00534000
    IF HTEMP ≥0 THEN                                                          00535000
    BEGIN                                                                     00536000
      D(PERIOD,T,-,←,DT);                                                     00537000
      D(PERIOD,←,T);                                                          00538000
                                                                              00539000
    END ELSE D(T,DT,+,←,T);                                                   00540000
    COMMENT CALCULATE D3X;                                                    00541000
                                                                              00542000
    BEGIN                                                                     00543000
      D(AU1,U1,×,AU2,U2,×,+,,3,×,←,AUDOTU3);                                  00544000
      D(BU1,U1,×,BU2,U2,×,+,,3,×,←,BUDOTU3);                                  00545000
      D(U1,THS1,+,THS1,+,MPDIVALPHCUBE,U1,AUDOTU3,AU1,×,-,×,-,                00546000
      MUDIVBETACUBE,U1,BUDOTU3,BU1,×,-,×,-,←,D3X1);                           00547000
      D(U2,THS2,+,THS2,+,MPDIVALPHCUBE,U2,AUDOTU3,AU2,×,-,×,-,                00548000
      MUDIVBETACUBE,U2,BUDOTU3,BU2,×,-,×,-,←,D3X2);                           00549000
                                                                              00550000
    END;                                                                      00551000
                                                                              00552000
  END;                                                                        00553000
  TRYAGAIN:                                                                   00554000
  BEGIN                                                                       00555000
    COMMENT COMPUTE ELEMENTS OF PERTURBATION INTERRAL;                        00556000
    FOR J :=0 STEP 1 UNTIL SIGMA DO                                           00557000
    BEGIN                                                                     00558000
```

```
COMMENT PRELIMJ;                                                    00559000
D(J,0,DT,×,SIGMA,0,/,←,TAU);                                        00560000
D(TAU,,2,/,←,TAU2);                                                 00561000
D(TAU,,3,/,←,TAU3);                                                 00562000
IF J ≠0 THEN                                                        00563000
BEGIN                                                               00564000
  COMMENT CALCULATE XA[J]AND UA[J];                                 00565000
  D(D3X1,TAU3,×,D2X1,+,TAU2,×,U1,+,TAU,×,X1,+,←,X1AJ);              00566000
  D(D3X2,TAU3,×,D2X2,+,TAU2,×,U2,+,TAU,×,X2,+,←,X2AJ);              00567000
  D(D3X1,TAU2,×,D2X1,+,TAU ,×,U1,+,←,U1AJ);                         00568000
  D(D3X2,TAU2,×,D2X2,+,TAU ,×,U2,+,←,U2AJ);                         00569000
  COMMENT MORE PRELIMJ;                                             00570000
  D(X1AJ,←,TEMPX1);                                                 00571000
  D(X2AJ,←,TEMPX2);                                                 00572000
  CALCABALPHBETAETC;                                                00573000
  COMMENT CALCULATE THETAA[J];                                      00574000
  D(X1AJ,US1AJ,+,US1AJ,+,MPDIVALPHCUBE,A1,×,-,MUDIVBETACUBE         00575000
  ,B1,×,-,←,THETA1AJ);                                              00576000
  D(X2AJ,US2AJ,+,US2AJ,+,MPDIVALPHCUBE,A2,×,-,MUDIVBETACUBE         00577000
  ,B2,×,-,←,THETA2AJ);                                              00578000
                                                                    00579000
END;                                                                00580000
COMMENT CALCULATE OA[J];                                            00581000
D(D3X1,TAU,×,D2X1,+,←,O1AJ);                                        00582000
D(D3X2,TAU,×,D2X2,+,←,O2AJ);                                        00583000
COMMENT CALCULATE DELTAA[J];                                        00584000
D(THETA1AJ,O1AJ,-,←,DELTA1AJ);                                      00585000
D(THETA2AJ,O2AJ,-,←,DELTA2AJ);                                      00586000
IF J =0 THEN                                                        00587000
BEGIN                                                               00588000
  D(,0,←,ETA1);                                                     00589000
  D(,0,←,ETA2);                                                     00590000
                                                                    00591000
END ELSE                                                            00592000
BEGIN                                                               00593000
  COMMENT CALCULATE ETA[J];                                         00594000
  D(AU1,DELTA1AJ,×,AU2,DELTA2AJ,×,+,,3,×,←,AUDOTDELTA3);            00595000
  D(BU1,DELTA1AJ,×,BU2,DELTA2AJ,×,+,,3,×,←,BUDOTDELTA3);            00596000
  D(,-3,DELTA1AJ,×,MPDIVALPHCUBE,DELTA1AJ,AUDOTDELTA3,AU1,×,        00597000
  -,×,-,MUDIVBETACUBE,DELTA1AJ,BUDOTDELTA3,BU1,×,-,×,-,←,           00598000
```

```
      ETA1AJ);                                                          00599000
      D(,-3,DELTA2AJ,×,MPDIVALPHCUBE,DELTA2AJ,AUDOTDELTA3,AU2,×,        00600000
      -,×,-,MUDIVBETACUBE,DELTA2AJ,BUDOTDELTA3,BU2,×,-,×,-,←,           00601000
      ETA2AJ);                                                          00602000
                                                                        00603000
    END;                                                                00604000
                                                                        00605000
  END;                                                                  00606000
                                                                        00607000
  BEGIN                                                                 00608000
    COMMENT CALCULATE NEXT X,U AND THETA;                               00609000
    FOR J :=0 STEP 1 UNTIL SIGMA DO                                     00610000
    BEGIN                                                               00611000
      A :=1;                                                            00612000
      D(DELTA1AJ,←,G1AJ);                                               00613000
      D(DELTA2AJ,←,G2AJ);                                               00614000
      A :=2;                                                            00615000
      D(DELS1AJ,DELS1AJ,+,←,G1AJ);                                      00616000
      D(DELS2AJ,DELS2AJ,+,←,G2AJ);                                      00617000
      A :=3;                                                            00618000
      D(ETA1AJ,←,G1AJ);                                                 00619000
      D(ETA2AJ,←,G2AJ);                                                 00620000
                                                                        00621000
    END;                                                                00622000
    D(,0,←,SUMX1);                                                      00623000
    D(,0,←,SUMX2);                                                      00624000
    D(,0,←,SUMU1);                                                      00625000
    D(,0,←,SUMU2);                                                      00626000
    FOR A:=3,2,1 DO                                                     00627000
    BEGIN                                                               00628000
      D(,0,←,TEMPX1);                                                   00629000
      D(,0,←,TEMPX2);                                                   00630000
      D(,0,←,TEMPU1);                                                   00631000
      D(,0,←,TEMPU2);                                                   00632000
      FOR J :=0 STEP 1 UNTIL SIGMA DO                                   00633000
      BEGIN                                                             00634000
        D(G1AJ,CSJA,×,TEMPX1,+,←,TEMPX1);                               00635000
        D(G2AJ,CSJA,×,TEMPX2,+,←,TEMPX2);                               00636000
        D(G1AJ,CSJA1,×,TEMPU1,+,←,TEMPU1);                              00637000
        D(G2AJ,CSJA1,×,TEMPU2,+,←,TEMPU2);                              00638000
```

```
                                                                    00639000
      END;                                                          00640000
      D(SUMX1,TEMPX1,+,DT,×,A+1,0,/,←,SUMX1);                       00641000
      D(SUMX2,TEMPX2,+,DT,×,A+1,0,/,←,SUMX2);                       00642000
      D(SUMU1,TEMPU1,+,DT,×,A ,0,/,←,SUMU1);                        00643000
      D(SUMU2,TEMPU2,+,DT,×,A ,0,/,←,SUMU2);                        00644000
                                                                    00645000
    END;                                                            00646000
                                                                    00647000
    BEGIN                                                           00648000
      J :=SIGMA;                                                    00649000
      D(SUMX1,DT,×,X1AJ,+,←,NEXTX1);                                00650000
      D(SUMX2,DT,×,X2AJ,+,←,NEXTX2);                                00651000
      D(SUMU1,U1AJ,+,←,NEXTU1);                                     00652000
      D(SUMU2,U2AJ,+,←,NEXTU2);                                     00653000
      COMMENT CALCULATE NEXT THETA;                                 00654000
      D(NEXTX1,←,TEMPX1);                                           00655000
      D(NEXTX2,←,TEMPX2);                                           00656000
      CALCABALPHBETAETC;                                            00657000
      D(NEXTX1,NEXTUS1,+,NEXTUS1,+,MPDIVALPHCUBE,A1,×,-,            00658000
      MUDIVBETACUBE,B1,×,-,←,NEXTTHETA1);                           00659000
      D(NEXTX2,NEXTUS2,+,NEXTUS2,+,MPDIVALPHCUBE,A2,×,-,            00660000
      MUDIVBETACUBE,B2,×,-,←,NEXTTHETA2);                           00661000
                                                                    00662000
    END;                                                            00663000
                                                                    00664000
  END;                                                              00665000
                                                                    00666000
END;                                                                00667000
COMMENT CALCULATE CAP;                                              00668000
                                                                    00669000
BEGIN                                                               00670000
  FOR J :=0 STEP 1 UNTIL SIGMA DO                                   00671000
  BEGIN                                                             00672000
    A :=0;                                                          00673000
    D(DELS1AJ,DELS1AJ,+,←,G1AJ);                                    00674000
    D(DELS2AJ,DELS2AJ,+,←,G2AJ);                                    00675000
    A :=1;                                                          00676000
    D(ETA1AJ,←,G1AJ);                                               00677000
    D(ETA2AJ,←,G2AJ);                                               00678000
```

```
    END;                                                              00679000
    D(,0,←,SUMX1);                                                    00680000
    D(,0,←,SUMX2);                                                    00681000
    FOR A :=1,0 DO                                                    00682000
    BEGIN                                                             00683000
      D(,0,←,TEMPX1);                                                 00684000
      D(,0,←,TEMPX2);                                                 00685000
      FOR J :=0 STEP 1 UNTIL SIGMA DO                                 00686000
      BEGIN                                                           00687000
        D(G1AJ,CSJA,×,TEMPX1,+,←,TEMPX1);                             00688000
        D(G2AJ,CSJA,×,TEMPX2,+,←,TEMPX2);                             00689000
                                                                      00690000
      END;                                                            00691000
      D(SUMX1,TEMPX1,+,DT,×,A+1,0,/,←,SUMX1);                         00692000
      D(SUMX2,TEMPX2,+,DT,×,A+1,0,/,←,SUMX2);                         00693000
                                                                      00694000
    END;                                                              00695000
    J :=SIGMA;                                                        00696000
    D(NEXTTHETA1,THETA1AJ,-,SUMX1,-,←,CAP1);                          00697000
    D(NEXTTHETA2,THETA2AJ,-,SUMX2,-,←,CAP2);                          00698000
                                                                      00699000
  END;                                                                00700000
  CUMMENT CALCULATE P,Q,PSQ,QSQ;                                      00701000
                                                                      00702000
  BEGIN                                                               00703000
    Q1 :=HCAP1×HDT/4;                                                 00704000
    Q2 :=HCAP2×HDT/4;                                                 00705000
    P1 :=Q1×HDT/5;                                                    00706000
    P2 :=Q2×HDT/5;                                                    00707000
    PSQ :=P1*2 +P2*2;                                                 00708000
    QSQ :=Q1*2 +Q2*2;                                                 00709000
                                                                      00710000
  END;                                                                00711000
  D(T,DT,-,←,TEMP);                                                   00712000
  IF HTEMP =0 THEN                                                    00713000
  BEGIN                                                               00714000
    IF HCAP1*2 +HCAP2*2 >GMAXSQ THEN                                  00715000
    BEGIN                                                             00716000
      INTEGER K;                                                      00717000
                                                                      00718000
```

```
      D(DT,,2,/,,←,DT);                                                    00719000
      D(DT,←,T);                                                           00720000
      D(X1,←,TEMPX1);                                                      00721000
      D(X2,←,TEMPX2);                                                      00722000
      CALCABALPHBETAETC;                                                   00723000
      GO TO TRYAGAIN;                                                      00724000
                                                                           00725000
    END ELSE                                                               00726000
    BEGIN                                                                  00727000
      GAM1SQ :=PSQ;                                                        00728000
      GAM2SQ :=QSQ;                                                        00729000
      GAM1 :=SQRT(GAM1SQ);                                                 00730000
      GAM2 :=SQRT(GAM2SQ);                                                 00731000
      GAM :=(GAM1 +GAM2)/2;                                                00732000
      GAMSQ :=GAM*2;                                                       00733000
      D(DT,←,DTIN);                                                        00734000
                                                                           00735000
    END;                                                                   00736000
                                                                           00737000
  END;                                                                     00738000
  D(DT,←,DTP);                                                             00739000
  IF PSQ≤GAM1SQ AND QSQ≤GAM2SQ AND (PSQ≥GAMSQ OR QSQ≥GAMSQ)THEN            00740000
  HDT :=HDT ELSE                                                           00741000
  BEGIN                                                                    00742000
    IF PSQ≥QSQ THEN                                                        00743000
    BEGIN                                                                  00744000
      GAMK :=GAM1;                                                         00745000
      R :=SQRT(PSQ);                                                       00746000
                                                                           00747000
    END ELSE                                                               00748000
    BEGIN                                                                  00749000
      GAMK:=GAM2;                                                          00750000
      R :=SQRT(QSQ);                                                       00751000
                                                                           00752000
    END;                                                                   00753000
    HDT :=HDT/4×(3+(GAM+GAMK)/(GAM+R));                                    00754000
                                                                           00755000
  END;                                                                     00756000
  COMMENT MOVE X,U AND THETA;                                              00757000
                                                                           00758000
```

```
        BEGIN                                                          00759000
          D(NEXTX1,←,X1);                                              00760000
          D(NEXTX2,←,X2);                                              00761000
          D(NEXTU1,←,U1);                                              00762000
          D(NEXTU2,←,U2);                                              00763000
          D(NEXTTHETA1,←,THETA1);                                      00764000
          D(NEXTTHETA2,←,THETA2);                                      00765000
                                                                       00766000
        END;                                                           00767000
        GO TO NEXTSTEP;                                                00768000
                                                                       00769000
      END;                                                             00770000
      FINISH;                                                          00771000
    END;                                                               00772000
    LAST:                                                              00773000
  END.                                                                 00774000
30          END-OF-FILE CARD
LABEL    00000000FILEIN 0010000001
```

B. Cowell's Method

1. General

The term "Cowell's Method" as it applies here refers to a class of multi-step algorithms derived from Bessel's central difference interpolation polynomial and applied to the numerical solution of ordinary differential systems. The method originates with Cowell and Crommelin [9] and is described in some detail by Herrick [10]. These versions, as such, shall be referred to in this paper as the "conventional" Cowell method. A variation of the conventional method, discussed in [11] forms the basis for the "modified" Cowell method wherein the necessity for the continued maintenance of a difference table as the integration progresses is eliminated. An amplification of these results is the intent of the treatment to be presented here.

In broad plan, two general algorithms will be developed, one applicable to the special first order equation $\dot{y} = f(t,y)$, the other to the special second order equation $\ddot{y} = f(t,y)$. Here the "dot" signifies differentiation with respect to t. It will then be shown how a combination of these results can be applied to the general second order equation $\ddot{y} = f(t,y,\dot{y})$. The extension of the technique to include systems of equations follows naturally.

At the outset it is assumed that a table of differences for the function $f(t,y(t))$ has been furnished by some starting procedure. The principal technique of the modified Cowell method then consists in manipulating the difference table in such a way that a general algorithm (derived from Bessel's formula) is made to yield a set of predictor-corrector formulas suitable for extending the tabulation. These formulas are expressed as linear combinations of the current and previously computed values of $f(t,y(t))$ in which the coefficients are permanent. That is, the coefficients depend only

upon the order of differences used and can be computed once and for all prior to an application.

2. The First Order Case

2.1. The General Algorithm

Let $f(t,y(t))$ be tabulated over an equipartition of step size h by means of some starting procedure which produces the 2M+1 points

$$\left\{ t_{n-i},\ f_{n-i} \right\} \quad , \quad i = 0,\ 1,\ 2,\ \ldots\ ,\ 2M \quad ,$$

where

$$\begin{cases} t_k = t_0 + kh \\ f_k = f(t_k, y_k) \end{cases} .$$

Under these circumstances and with the introduction of the change of variable

$$t = hs + t_n \ , \quad s = (t-t_n)/h \quad ,$$

Bessel's interpolation polynomial of degree 2M for $f(t,y(t))$ takes the form

$$\begin{aligned} I_{2M}(x) &= I_{2M}(hs + x_n) \quad , \\ &= \sum_{i=0}^{M} \left\{ \tfrac{1}{2} \left(\Delta^{2i} f_{n-i+1} + \Delta^{2i} f_{n-i} \right) B_{2i}(s) \right. \qquad (2.1\text{-}1) \\ &\quad \left. + \Delta^{2i+1} f_{n-i}\, B_{2i+1}(s) \right\} \quad , \end{aligned}$$

where Δ denotes the conventional forward difference operator, and the polynomial coefficients $B_k(s)$ are given by

$$B_0(s) = 1 \quad , \tag{2.1-2a}$$

$$B_{2i}(s) = \frac{1}{(2i)!} \; s\,(s-1)(s+1)(s-2)(s+2) \;..$$

$$\ldots\,(s-i)(s+i-1) \;.. \tag{2.1-2b}$$

$$B_{2i+1}(s) = \frac{1}{(2i+1)} \; (s - \tfrac{1}{2})\, B_{2i}(s) \quad . \tag{2.1-2c}$$

It will be noted for later convenience that the polynomials $B_{2i+1}(s)$ are symmetric with respect to the point $s = \frac{1}{2}$ over the interval $0 \leq s \leq 1$.

The interpolation polynomial (2.1-1) can be put in a more convenient operational form by introducing the displacement operator E, the central difference operator δ, and the averaging operator μ defined as follows:

$$Eg(x) = g(x+h) \quad , \tag{2.1-3a}$$

$$\delta g(x) = (E^{\frac{1}{2}} - E^{-\frac{1}{2}})\, g(x) = g(x+h/2) - g(x-h/2) \quad , \tag{2.1-3b}$$

$$\mu g(x) = \tfrac{1}{2}\,(E^{\frac{1}{2}} + E^{-\frac{1}{2}})\, g(x) = \tfrac{1}{2}\left\{g(x+h/2) + g(x-h/2)\right\} . \tag{2.1-ec}$$

It can then be shown that the following relations hold:

$$\Delta = E - 1 = E\nabla = \nabla E \quad , \tag{2.1-4a}$$

$$\nabla = 1 - E^{-1} = E^{-1}\Delta = \Delta E^{-1} \quad , \tag{2.1-4b}$$

$$\delta = E^{\frac{1}{2}} - E^{-\frac{1}{2}} \quad , \tag{2.1-4c}$$

$$\mu = \tfrac{1}{2}\,(E^{\frac{1}{2}} + E^{-\frac{1}{2}}) \quad , \tag{2.1-4d}$$

$$\mu\delta = \tfrac{1}{2}\left(E - E^{-1}\right) = \tfrac{1}{2}\left(\Delta + \nabla\right) \quad , \tag{2.1-4e}$$

$$= \tfrac{1}{2}\left(\Delta + \Delta E^{-1}\right) \quad , \tag{2.1-4f}$$

$$= \tfrac{1}{2}\left(\nabla E + \nabla\right) \quad . \tag{2.1-4g}$$

Formal manipulation of these operators now yields the set of conversion formulas

$$\Delta f_j = f_{j+1} - f_j = \nabla f_{j+1} \quad , \tag{2.1-5a}$$

$$\nabla f_j = f_j - f_{j-1} = \Delta f_{j-1} \quad , \tag{2.1-5b}$$

$$\delta f_j = f_{j+\frac{1}{2}} - f_{j-\frac{1}{2}} \quad , \tag{2.1-5c}$$

$$\mu f_j = \tfrac{1}{2}\left(f_{j+\frac{1}{2}} + f_{j-\frac{1}{2}}\right) \quad , \tag{2.1-5d}$$

$$\mu\delta f_j = \tfrac{1}{2}\left(\Delta f_j + \Delta f_{j-1}\right) \quad , \tag{2.1-5e}$$

$$= \tfrac{1}{2}\left(\nabla f_{j+1} + \nabla f_j\right) \quad , \tag{2.1-5f}$$

$$\delta^k f_j = \Delta^k f_{j-k/2} = \nabla^k f_{j+k/2} \quad , \tag{2.1-5g}$$

$$\mu\delta^k f_j = \tfrac{1}{2}\left\{\Delta^k f_{j-(k-1)/2} + \Delta^k f_{j-(k+1)/2}\right\} \quad , \tag{2.1-5h}$$

$$= \tfrac{1}{2}\left\{\nabla^k f_{j+(k+1)/2} + \nabla^k f_{j+(k-1)/2}\right\} \quad , \tag{2.1-5i}$$

$$\Delta^k f_j = \delta^k f_{j+k/2} = \nabla^k f_{j+k} \quad , \tag{2.1-5j}$$

$$\tfrac{1}{2}\left\{\Delta^k f_j + \Delta^k f_{j-1}\right\} = \mu^k f_{j+(k-1)/2} \quad . \tag{2.1-5k}$$

Setting $k = 2i + 1$ and $j = n - i$ in (2.1-5j) gives

$$\Delta^{2i+1} f_{n-i} = \delta^{2i+1} f_{n+1/2} \quad . \tag{2.1-6a}$$

Similarly, setting $k = 2i$ and $j = n - i + 1$ in (2.1-5k) gives

$$\tfrac{1}{2} \left(\Delta^{2i} f_{n-i+1} + \Delta^{2i} f_{n-i}\right) = \mu\delta^{2i} f_{n+1/2} \quad . \tag{2.1-6b}$$

Hence, substituting these last two results into (2.1-1) yields the operational form of Bessel's formula

$$I_{2M}(hs + t_n) = \sum_{i=0}^{M} \left\{ B_{2i}(s)\, \mu\delta^{2i} + B_{2i+1}(s)\delta^{2i+1} \right\} f_{n+1/2} \quad . \tag{2.1-7}$$

Now, from the differential equation one can write

$$\int_{t_n}^{t_{n+1}} y'(t)\, dx = \int_{t_n}^{t_{n+1}} f(t, y(t))\, dx \quad . \tag{2.1-8}$$

Replacing $f(t,y(t))$ by the interpolating polynomial (2.1-7) in this expression then yields the approximation

$$y_{n+1} - y_n \approx h \int_0^1 I_{2M}(hs + t_n)\, ds \quad , \tag{2.1-9}$$

which can be written in the form

$$\delta y_{n+1/2} \approx h \sum_{i=0}^{M} \left(\gamma_{2i}\, \mu\delta^{2i} + \gamma_{2i+1}\, \delta^{2i+1}\right) f_{n+1/2} \quad , \tag{2.1-10}$$

where

$$\gamma_{2i} = \int_0^1 B_{2i}(s)\, ds \quad ,$$

$$\gamma_{2i+1} = \int_0^1 B_{2i+1}(s)\, ds \quad . \tag{2.1-11}$$

But, from the aforementioned symmetry of the polynomials $B_{2i+1}(s)$ it follows that $\gamma_{2i+1} = 0$ for all values of i. Hence (2.1-10) reduces to

$$\delta y_{n+1/2} \approx \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i} \right\} h f_{n+1/2} \quad . \tag{2.1-12}$$

Replacing n by n - 1/2 and formally multiplying (2.1-12) by the inverse operator δ^{-1} yields the final form of the general algorithm for the first order case, namely

$$y_n \approx \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i-1} \right\} h f_n \quad . \tag{2.1-13}$$

As a formal approximate solution of the differential equation, this can be written

$$y \approx \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i-1} \right\} h y' \quad . \tag{2.1-14}$$

It will be noted that (2.1-13) is an implicit (closed type) formula since the sought-for value $y_n = y(t_n)$ also appears in the function $f_n = f(t_n, y_n)$ on the right side of the equation.

2.2. Evaluation of the Coefficients

A simple way to evaluate the coefficients γ_{2i} defined by (2.1-11) can be developed by writing (2.1-2b) in the form

$$(2i)!\, B_{2i}(s) = s(s-1)(s+1)\ldots(s-i)(s+i-1) \quad , \qquad (2.2\text{-}1a)$$

$$= \sum_{j=0}^{2i-1} A_{2i,j}\, s^{2i-j}, \quad i \geq 1 \quad . \qquad (2.2\text{-}1b)$$

It will be noted from (2.2-1a) that the leading coefficient $A_{2i,0}$ is unity for all i. Also, it can be shown that the sum of the roots of the polynomial (2.2-1a) is i. Hence, from elementary theory of equations it follows that $A_{2i,0} = 1$ and $A_{2i,1} = -\,i$ for all i. A recursion formula which affords an easy construction of the integer coefficients $A_{2i,j}$ can be derived by replacing i by i+1 in (2.2-1a) to get

$$(2i+2)!\, B_{2i+2}(s) = s(s-1)(s+1)\ldots(s-i-1)(s+i) \quad ,$$

$$= (s-i-1)(s+i) \sum_{j=0}^{2i-1} A_{2i,j}\, s^{2i-j} \quad . \qquad (2.2\text{-}2a)$$

On the other hand, replacing i by i+1 in (2.2-1b) gives

$$(2i+2)!\, B_{2i+2}(s) = \sum_{j=0}^{2i+1} A_{2i+2,j}\, s^{2i+2-j} \quad . \qquad (2.2\text{-}2b)$$

From these relations it follows that

$$\sum_{j=0}^{2i+1} A_{2i+2,j}\, s^{2i+2-j} \equiv (s-i-1)(s+i) \sum_{j=0}^{2i-1} A_{2i,j}\, s^{2i-j} \quad . \qquad (2.2\text{-}3)$$

Equating coefficients in this expression then yields the set of recursion formulas

$$A_{2i,0} = 1, \quad A_{2i,1} = -\,i \quad i = 0,1,2,\ldots \quad , \qquad (2.2\text{-}4a)$$

$$A_{2i,j} = A_{2i-2,j} - A_{2i-2,j-1} - i\,(i-1)\,A_{2i-2,j-2}$$

$$i = 2, 3, \ldots ; \; j = 2, 3, \ldots , 2i - 1 . \qquad (2.2\text{-}4b)$$

It will be noted from these relations that after using (2.2-4a) to fill the first two columns of the matrix

$$A = (A_{2i,j}) \; i = 1, 2, 3, \ldots , M; \; j = 0,1,2,\ldots,2i-1 , \qquad (2.2\text{-}5)$$

each row after the first generates its successor in accordance with (2.2-4b). A convenient check on the process is the relation

$$\sum_{j=0}^{2i-1} A_{2i,j} = 0 , \qquad (2.2\text{-}6)$$

that is, the sum of the entries in each row after the first must vanish. Having constructed the matrix A, it follows immediately from (2.1-2a), (2.2-1b), and the definition (2.1-11) that

$$\gamma_0 = 1 ,$$

$$\gamma_{2i} = \frac{1}{(2i)!} \sum_{j=0}^{2i-1} \frac{1}{(2i-j+1)} A_{2i,j}, \; i \geq 1 . \qquad (2.2\text{-}7)$$

A few values of the coefficients γ_{2i} are given in a rational form in Table I.

TABLE I

THE γ COEFFICIENTS (RATIONAL FORM)

2i	γ_{2i}
0	1
2	-1/12
4	11/720
6	-191/60480
8	2497/3628800
10	-14797/95800320
12	92427157/2615348736000
14	-257184391/31384184832000
16	61430943169/32011868528640000
18	-23133945892303/5109094216770944000

3. The Second Order Case

3.1. The General Algorithm

By purely formal manipulations the result (2.1-14) for the first order case can be made to yield a general algorithm applicable to the special second order equation. To do this y can be replaced by $\dot{y}$ in (2.1-14) to get

$$\dot{y} = \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i-1} \right\} h\ddot{y} \quad . \tag{3.1-1}$$

Substitution of this expression into the right member of (2.1-14) then gives

$$y = \mu^2 \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i-1} \right\}^2 h^2 \ddot{y} \quad . \tag{3.1-2}$$

It is easy to show by (2.1-4c,d) that

$$\mu^2 = 1 + \frac{\delta^2}{4} \quad . \tag{3.1-3}$$

Hence, putting this result into (3.1-2) leads to

$$y = \left\{ \left(\sum_{i=0}^{M} \gamma_{2i} \delta^{2i-1} \right)^2 + \frac{1}{4} \left(\sum_{i=0}^{M} \gamma_{2i} \delta^{2i} \right)^2 \right\} h^2 \ddot{y} \quad . \tag{3.1-4}$$

Formally expanding the squared operators in this expression, collecting terms, and truncating the result to include differences of order 2M yields the formal approximate solution of the special second order equation

$$y = \left\{ \sum_{i=0}^{M+1} \gamma^*_{2i} \delta^{2i-2} \right\} h^2 \ddot{y} \quad , \tag{3.1-5}$$

which, when written as an algorithm, becomes

$$y_n = \left\{ \sum_{i=0}^{M+1} \gamma^*_{2i} \delta^{2i-2} \right\} h^2 f_n \quad , \tag{3.1-6}$$

or

$$\delta^2 y_n = \left\{ \sum_{i=0}^{M} \gamma_{2i}{}^* \delta^{2i} \right\} h^2 f_n \quad , \tag{3.1-7}$$

where

$$\delta^2 y_n = y_{n+1} - 2y_n + y_{n-1} \quad . \tag{3.1-8}$$

3.2. Evaluation of the Coefficients

The coefficients γ^* appearing in the general algorithm (3.1-6) can be evaluated in terms of the γ's by the brute strength expedient of expanding (3.1-4) and collecting terms according to powers of δ. The result is collected below in the set of relations:

$$\gamma_0^* = 1 \ , \tag{3.2-1a}$$

$$\begin{cases} \gamma_{2i}^* = 2 \sum_{j=0}^{(i-1)/2} \gamma_{2j}\gamma_{2(i-j)} + \frac{1}{2} \sum_{j=0}^{(i-3)/2} \gamma_{2j}\gamma_{2(i-j-1)} + \frac{1}{4} \gamma_{i-1}^2 \\ \\ i = 1, 3, 5, 7, \ldots \end{cases} \tag{3.2-1b}$$

$$\begin{cases} \gamma_{2i}^* = 2 \sum_{j=0}^{(i-2)/2} \gamma_{2j}\gamma_{2(i-j)} + \frac{1}{2} \sum_{j=0}^{(i-2)/2} \gamma_{2j}\gamma_{2(i-j-1)} + \gamma_i^2 \\ \\ i = 2, 4, 6, 8, \ldots \ . \end{cases} \tag{3.2-1c}$$

A few values of the coefficients γ_{2i}^* are given in rational form in Table II. Decimal equivalents to high precision for both γ_{2i} and γ_{2i}^* are given in Table III.

TABLE II

THE γ^* COEFFICIENTS (RATIONAL FORM)

2i	γ_{2i}^*
0	1
2	1/12
4	-1/240
6	31/60480
8	-289/3628800
10	317/22809600
12	-6803477/2615348736000
14	69429517/56491532697600

TABLE 3

GAMMA COEFFICIENTS

(HIGH PRECISION)

2I	GAMMA(2I)	GAMMA*(2I)
0	9.99999999999999999999999@-01	9.99999999999999999999999@-01
2	-8.33333333333333333333342@-02	8.33333333333333333333327@-02
4	1.52777777777777777777772@-02	-4.16666666666666666666669@-03
6	-3.15806878306878306878303@-03	5.12566137566137566137735@-04
8	6.88106261022927689594395@-04	-7.96406525573192239858255@-05
10	-1.54456686574742130297685@-04	1.38976571268237934904748@-05
12	3.53402801422808036564647@-05	-2.60136512823351447690096@-06
14	-8.19471081937324221274833@-06	5.10400225009737796334099@-07
16	1.91900523126413849215314@-06	-1.03590601155729637677948@-07
18	-4.52799359513728980482266@-07	2.15681866768386995860797@-08
20	1.07483062036975514183510@-07	-4.58106089075435731645171@-09

4. The Starting Table of Differences

Both the conventional and modified versions of Cowell's method require a starting table of differences. It is assumed that the order of differences to be used is 2M, where M is a positive integer, and that values for the quantities

$$\left\{ t_{M-i},\ y_{M-i},\ f_{M-i} \right\} , \quad i = 0,\ 1,\ 2,\ \ldots ,\ 2M \ , \tag{4-1}$$

$$\delta^{-2} f \ , \ \delta^{-1} f_{\frac{1}{2}}, \ \ \delta^{-1} f_{-\frac{1}{2}} \ , \tag{4-2}$$

have been furnished by means of some starting procedure (to be discussed later) based on a given set of initial conditions for the differential system. From these values a table of differences is then constructed out to and including differences of order 2M and back to and including inverse differences of order -2 of the function f. An illustration of such a table written in central difference notation is shown for the case 2M = 4 in the bounded region of Figure 5. Within this region the starting values are further separated by dashed lines.

Having given the above starting table of differences, the principal distinguishing feature of Cowell's method is the extension of the table using estimated differences obtained under the assumption that differences of order 2M are constant. The extension is carried out down to and including all differences associated with the entry f_{M+1} by reversing the normal table construction procedure. The results of this process for the case 2M = 4 is shown as the unbounded region of Figure 5. In the discussion to follow, this portion of the figure shall be referred to as the estimated table of differences.

It should be kept in mind that although central difference notation is used in the illustration, the entries represent nothing more than forward or

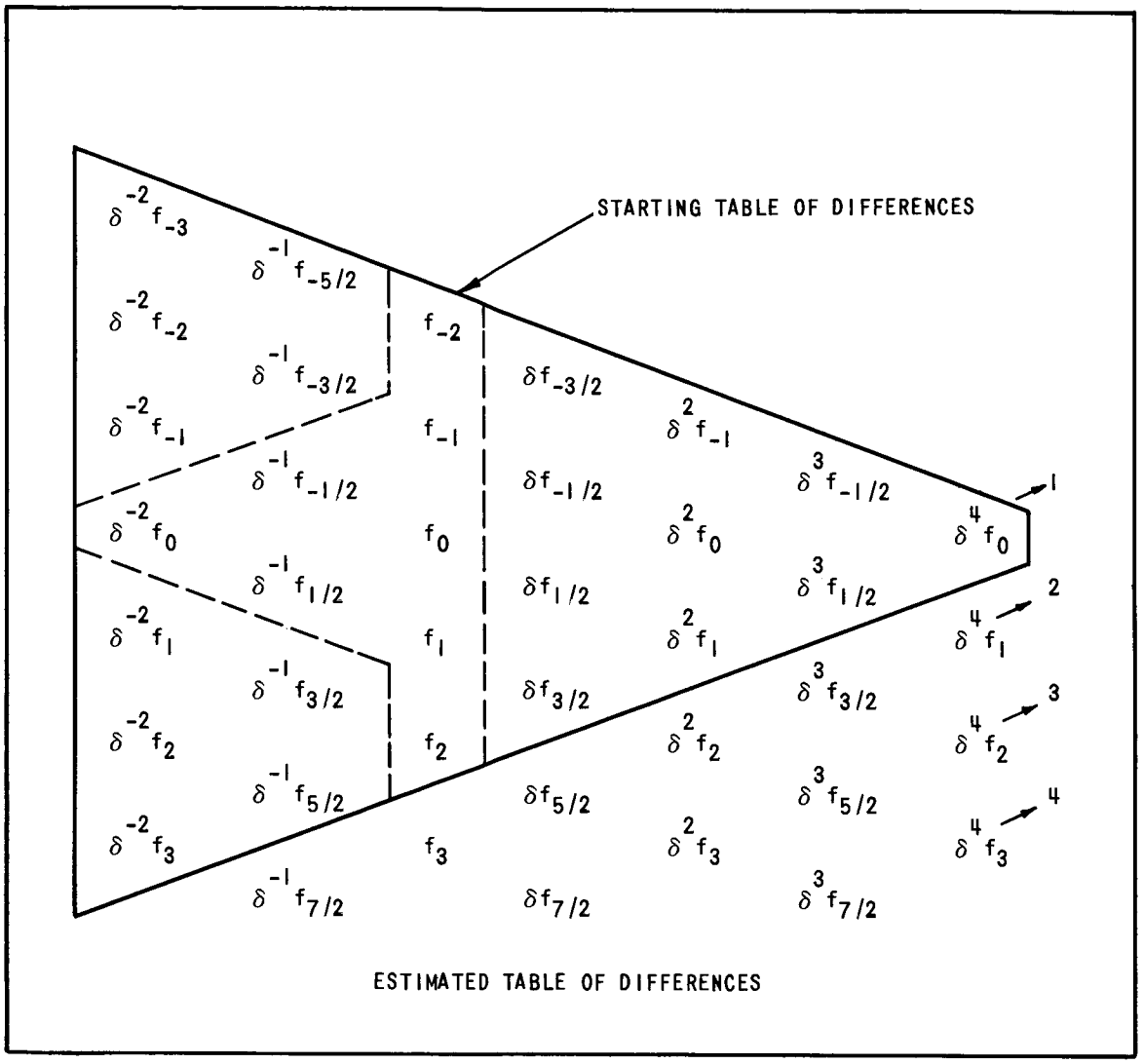

Figure 5. The Starting Table of Differences for the Cowell Method.

backward differences (depending on viewpoint) based on the tabulated values of the function f. In an actual numerical case the numbers are the same regardless of viewpoint.

4.1. The modified Cowell Method

In the modified (or expanded) version of Cowell's method the necessity for maintaining a current difference table during the integration is eliminated by rewriting the central differences appearing in the general algorithms (2.1-13) and (3.1-6) in terms of the tabulated values of the function f. The results of such a process offer particular time saving advantages in applications calling for frequent changes in stepsize. The breakdown is performed in two ways. The first expansion produces predictor formulas which give y_n entirely in terms of previously computed information. The second expansion produces corrector formulas which give y_n in terms of the current value $f_n = f(t_n, y_n)$ as well as previously computed values. The results of such a process are summarized in the form of general expanded algorithms in the sections to follow. Specific illustrations of the breakdown process are given in Section 7.

5.1. Predictor-Corrector Formulas for the First Order Case

Under the first expansion the general algorithm (2.1-13) takes the matrix form

$$y_n = \left\{ \sum_{i=0}^{M} \gamma_{2i} \mu \delta^{2i-1} \right\} h f_n = h \left(\gamma_0 \delta^{-1} f_{n-\frac{1}{2}} + Pf \right) , \qquad (5.1\text{-}1)$$

where f is the column vector of tabulated values

$$f = \left\{ f_{n-1}, f_{n-2}, \cdots, f_{n-2M-1} \right\} , \qquad (5.1\text{-}2)$$

and P is the row vector of constant coefficients defined by the matrix equation

$$P = \tfrac{1}{2}\,\gamma DB = (P_1,\ P_2,\ P_3,\ \cdots,\ P_{2M+1}) \quad . \tag{5.1-3}$$

In this last expression γ denotes the row vector

$$\gamma = (\gamma_0,\ \gamma_2,\ \gamma_4,\ \cdots,\ \gamma_{2M}) \quad , \tag{5.1-4}$$

B denotes the square matrix of signed binomial coefficients

$$B\left[B_{i,j}\right] = \left[(-1)^j \binom{i}{j}\right] \quad i,\ j = 0,\ 1,\ 2,\ \cdots,\ 2M \tag{5.1-5}$$

and D represents a rectangular matrix with integer entries defined as follows:

$$D = [D_{i,j}] \quad i = 0,\ 1,\ 2,\ \cdots,\ M;\ j = 0,\ 1,\ 2,\ \cdots,\ 2M \tag{5.1-6}$$

$$D_{0,j} = 1,\ j = 0,\ 1,\ 2,\ \cdots,\ 2M \quad , \tag{5.1-7a}$$

$$D_{1,j} = \begin{cases} 0, & j = 0 \\ j + 1, & j = 1,\ 2,\ 3,\ \cdots,\ 2M \end{cases} \quad , \tag{5.1-7b}$$

$$D_{i,j} = 0 \quad i = 2,\ 3,\ 4,\ \cdots,\ M;\ j = 0,\ 1,\ 2,\ \cdots,\ 2i - 2 \quad , \tag{5.1-7c}$$

$$D_{i,j} = \frac{(j+1)}{i!}\,(j-i)(j-i-1)(j-i-2)..(j-2i+2) \quad ,$$

$$i = 2,\ 3,\ 4,\ \cdots,\ M;\ j = 2i-1,\ 2i,\ 2i+1,\ \cdots,\ 2M \quad . \tag{5.1-7d}$$

In expanded form (5.1-1) becomes the predictor formula

$$y_n = h \left\{ \gamma_0 \delta^{-1} f_{n-\frac{1}{2}} + \sum_{j=1}^{2M+1} P_j f_{n-j} \right\} , \qquad (5.1\text{-}8)$$

in which the coefficients P_j are defined by (5.1-3). In the subsequent discussion this expression shall be referred to as the general predictor formula for the first order case. A detailed illustration of the derivation of (5.1-8) for the case $2M = 4$ is given in section 7.

Under the second expansion process (5.1-13) takes the matrix form

$$y_n = h \left(\gamma_0 \delta^{-1} f_{n-\frac{1}{2}} + Cf \right) , \qquad (5.1\text{-}9)$$

where f is the column vector

$$f = (f_n, f_{n-1}, f_{n-2}, \ldots, f_{n-2M}) , \qquad (5.1\text{-}10)$$

and C is the row vector of constants defined by

$$C = \tfrac{1}{2} \gamma E B = (C_1, C_2, C_3, \ldots, C_{2M+1}) . \qquad (5.1\text{-}11)$$

In this expression γ and B are the same as previously defined and E denotes a rectangular matrix with integer entries given as follows:

$$E = [E_{i,j}] \quad i = 0, 1, 2, \ldots, M; \; j = 0, 1. 2, \ldots, 2M , \qquad (5.1\text{-}12)$$

$$E_{0,j} = \begin{cases} 1, & j = 0 \\ 0, & j = 1,2,3,\ldots,2M \end{cases} , \qquad (5.1\text{-}13a)$$

$$E_{1,j} = \begin{cases} 0, & j = 0 \\ 2, & j = 1 \\ 1, & j = 2,3,4,\ldots,2M \end{cases} , \qquad (5.1\text{-}13b)$$

$$E_{2,j} = \begin{cases} 0, & j = 0,1,2 \\ j-1, & j = 3,4,5,\ldots,2M \end{cases} , \qquad (5.1\text{-}13c)$$

$$E_{i,j} = 0 \quad i = 3,4,5,\ldots,M;\ j = 0,1,2,\ldots,2i-2 \ , \tag{5.1-13d}$$

$$E_{i,j} = \frac{(j-1)}{(i-1)!}\ (j-i-1)(j-i-2)\ldots(j-2i+2)$$

$$i = 3,4,5,\ldots M;\ j = 2i-1,\ 2i,\ 2i+1,\ldots 2M \tag{5.1-13e}$$

In expanded form (5.1-9) becomes the corrector formula

$$y_n = h\left\{\gamma_0\delta^{-1}f_{n-\frac{1}{2}} + \sum_{j=1}^{2M+1} C_j f_{n-j+1}\right\}, \tag{5.1-14}$$

where the coefficients C_j are calculated from (5.1-11). This is the general corrector formula for the first order case (see Section 7).

5.2. Predictor-Corrector Formulas for the Second Order Case

Under the first expansion the general algorithm (3.1-6) becomes

$$y_n = \left\{\sum_{i=0}^{M+1} \gamma_{2i}^{\ *}\,\delta^{2i-2}\right\} h^2 f_n = h^2(\gamma_0^{\ *}\delta^{-2}f_n + P^* f) \ , \tag{5.2-1}$$

in which

$$f = \left\{f_{n-1},\ f_{n-2},\ f_{n-3},\cdots,f_{n-2M-1}\right\}, \tag{5.2-2}$$

$$P^* = \gamma^* D^* B = (P_1^*,\ P_2^*,\ P_3^*,\ldots,P_{2M+1}^*) \tag{5.2-3}$$

$$\gamma^* = (\gamma_2^*,\ \gamma_4^*,\ \gamma_6^*,\ldots,\gamma_{2M+2}^*) \ , \tag{5.2-4}$$

and D^* is a rectangular array with integer entries defined as follows:

$$D^* = [D_{i,j}^*] \quad i = 0,1,2,\ldots,M;\ j = 0,1,2,\ldots,2M \tag{5.2-5a}$$

$$D^*_{0,j} = 1, \quad j = 0,1,2,\ldots,2M \quad , \tag{5.2-5b}$$

$$D^*_{1,j} = \begin{cases} 0, & j = 0,1 \\ j-1, & j = 2,3,\ldots,2M \end{cases} \quad , \tag{5.2-5c}$$

$$D^*_{i,j} = 0 \quad i = 2,3,\ldots,M; \; j = 0,1,2,\ldots,2i-1 \quad , \tag{5.2-5d}$$

$$D^*_{i,j} = \frac{(j-i)}{i!} \; (j-i-1)(j-i-2)..(j-2i+1)$$

$$i = 2,3,\ldots,M; \; j = 2i, \; 2i+1,\ldots,2M \quad . \tag{5.2.5e}$$

In expanded form (5.2-1) becomes the predictor formula

$$y_n = h^2 \left\{ \gamma_0^* \delta^{-2} f_n \quad + \sum_{j=1}^{2M+1} P^*_j f_{n-j} \right\} , \tag{5.2-6}$$

where the coefficients P^*_j are calculated from (5.2-3). This is the general predictor formula for the second order case (see Section 7.3).

Under the second expansion (3.1-6) takes the matrix form

$$y_n = h^2 \left\{ \gamma_0^* \delta^{-2} f_n + C^* f \right\} , \tag{5.2-7}$$

where

$$f = \left\{ f_n, f_{n-1}, f_{n-2}, \ldots, f_{n-2M} \right\} , \tag{5.2-8}$$

$$C^* = \gamma^* E^* B = (C_1^*, \; C_2^*, \; C_3^*, \ldots, \; C_{2M+1}^*) \quad , \tag{5.2-9}$$

and E^* is a rectangular matrix with elements defined by

$$E^* = [E^*_{i,j}] \quad i = 0,1,2,\ldots,M;\ j = 0,1,2,\ldots,2M \ , \tag{5.2-10a}$$

$$E^*_{0,j} = \begin{cases} 1, & j = 0 \\ 0, & j = 1,2,3,\ldots,2M \end{cases} , \tag{5.2-10b}$$

$$E^*_{1,j} = \begin{cases} 0, & j = 0,1 \\ 1, & j = 2,3,4,\ldots,2M \end{cases} , \tag{5.2-10c}$$

$$E^*_{2,j} = \begin{cases} 0, & j = 0,1,2,3 \\ j-3, & j = 4,5,6,\ldots,2M \end{cases} , \tag{5.2-10d}$$

$$E^*_{i,j} = 0 \quad i = 3,4,5,\ldots,M;\ j = 0,1,2,\ldots,2i-1 \ , \tag{5.2-10e}$$

$$E^*_{i,j} = \frac{(j-i-1)}{(i-1)!}\ (j-i-2)\ldots(j-2i+1) \ ,$$

$$i = 3,4,5,\ldots,M;\ j = 2i,\ 2i+1,\ldots,2M \ . \tag{5.2-10f}$$

The expanded form of (5.2-7) becomes the general corrector formula for the second order case, namely

$$y_n = h^2 \left\{ \gamma_0^* \delta^{-2} f_n + \sum_{j=1}^{2M+1} C^*_j f_{n-j+1} \right\} . \tag{5.2-11}$$

5.3. Mid-Range Formulas

The formulas to be developed here find application in the construction of the starting difference table wherein a knowledge of the quantities $\delta^{-1} f_{M-\frac{1}{2}}$, $\delta^{-1} f_{M+\frac{1}{2}}$, and $\delta^{-2} f_M$ is required. They are also needed in a test criterion for change in step size to be described later.

Returning to the general algorithm for the first order case, (2.1-13), and setting n = M gives

$$y_M = h \left\{ \gamma_0 \mu\delta^{-1} f_M + \sum_{i=1}^{M} \gamma_{2i} \mu\delta^{2i-1} f_M \right\}, \qquad (5.3\text{-}1)$$

which, upon substituting from (2.1-5i), becomes

$$y_M = h \left\{ \gamma_0 \mu\delta^{-1} f_M + \tfrac{1}{2} \sum_{i=1}^{M} \gamma_{2i} \left(\nabla^{2i-1} f_{M+1} + \nabla^{2i-1} f_{M+i-1} \right) \right\}. \qquad (5.3\text{-}2)$$

Making use of the identity

$$\gamma_0 \mu\delta^{-1} f_M = \gamma_0 \delta^{-1} f_{M-\frac{1}{2}} + \tfrac{1}{2} \gamma_0 f_M \ , \qquad (5.3\text{-}3)$$

and also breaking down the backward differences into terms of function values by means of the relation

$$\nabla^k f_j = \sum_{r=0}^{k} (-1)^r \binom{k}{r} f_{j-r} \ , \qquad (5.3\text{-}4)$$

finally converts (5.3-2) into the matrix form

$$y_M = h \left\{ \gamma_0 \delta^{-1} f_{M-\frac{1}{2}} + \tfrac{1}{2} \gamma_0 f_M + Rf \right\} , \qquad (5.3\text{-}5)$$

where f is the column vector

$$f = \left\{ f_{2M} + f_{2M-1}, \ f_{2M-1} + f_{2M-2}, \ldots, f_1 + f_0 \right\} , \qquad (5.3\text{-}6)$$

and R is the row vector defined by the matrix equation

$$R = \tfrac{1}{2} \gamma \bar{B} \ , \qquad (5.3\text{-}7)$$

in which

$$\gamma = (\gamma_2, \gamma_4, \gamma_6, \cdots, \gamma_{2M}) \quad , \tag{5.3-8}$$

and $\bar{B}$ is the Mx2M rectangular matrix formed by extracting the rows of odd index i = 1,3,5,...,2M-1 from the matrix B of signed binomial coefficients, defined previously, and writing them in pyramidal form. For example, if differences of order 2M = 6 are used, then M = 3 and $\bar{B}$ takes the form

$$\bar{B} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & -3 & 3 & -1 & 0 \\ 1 & -5 & 10 & -10 & 5 & -1 \end{bmatrix} . \tag{5.3-9}$$

In a similar manner, beginning with the general algorithm for the second order case, (3.1-6), and setting n = M gives

$$y_M = h^2 \left\{ \gamma_0^* \delta^{-2} f_M + \sum_{i=1}^{M+1} \gamma_{2i}^* \delta^{2i-2} f_M \right\}, \tag{5.3-10}$$

which, upon making use of the relation

$$\delta^k f_j = \nabla^k f_{j+k/2} \quad , \tag{5.3-11}$$

becomes

$$y_M = h^2 \left\{ \gamma_0^* \delta^{-2} f_M + \sum_{i=1}^{M+1} \gamma_{2i}^* \nabla^{2i-2} f_{M+i-1} \right\}. \tag{5.3-12}$$

Application of (5.3-4) to break the differences down then yields the matrix form

$$y_M = h^2 \left\{ \gamma_0^* \delta^{-2} f_M + R^* f \right\} \quad , \tag{5.3-13}$$

where f is the column vector

$$f = f_{2M}, f_{2M-1}, \ldots, f_0 \quad , \tag{5.3-14}$$

and R^* is a row vector defined by the matrix equation

$$R^* = \gamma^* \tilde{B} \quad , \tag{5.3-15}$$

in which

$$\gamma^* = (\gamma_2^*, \gamma_4^*, \ldots, \gamma_{2M+2}^*) \quad , \tag{5.3-16}$$

and $\tilde{B}$ is the (M+1) x (2M+1) rectangular matrix formed by extracting the rows of even index i = 0, 2, 4,..., 2M from the matrix B and writing them in pryamidal form. Thus, in case differences of order 2M = 6 are used, $\tilde{B}$ takes the form

$$\tilde{B} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 \\ 1 & -6 & 15 & -20 & 15 & -6 & 1 \end{bmatrix} . \tag{5.3-17}$$

In application it is convenient to expand and collect terms in (5.3-5) to get

$$y_M = h \left\{ \gamma_0 \delta^{-1} f_{M-\frac{1}{2}} + \sum_{j=1}^{2M+1} M_j f_{2M-j+1} \right\} \quad , \tag{5.3-18}$$

where the constants M_j are obtained by equating coefficients in the expression

$$\sum_{j=1}^{2M+1} M_j f_{2M-j+1} = \tfrac{1}{2} \gamma_0 f_M + Rf \quad , \tag{5.3-19}$$

with f given by (5.3-6). Similarly, (5.3-13) can be written

$$y_M = h^2 \left\{ \gamma_0^* \delta^{-2} f_M + \sum_{j=1}^{2M+1} M_j^* f_{2M-j+1} \right\} , \qquad (5.3\text{-}20)$$

where the constants M_j^* are obtained from

$$\sum_{j=1}^{2M+1} M_j^* f_{2M-j+1} = R^* f \quad , \qquad (5.3\text{-}21)$$

in which f is now defined by (5.3-14).

A tabulation of the coefficients P, C, P^*, C^*, M, M^* to high precision is included in Table IV.

6. A General Description of an Application of Cowell's Method to the Simplest Second Order Initial Value Problem

The discussion that follows concerns the modified Cowell method as applied to the initial value problem

$$\ddot{y} = F(t, y, \dot{y}) \quad ,$$
$$y(t_0) = y_0, \ \dot{y}(t_0) = \dot{y}_0 \quad , \qquad (6.0\text{-}1)$$

where again the conventional "dot" notation signifies differentiation with respect to the independent variable t. Extension of the technique to include systems of equations will follow easily. For convenience in referencing, the principal algorithms developed in the preceding sections are rewritten here as follows. Formulas (5.1-8) and (5.2-6) become, respectively,

$$\dot{y}_n = h \left\{ \gamma_0 \delta^{-1} F_{n-\frac{1}{2}} + \sum_{j=1}^{2M+1} P_j F_{n-j} \right\} , \qquad (6.0\text{-}2a)$$

$$y_n = h^2 \left\{ \gamma_0^* \delta^{-2} F_n + \sum_{j=1}^{2M+1} P_j^* F_{n-j} \right\} , \qquad (6.0\text{-}2b)$$

TABLE 4

PREDICTOR-CORRECTOR AND MIDRANGE FORMULA COEFFICIENTS

CORRESPONDING TO DIFFERENCES OF ORDER 6

FIRST ORDER CASE

I	PREDICTOR P(I)	CORRECTOR C(I)
1	2.589955357142857142857070+00	3.042245370370370370410-01
2	-6.935251322751322751322850+00	4.603835978835978835978910-01
3	1.111928736772486772486760+01	-5.465360449735449735449870-01
4	-1.090846560846560846560850+01	4.714285714285714285714300-01
5	6.471188227513227513227400+00	-2.606068121693121693121640-01
6	-2.140939153439153439153390+00	8.247354497354497354497240-02
7	3.042245370370370370410-01	-1.136739417989417989417940-02

SECOND ORDER CASE

I	PREDICTOR P*(I)	CORRECTOR C*(I)
1	5.258793540564373897707172-01	6.549575617283950617283953-02
2	-1.485946626984126984126962+00	6.740906084656084656084192-02
3	2.395722056873306873306902+00	-1.106357473544973544973392-01
4	-2.352342372134038800705452+00	1.043705908289241622574762-01
5	1.394804067460317460317442+00	-5.999090608465608465607442-02
6	-4.611789021164021164021162-01	1.939318783068783068782702-02
7	6.549575617283950617283952-02	-2.708608906525573192239372-03

MIDRANGE

	FIRST ORDER CASE	SECOND ORDER CASE
r	M(r)	M*(r)
1	-1.57903439153439153439157@-03	-7.96406525573192239858255@-05
2	1.39550264550264550264547@-02	9.90410052910052910052693@-04
3	-6.48396164021164021164030@-02	-7.41154100529100529100502@-03
4	5.00000000000000000000000@-01	9.63348765432098765432082@-02
5	6.48396164021164021164030@-02	-7.41154100529100529100502@-03
6	-1.39550264550264550264547@-02	9.90410052910052910052693@-04
7	1.57903439153439153439157@-03	-7.96406525573192239858255@-05

TABLE 4

PREDICTOR-CORRECTOR AND MIDRANGE FORMULA COEFFICIENTS

CORRESPONDING TO DIFFERENCES OF ORDER 10

FIRST ORDER CASE

T	PREDICTOR P(T)	CORRECTOR C(T)
1	3.726253940487881460103@+00	2.742655400315990593768470-01
2	-1.655949266557773502217910+01	7.093330001402915059584400-01
3	4.777560265038780663780790+01	-1.474897943839786755453260+00
4	-9.377726152898027898027890+01	2.521788551739618806288230+00
5	1.297795026467552509219220+02	-3.239633318552589385922358[illegible]
6	-1.288225974025974025974040+02	3.068823152156885489818270[illegible]
7	9.153530254804994388327670+01	-2.111917907998637155303410+00
8	-4.558936184764309764309830+01	1.027674337622254288920750+00
9	1.515063111584595959596000+01	-3.355477424295284591944300-01
10	-3.022844996759927315482840+00	6.602641410801133023353450-02
11	2.742655400315990593768470-01	-5.924056412337662337660110-03

SECOND ORDER CASE

I	PREDICTOR P*(I)	CORRECTOR C*(I)
1	7.6593662597774494269204@-01	5.7603625837453135733564 9@-02
2	-3.5258121860541048702425 3@+00	1.3329674176576044962282 4@-01
3	1.0157528798207276629896 1@+01	-3.5761276499418240489645 7@-01
4	-1.9870968374704733831719 1@+01	6.5293053502750923385791 6@-01
5	2.7440877186647585953143 4@+01	-8.6177184834519903964265 7@-01
6	-2.7187621159030013196681 3@+01	8.2800204974423724423635 7@-01
7	1.9290481789220938526495 5@+01	-5.7474602212666448777491 9@-01
8	-9.5968170399283990553840 7@+00	2.8129526286140373441925 8@-01
9	3.1864008896574930801124 3@+00	-9.2218776748631659345824 0@-02
10	-6.3527682249790798071227 8@-01	1.8201468597570614766357 1@-02
11	5.7603625837453135733564 9@-02	-1.6369382859234876430620 9@-03

MIDRANGE

T	FIRST ORDER CASE M(I)	SECOND ORDER CASE M*(I)
1	-7.722834328737106514884411@-05	-2.601365128233514476900956@-06
2	9.618798768104323659879277@-04	3.991130840915893825948388@-05
3	-5.728518443362193362193388@-03	-3.078833403424177233701636@-04
4	2.247873075998075998076033@-02	1.691708267849140865014056@-03
5	-7.289995064734648067981477@-02	-8.736096481322175766620756@-03
6	5.000000000000000000000000@-01	9.796325655440238773571996@-02
7	7.289995064734648067981477@-02	-8.736096481322175766620756@-03
8	-2.247873075998075998076033@-02	1.691708267849140865014056@-03
9	5.728518443362193362193388@-03	-3.078833403424177233701636@-04
10	-9.618798768104323659879277@-04	3.991130840915893825948388@-05
11	7.722834328737106514884411@-05	-2.601365128233514476900956@-06

TABLE 4

PREDICTOR-CORRECTOR AND MIDRANGE FORMULA COEFFICIENTS

CORRESPONDING TO DIFFERENCES OF ORDER 14

FIRST ORDER CASE

I	PREDICTOR P(I)	CORRECTOR C(I)
1	4.77698002833012693365979@+00	2.56309496574389152514176@-01
2	-2.96611322621874112725087@+01	9.31437579714289645947268@-01
3	1.23634181301356812631200@+02	-2.74863512187655025852106@+00
4	-3.63989983932857306975479@+02	7.01336036000974823725819@+00
5	7.91976268825727851869435@+02	-1.41275211088161137936502@+01
6	-1.31040734647652510995771@+03	2.22788506128372268693955@+01
7	1.67617874961607409704927@+03	-2.75783161217074016242986@+01
8	-1.66978339718756649973589@+03	2.68271391598799006206140@+01
9	1.29488724830408138924389@+03	-2.04317847313723033072140@+01
10	-7.75105863583125866800939@+02	1.20582379492636809104905@+01
11	3.51645587210195203454016@+02	-5.40844537023524180090600@+00
12	-1.17028540678203459288125@+02	1.78312438615401072172365@+00
13	2.69702886862948328687710@+01	-4.07719736856394894175447@-01

14 -3.84866934816904917213567@+00 5.77915459@3971854783?772@-?4

15 2.56309496574389152514175@-01 -3.82699955321188442322?279@-?3

SECOND ORDER CASE

I	PREDICTOR P*(I)	CORRECTOR C*(I)
1	9.8389782739453841075921 1@-01	5.2460247680248764811664 8@-02
2	-6.2317137112765832244084 7@+00	1.9699411219080693858423 2@-01
3	2.5813963618931882309649 7@+01	-7.2338770485046291918361 6@-01
4	-7.5605614590875005566072 8@+01	1.9445509244186943203411 9@+00
5	1.6390696099211514792357 3@+02	-3.9973765073354415981524 2@+00
6	-2.7049531258610243689888 6@+02	6.3688372083291071941402 8@+00
7	3.4532809790294634507396 6@+02	-7.9317729464573690165081 2@+00
8	-3.4349786388959218729160 3@+02	7.7464040805455435109043 2@+00
9	2.6606226532716016892401 8@+02	-5.9161700671913857285393 3@+00
10	-1.5910986955821044516207 4@+02	3.4987256875151010416348 7@+00
11	7.2127076175623126570420 7@+01	-1.5717457744234044324871 @+00
12	-2.3988166997887269121559 4@+01	5.1883809208356260249470 9@-01
13	5.5251726366942717100425 6@+00	-1.1875430337410113225181 3@-01
14	-7.8802006126844908928959 7@-01	1.6844630268151404817787 6@-02
15	5.2460247680248764811664 8@-02	-1.1163460647176171146115 6@-03

MIDRANGE

I	FIRST ORDER CASE M(I)	SECOND ORDER CASE M*(I)
1	-4.0973554096866211063741 4@-06	-1.0359060115572963767794 8@-07
2	6.6838404987379855104722 8@-05	1.9606686411899527238253 4@-06
3	-5.2025784562840545534547 8@-04	-1.8152912533521765061603 7@-05
4	2.5916159651554272365559 9@-03	1.1130470208048722093230 2@-04
5	-9.4299993219431980179334 8@-03	-5.2386558160144540587927 9@-04
6	2.7737991166059673555264 3@-02	2.1517447627427318088307 3@-03
7	-7.6990174607492574175775 5@-02	-9.4514160348005442032642 0@-03
8	5.0000000000000000000000 0@-01	9.8790389305477849576042 9@-02
9	7.6990174607492574175775 5@-02	-9.4514160348005442032642 0@-03
10	-2.7737991166059673555264 3@-02	2.1517447627427318088307 3@-03
11	9.4299993219431980179334 8@-03	-5.2386558160144540587927 9@-04
12	-2.5916159651554272365559 9@-03	1.1130470208048722093230 2@-04
13	5.2025784562840545534547 8@-04	-1.8152912533521765061603 7@-05
14	-6.6838404987379855104722 8@-05	1.9606686411899527238253 4@-06
15	4.0973554096866211063741 4@-06	-1.0359060115572963767794 8@-07

and serve as predictors corresponding to a first integration and complete integration of (6.0-1). Similarly, formulas (5.1-14) and (5.2-11) take the respective forms

$$\dot{y}_n = h\left\{\gamma_0 \delta^{-1} F_{n-\frac{1}{2}} + \sum_{j=1}^{2M+1} C_j F_{n-j+1}\right\} , \qquad (6.0\text{-}3a)$$

$$y_n = h^2\left\{\gamma_0^* \delta^{-2} F_n + \sum_{j=1}^{2M+1} C_j^* F_{n-j+1}\right\} , \qquad (6.0\text{-}3b)$$

and serve as correctors. The mid-range formulas (5.3-18) and (5.3-20) become, respectively,

$$\dot{y}_M = h\left\{\gamma_0 \delta^{-1} F_{M-\frac{1}{2}} + \sum_{j=1}^{2M+1} M_j F_{2M+1-j}\right\} , \qquad (6.0\text{-}4a)$$

$$y_M = h^2\left\{\gamma_0^* \delta^{-2} F_M + \sum_{j=1}^{2M+1} M_j^* F_{2M+1-j}\right\} . \qquad (6.0\text{-}4b)$$

Two additional relations necessary for constructing a starting table of differences can be deduced directly from Figure 6. They are

$$\delta^{-1} F_{M+\ell+\frac{1}{2}} = {}^{-1}F_{M+\ell-\frac{1}{2}} + F_{M+\ell} \quad \ell = -M, -M+1, -M+2, \ldots, M \qquad (6.0\text{-}5a)$$

and

$$\delta^{-2} F_{M+\ell} = {}^{-2}F_{M+\ell-1} + {}^{-1}F_{M+\ell-\frac{1}{2}} \quad \ell = -M, -M+1, -M+2, \ldots, M+1 \qquad (6.0\text{-}5b)$$

With the above collection of formulas at hand, the essential steps involved in applying Cowell's method to (6.0-1) can be sequenced as shown in the simple flow diagram of Figure 6 and described below:

1. Set Q = 2M, the prescribed even order of differences which determines the dimensions of the starting table of central differences.

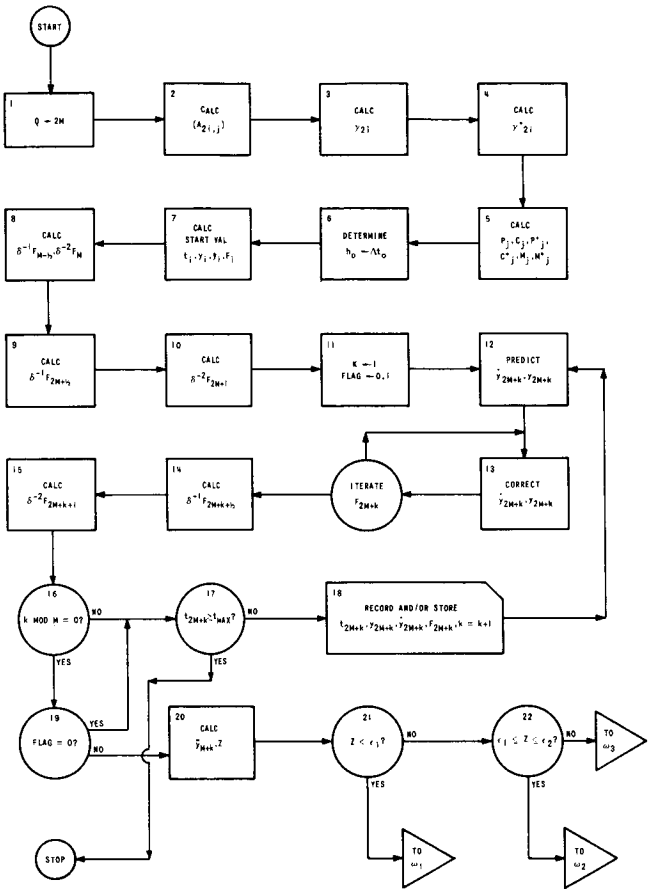

Figure 6. The Flow Diagram for the Cowell Method.

2. Calculate the coefficients

$$A_{2i,j} \quad i = 0,1,2,\ldots,M+1 \quad j = 0,1,2,\ldots,2i-1$$

using relations (2.2-4).

3. Calculate the coefficients

$$\delta_{2i} \quad i = 0,1,2,\ldots,M+2$$

using relations (2.2-7).

4. Calculate the coefficients

$$\delta^*_{2i} \quad i = 0,1,2,\ldots,M+1$$

using relations (3.2-1).

5. Calculate the coefficients

$$P_j,\ C_j,\ P^*_j,\ C^*_j,\ M_j,\ M^*_j \quad j = 1,2,3,\ldots,2M+1$$

using relations (5.1-3), (5.1-11), (5.2-3), (5.2-9), (5.3-19), and (5.3-21) with f replaced by F.

6. In accordance with some predetermined accuracy criterion, select an initial step size $h_0 = \Delta t_0$.

7. Using some appropriate starting procedure tabulate (compute and store) the starting values

$$t_i,\ y_i,\ \dot{y}_i,\ F_i \quad i = 0,1,2,\ldots 2M\ ,$$

where $t_i = t_0 + ih$, $h = h_0$, and F_i is calculated from (6.0-1).

8. Calculate the quantities

$$\delta^{-1}F_{M-\frac{1}{2}},\ \delta^{-2}F_M$$

from formulas (6.0-4).

9. Compute the quantity

$$\delta^{-1}F_{2M+\frac{1}{2}} \equiv \delta^{-1}F_{2M+1-\frac{1}{2}}$$

by successively applying (6.0-5a) with $\ell = 0,1,2,\ldots,M$.

10. Compute the quantity

$$\delta^{-2}F_{2M+1}$$

by successively applying (6.0-5b) with $\ell = 1,2,3,\ldots,M+1$.

11. Set k = 1 and Flag = 0 or 1, depending on a predetermined mode of integration. (See Step 19).

12. Predict

$$\dot{y}_{2M+k},\ y_{2M+k}$$

by using formulas (6.0-2) with n = 2M+k. Calculate F_{2m+k} from (6.0-1).

13. Correct

$$\dot{y}_{2M+k},\ y_{2M+k}$$

by using formulas (6.0-3) with n = 2M+k. Recalculate F_{2M+k} from (6.0-1). Re-enter step 13 until F_{2M+k} is stable.

14. Calculate

$$\delta^{-1}F_{2M+k+\frac{1}{2}}$$

by using (6.0-5a) with $\ell = M+k$.

15. Calculate

$$\delta^{-2} F_{2M+k+1}$$

from (6.0-5b) with $\ell = M+k+1$.

16. Branch Decision:

Is k Mod M = 0?

a. No. Continue.

b. Yes. Go to step 19.

17. Termination Decision:

Is $t_{2M+k} \geq t_{max}$?

a. No. Continue.

b. Yes. Stop.

18. Record

$$t_{2M+k},\ y_{2M+k},\ \dot{y}_{2M+k},\ F_{2M+k}$$

and set k = k+1 and re-enter step 12.

19. Branch Decision:

Is Flag = 0?

a. No. Continue.

b. Yes. Re-enter step 17.

20. Compute $\bar{y}_{M+k}$ by using (6.0-4b) with y_M replaced by $\bar{y}_{M+k}$, F_M replaced by F_{M+k}, and F_{2M-j+1} replaced by F_{2M+k-j}. Compute

$$Z = |y_{M+k} - \bar{y}_{M+k}|$$

21. Is $Z \leq \epsilon$?

a. No. Continue.

b. Yes. Go to ω_1.

22. Is $\epsilon_1 < Z \leq \epsilon_2$?

a. No. Go to ω_3.

b. Yes. Go to ω_2.

In the sections immediately following certain stages included in the flow diagram of Figure 6 are discussed in greater detail.

6.1. The Starting Procedure and Initial Step Size Determination

For the purpose of this report, the starting procedure selected was the classical Runge-Kutta 4th order one-step algorithm. As a preliminary to its use, however, the initial value problem (6.0-1) must be converted to a first order system. The substitutions

$$u_1 = y$$
$$u_2 = \dot{y} \qquad (6.1\text{-}1)$$

when applied to (6.0-1), yield the system

$$\dot{u}_1 = u_2 = f_1(t,u_1,u_2)$$
$$\dot{u}_2 = F(t,u_1,u_2) = f_2(t,u_1,u_2) \qquad (6.1\text{-}2)$$

with initial conditions

$$u_1(t_0) = y_0,\ u_2(t_0) = \dot{y}_0 \ . \qquad (6.1\text{-}3)$$

The classical Runge-Kutta equations then take the form

$$
\begin{aligned}
k_{i1} &= f_i(t_0, y_0, \dot{y}_0) \\
k_{i2} &= f_i(t_0 + \tfrac{h}{2},\ y_0 + \tfrac{h}{2}\ k_{11},\ \dot{y}_0 + \tfrac{h}{2}\ k_{21}) \\
k_{i3} &= f_i(t_0 + \tfrac{h}{2},\ y_0 + \tfrac{h}{2}\ k_{12},\ \dot{y}_0 + \tfrac{h}{2}\ k_{22}) \\
k_{i4} &= f_i(t_0 + h,\ y_0 + hk_{13},\ \dot{y}_0 + hk_{23}) \\
\Phi_i &= \frac{1}{6}\left\{k_{i1} + 2(k_{i2} + k_{i3}) + k_{i4}\right\} \quad i = 1,2 \\
u_i(t_0+h) &= u_i(t_0) + h\Phi_i
\end{aligned}
\tag{6.1-4}
$$

Successive applications of (6.1-4) yield the required starting tabulation

$$\left\{t_i,\ y_i,\ \dot{y}_i,\ F_i\right\} \quad i = 0,1,2,\ldots,2M$$

once the initial increment

$$h = h_0 = \Delta t_0$$

is prescribed.

The technique applied here for determining h_0, step 6, is detailed flow-wise as follows (see Figure 7).

6a. Select an arbitrary initial value for h, say

$$h = (t_{max} - t_0)/2^p \quad , \tag{6.1-5}$$

with $p = 3$.

6b. Apply (6.1-4) once to get the vector

$$U\ (t_0+h) = (u_1(t_0+h),\ u_2(t_0+h))$$

in one step and calculate

$$Z_1 = \sqrt{u^2_1+u^2_2} \quad .$$

6c. Replace h by h/2 and apply (6.1-4) twice to get $U\ (t_0+2h)$ in two steps. Calculate

$$Z_2 = \sqrt{u^2_1+u^2_2} \quad .$$

6d. Calculate

$$Z = |\ Z_2 - Z_1\ | \quad .$$

6e. Test:

Is $Z > \epsilon_0$?

a. Yes. Continue.

b. No. Go to step 6.7.

6f. Set p = p + 1. Calculate

$$h = (t_{max} - t_0)/2^p$$

and re-enter step 6b.

6g. Set $h_0 = 2h$.

The above procedure has been found to work well as an initializing routine in conjunction with Cowell's method. It also finds convenient application as a restart procedure in modes of integration calling for intermediate

step size control. This option is provided for in Figure 6 by setting Flag = 1 in step 11.

6.2. Intermediate Step Size Control

The option of step size control is exercised by setting a variable Flag in step 11 of Figure 6. If Flag = 0, the integration proceeds until termination using the constant step size $h = h_0$. If Flag = 1, the integration proceeds for M steps. After completion of the M-th step, the current mid-range value of the dependent variable is tested against a step-change criterion which comprises steps 20, 21, and 22 of Figure 6. If the agreement between the current mid-range value y_{M+k} yielded by Cowell's method and a comparison value $\bar{y}_{M+k}$ computed from mid-range formula (6.0-4b) is sufficiently good, i.e., if their relative difference is less in absolute value than a prescribed minimum tolerance ϵ_1, this is taken as an indication that the step size h should be increased. In this case, the technique used in the present application calls for a doubling of the step size which is achieved at branch point ω_1 as indicated in step 21 (see Figure 6). If the above-mentioned agreement is not as good as ϵ_1 but is still at least as good as a prescribed maximum tolerance ϵ_2, this is taken as an indication that the step size can remain unchanged. However, in a computer implemented application some re-orientation in the arrays containing the tabulated variables may be necessary. This is provided for at branch point ω_2 as indicated in step 22 (see Figure 6). If both of the tolerances ϵ_1 and ϵ_2 are exceeded, a decrease in step size is indicated. In the present application this calls for a restart at the mid-range point which includes a new step size determination. That is, the last M points yielded by Cowell's method are rejected and steps 6 and 7 are repeated with $t_0 = t_{M+k}$, $y_0 = y_{M+k}$, and $\dot{y}_0 = \dot{y}_{M+k}$. This is provided for at branch point ω_3 as indicated in step 22 (see Figure 6).

Table IV contains a listing of high precision values of the coefficients P, C, P^*, C^*, M, and M^* corresponding to differences of orders Q = 2M = 6, 10, and 14.

7. Predictor-Corrector Formulas (Expanded Form)

7.1. First Order Case - Predictor

To demonstrate the expansion process in terms of the differences appearing in Figure 5 for a simple case set n = 3 and 2M = 4 in (2.1-13) to get

$$y_3 = h \sum_{i=0}^{2} \gamma_{2i} \mu\delta^{2i-1} f_3 \ ,$$

$$= h \left\{ \gamma_0 \mu\delta^{-1} + \gamma_2 \mu\delta + \gamma_4 \mu\delta^3 \right\} f_3 \ ,$$

$$= h \left\{ \tfrac{1}{2} \left(\delta^{-1} f_{5/2} + \delta^{-1} f_{7/2} \right) \gamma_0 \right.$$

$$+ \tfrac{1}{2} \left(\delta f_{5/2} + \delta f_{7/2} \right) \gamma_2$$

$$\left. + \tfrac{1}{2} \left(\delta^3 f_{5/2} + \delta^3 f_{7/2} \right) \gamma \right\} . \qquad (7.1\text{-}1)$$

The objective now is to express each of the central differences in (7.1-1) entirely in terms of the function values

$$\left\{ f_{2-i} \right\} , \quad i = 0,1,2,3,4 \ .$$

This is conveniently done in two stages. The first stage consists in writing each estimated difference appearing in Figure 5 in terms of the known differences on the diagonal joining f_2 to $\delta^4 f_0$ (diagonal 1 in the figure). Use is then made of the fact that on this diagonal the entries are simply the backward differences of f_2. That is,

$$\delta^k f_r = \nabla^k f_2 \quad , \tag{7.1-2}$$

independent of r. This allows the estimated differences to be restated in terms of backward differences, and hence the algorithm (7.1-1) can be so expressed. The second stage consists in making use of the relation

$$\nabla^k f_i = \sum_{j=0}^{k} (-1)^j \binom{k}{j} f_{i-j} \tag{7.1-3}$$

to break the backward differences down into terms of function values. Collecting terms then yields the required expanded formula (5.1-8).

Making use of the manner in which the table of differences was constructed, the first stage begins with writing

$$\begin{aligned}
\delta^4 f_1 &= \delta^4 f_0 \quad , \\
\delta^3 f_{3/2} &= \delta^3 f_{1/2} + \delta^4 f_1 \quad , \\
\delta^2 f_2 &= \delta^2 f_1 + \delta^3 f_{3/2} \quad , \\
\delta f_{5/2} &= \delta f_{3/2} + \delta^2 f_2 \quad , \\
f_3 &= f_2 + \delta f_{5/2} \quad , \\
\delta^{-1} f_{7/2} &= \delta^{-1} f_{5/2} + f_3 \quad .
\end{aligned} \tag{7.1-4}$$

Adding these relations m at a time for m = 1,2,...,6 and using (7.1-2) then gives the set of equivalents

$$
\begin{aligned}
\delta^4 f_1 &= \nabla^4 f_2 \ , \\
\delta^3 f_{3/2} &= \nabla^3 f_2 + \nabla^4 f_2 \ , \\
\delta^2 f_2 &= \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2 \ , \\
\delta f_{5/2} &= \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2 \ , \\
f_3 &= f_2 + \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2 \ , \\
\delta^{-1} f_{7/2} &= \delta^{-1} f_{5/2} + f_2 + \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2 \ ,
\end{aligned}
\tag{7.1-5}
$$

which serves to express each entry on diagonal 2 of Figure 5 in terms of the backward differences on diagonal 1. Diagonal 3 is treated in the same manner relative to diagonal 2, the final reduction being achieved through relations (7.1-5). Thus,

$$
\begin{aligned}
\delta^4 f_2 &= \delta^4 f_1 \ , \\
\delta^3 f_{5/2} &= \delta^3 f_{3/2} + \delta^4 f_2 \ , \\
\delta^2 f_3 &= \delta^2 f_2 + \delta^3 f_{5/2} \ , \\
\delta f_{7/2} &= \delta f_{5/2} + \delta^2 f_3 \ ,
\end{aligned}
\tag{7.1-6}
$$

from which, upon adding m at a time for m = 1,2,3,4,

$$\begin{aligned}
\delta^4 f_2 &= \delta^4 f_1 \ , \\
\delta^3 f_{5/2} &= \delta^3 f_{3/2} + \delta^4 f_1 \ , \\
\delta^2 f_3 &= \delta^2 f_2 + \delta^3 f_{3/2} + \delta^4 f_1 \ , \\
\delta f_{7/2} &= \delta f_{5/2} + \delta^2 f_2 + \delta^3 f_{3/2} + \delta^4 f_1 \ .
\end{aligned} \tag{7.1-7}$$

Using relations (7.1-5) in these equations then gives the result

$$\begin{aligned}
\delta^4 f_2 &= \nabla^4 f_2 \ , \\
\delta^3 f_{5/2} &= \nabla^3 f_2 + 2\nabla^4 f_2 \ , \\
\delta^2 f_3 &= \nabla^2 f_2 + 2\nabla^3 f_2 + 3\nabla^4 f_2 \ , \\
\delta f_{7/2} &= \nabla f_2 + 2\nabla^2 f_2 + 3\nabla^3 f_2 + 4\nabla^4 f_2 \ .
\end{aligned} \tag{7.1-8}$$

Finally, for diagonal 4 relative to diagonal 3 one can write

$$\begin{aligned}
\delta^4 f_3 &= \delta^4 f_2 \ , \\
\delta^3 f_{7/2} &= \delta^3 f_{5/2} + \delta^4 f_3 \ ,
\end{aligned} \tag{7.1-9}$$

from which

$$\delta^4 f_3 = \nabla^4 f_2 \ ,$$

$$\delta^3 f_{7/2} = \delta^3 f_{5/2} + \delta^4 f_2 \ , \qquad (7.1\text{-}10)$$

$$= \nabla^3 f_2 + 3\nabla^4 f_2 \ .$$

Substituting from (7.1-5,8,10) into (7.1-1) now leads to the relations

$$\mu\delta^{-1} f_3 = \tfrac{1}{2}\left(\delta^{-1} f_{5/2} + \delta^{-1} f_{7/2}\right) \ ,$$

$$= \delta^{-1} f_{5/2} + \tfrac{1}{2}\left(f_2 + \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2\right) \ , \qquad (7.1\text{-}11)$$

$$\mu\delta f_3 = \tfrac{1}{2}\left(\delta f_{5/2} + \delta f_{7/2}\right) \ ,$$

$$= \tfrac{1}{2}\left(2\nabla f_2 + 3\nabla^2 f_2 + 4\nabla^3 f_2 + 5\nabla^4 f_2\right) \ , \qquad (7.1\text{-}12)$$

$$\mu\delta^3 f_3 = \tfrac{1}{2}\left(\delta^3 f_{5/2} + \delta^3 f_{7/2}\right) \ ,$$

$$= \tfrac{1}{2}\left(2\nabla^3 f_2 + 5\nabla^4 f_2\right) \ , \qquad (7.1\text{-}13)$$

so that (7.1-1) takes the form

$$y_3 = \Big\{ h \quad \gamma_0 \delta^{-1} f_{5/2}$$

$$+ \tfrac{1}{2} \Big[\left(f_2 + \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2\right) \gamma_0$$

$$+ \left(2\Delta f_2 + 3\Delta^2 f_2 + 4\Delta^3 f_2 + 5\Delta^4 f_2\right) \gamma_2$$

$$+ \left(2\nabla^3 f_2 + 5\nabla^4 f_2\right) \gamma_4 \Big] \Big\} ,$$

$$= h \left\{ \gamma_0 \delta^{-1} f_{5/2} + \tfrac{1}{2} \left[\gamma_0 f_2 + (\gamma_0 + 2\gamma_2) \nabla f_2 \right.\right.$$

$$+ (\gamma_0 + 3\gamma_0) \nabla^2 f_2 + (\gamma_0 + 4\gamma_2 + 2\gamma_4) \nabla^3 f_2$$

$$\left.\left. + (\gamma_0 + 5\gamma_2 + 5\gamma_4) \nabla^4 f_2 \right] \right\} . \qquad (7.1\text{-}14)$$

This completes the first stage of the reduction. The second stage is achieved by substituting from (7.1-3) into (7.1-14) to get

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} \right.$$

$$+ \tfrac{1}{2} \left[\gamma_0 f_2 + (\gamma_0 + 2\gamma_2)(f_2 - f_1) \right.$$

$$+ (\gamma_0 + 3\gamma_2)(f_2 - 2f_1 + f_0)$$

$$+ (\gamma_0 + 4\gamma_2 + 2\gamma_4)(f_2 - 3f_1 + 3f_0 - f_{-1})$$

$$\left.\left. + (\gamma_0 + 5\gamma_2 + 5\gamma_4)(f_2 - 4f_1 + 6f_0 - 4f_{-1} + f_{-2}) \right] \right\} ,$$

which, after multiplying out and collecting terms, can be written in the convenient matrix form

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} + Pf \right\} , \qquad (7.1\text{-}15)$$

where P is the row vector defined by

$$P = \tfrac{1}{2} \gamma DB \quad , \qquad (7.1\text{-}16)$$

in which γ is the row vector

$$\gamma = (\gamma_0, \gamma_2, \gamma_4) \quad , \qquad (7.1\text{-}17)$$

D is the 3x5 rectangular matrix

$$D = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 3 & 4 & 5 \\ 0 & 0 & 0 & 2 & 5 \end{bmatrix}, \qquad (7.1\text{-}18)$$

B is the 5x5 triangular matrix of signed binomial coefficients

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 1 & -3 & 3 & -1 & 0 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}, \qquad (7.1\text{-}19)$$

and f is the column vector of function values

$$f = \left\{ f_2,\ f_1,\ f_0,\ f_{-1},\ f_{-2} \right\} . \qquad (7.1\text{-}20)$$

In expanded form (7.1-15) becomes

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} + \sum_{j=1}^{5} P_j f_{3-j} \right\} , \qquad (7.1\text{-}21)$$

wherein the coefficients P_j are rational numbers given by (7.1-16).

7.2. First Order Case - Corrector

Here the objective is to express each of the central differences in (7.1-1) entirely in terms of the function values

$$\left\{ f_{3-i} \right\}, \quad i = 0,1,2,3,4 \ .$$

The first stage of the process consists in writing each estimated difference appearing below diagonal 2 of Figure 5 in terms of the differences on

diagonal 2. Use is then made of the fact that on diagonal 2 the entries are equivalent to backward differences of f_3. That is,

$$\delta^k f_r = \nabla^k f_3 \quad , \tag{7.2-1}$$

independent of r. Proceeding directly to equations (7.1-7) and using (7.2-1) gives the desired result for diagonal 3 relative to diagonal 2. The results obtained are as follows:

$$\begin{aligned} \delta^4 f_2 &= \nabla^4 f_3 \quad , \\ \delta^3 f_{5/2} &= \nabla^3 f_3 + \nabla^4 f_3 \quad , \\ \delta^2 f_3 &= \nabla^2 f_3 + \nabla^3 f_3 + \nabla^4 f_3 \quad , \\ \delta f_{7/2} &= \nabla f_3 + \nabla^2 f_3 + \nabla^3 f_3 + \nabla^4 f_3 \quad . \end{aligned} \tag{7.2-2}$$

Adding equations (7.1-9) gives

$$\begin{aligned} \delta^4 f_3 &= \delta^4 f_2 \quad , \\ \delta^3 f_{7/2} &= \delta^3 f_{5/2} + \delta^4 f_2 \quad , \end{aligned} \tag{7.2-3}$$

so that application of (7.2-1) and (7.2-2) yields

$$\begin{aligned} \delta^4 f_3 &= \nabla^4 f_3 \quad , \\ \Delta^3 f_{7/2} &= \nabla^3 f_3 + 2\nabla^4 f_3 \quad . \end{aligned} \tag{7.2-4}$$

Finally,

$$\delta^{-1}f_{7/2} = \delta^{-1}f_{5/2} + f_3 \quad . \tag{7.2-5}$$

Substitution from (7.2-2), (7.2-4), and (7.2-5) now gives

$$\mu\delta^{-1}f_3 = \tfrac{1}{2}\,(\delta^{-1}f_{5/2} + \delta^{-1}f_{7/2}) = \delta^{-1}f_{5/2} + \tfrac{1}{2}\,f_3 \quad , \tag{7.2-6a}$$

$$\mu\delta f_3 = \tfrac{1}{2}\,(\delta f_{5/2} + \delta f_{7/2}) = \tfrac{1}{2}\,(2\nabla f_3 + \nabla^2 f_3 + \nabla^3 f_3 + \nabla^4 f_3) \quad , \tag{7.2-6b}$$

$$\mu\delta^3 f_3 = \tfrac{1}{2}\,(\delta^3 f_{5/2} + \delta^3 f_{7/2}) = \tfrac{1}{2}\,(2\nabla^3 f_3 + 3\nabla^4 f_3) \quad , \tag{7.2-6c}$$

so that (7.1-1) becomes

$$\begin{aligned} y_3 = h\Big\{&(\delta^{-1}f_{5/2} + \tfrac{1}{2}\,f_3)\,\gamma_0 \\ &+ \tfrac{1}{2}\,(2\nabla f_3 + \nabla^2 f_3 + \nabla^3 f_3 + \nabla^4 f_3)\,\gamma_2 \\ &+ \tfrac{1}{2}\,(2\nabla^3 f_3 + 3\nabla^4 f_3)\,\gamma_4\Big\} \quad , \\ = h\Big\{&\gamma_0\delta^{-1}f_{5/2} + \tfrac{1}{2}\,\Big[\gamma_0 f_3 + 2\gamma_2\nabla f_3 + \gamma_2\nabla^2 f_3 \\ &+ (\gamma_2 + 2\gamma_4)\,\nabla^3 f_3 + (\gamma_2 + 3\gamma_4)\,\nabla^4 f_3\Big]\Big\}. \end{aligned} \tag{7.2-7}$$

The second stage is now completed by applying (7.1-3) in (7.2-7) to get

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} + \tfrac{1}{2} \left[\gamma_0 f_3 + 2\gamma_2 (f_3 - f_2) \right.\right.$$

$$+ \gamma_2 (f_3 - 2f_2 + f_1)$$

$$+ (\gamma_2 + 2\gamma_4)(f_3 - 3f_2 + 3f_1 - f_0)$$

$$\left.\left. + (\gamma_2 + 3\gamma_4)(f_3 - 4f_2 + 6f_1 - 4f_0 + f_{-1}) \right] \right\} , \qquad (7.2\text{-}8)$$

which, upon collecting terms, can be put in the matrix form

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} + Cf \right\} , \qquad (7.2\text{-}9)$$

where C is the row vector defined by the matrix product

$$C = \tfrac{1}{2} \gamma E B \quad , \qquad (7.2\text{-}10)$$

in which γ and B are the same as defined in Section 7.1; E is the 3x5 rectangular matrix

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 1 \\ 0 & 0 & 0 & 2 & 3 \end{bmatrix} , \qquad (7.2\text{-}11)$$

and f is the column vector

$$f = \left\{ f_3, \; f_2, \; f_1, \; f_0, \; f_{-1} \right\} . \qquad (7.2\text{-}12)$$

In expanded form (7.2-9) becomes

$$y_3 = h \left\{ \gamma_0 \delta^{-1} f_{5/2} + \sum_{j=1}^{5} C_j f_{3-j+1} \right\} , \qquad (7.2\text{-}13)$$

where the coefficients C_j are given by (7.2-10).

7.3. Second Order Case - Predictor

Setting n = 3 and 2M = 4 in general algorithm (3.1-6) gives

$$y_3 = \left\{ \sum_{i=0}^{3} \gamma_{2i}^{*} \delta^{2i-2} \right\} h^2 f_3 \quad ,$$

$$= h^2 (\gamma_0^{*} \delta^{-2} f_3 + \gamma_2^{*} f_3 + \gamma_4^{*} \delta^2 f_3 + \gamma_6^{*} \delta^4 f_3) \quad , \qquad (7.3\text{-}1)$$

which is to be re-expressed in terms of the function values

$$\left\{ f_{2-i} \right\} \quad , \quad i = 0,1,2,3,4 \quad .$$

This is readily done by referring to (7.1-5), (7.1-8), and (7.1-10) to get

$$f_3 = f_2 + \nabla f_2 + \nabla^2 f_2 + \nabla^3 f_2 + \nabla^4 f_2 \quad ,$$

$$\delta^2 f_3 = \nabla^2 f_2 + 2\nabla^3 f_2 + 3\nabla^4 f_2 \quad , \qquad (7.3\text{-}2)$$

$$\delta^4 f_3 = \nabla^4 f_2 \quad .$$

Substitution into (7.3-1) then gives after rearrangement

$$y_3 = h^2 \left\{ \gamma_0^{*} \delta^{-2} f_3 + \gamma_2^{*} f_2 + \gamma_2^{*} \nabla f_2 \right.$$

$$+ (\gamma_2^{*} + \gamma_4^{*}) \nabla^2 f_2 + (\gamma_2^{*} + 2\gamma_4^{*}) \nabla^3 f_2$$

$$\left. + (\gamma_2^{*} + 3\gamma_4^{*} + \gamma_6^{*}) \nabla^4 f_2 \right\} \quad , \qquad (7.3\text{-}3)$$

which completes the first stage. Application of (7.1-3) and collecting terms according to function values finally yields the expanded algorithm which can be written

$$y_3 = h^2 \left\{ \gamma_0^* \delta^{-2} f_3 + P^* f \right\}, \qquad (7.3\text{-}4)$$

where

$$P^* = \gamma^* D^* B \quad , \qquad (7.3\text{-}5)$$

$$\gamma^* = (\gamma_2^*, \gamma_4^*, \gamma_6^*) \quad , \qquad (7.3\text{-}6)$$

$$D^* = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad (7.3\text{-}7)$$

and B and f are the same as defined by (7.1-19) and (7.1-20), respectively. In expanded form (7.3-4) becomes the predictor formula

$$y_3 = h^2 \left\{ \gamma_0^* \delta^{-2} f_3 + \sum_{j=1}^{5} P_j^* f_{3-j} \right\}, \qquad (7.3\text{-}8)$$

wherein the coefficients P_j^* are given by (7.3-5).

7.4. Second Order Case - Corrector

The objective here is to express (7.3-1) in terms of the function values

$$\left\{ f_{3-i} \right\}, \quad i = 0,1,2,3,4 \quad .$$

This is easily done by referring to (7.2-2) and (7.2-4) to get

$$\delta^2 f_3 = \nabla^2 f_3 + \nabla^3 f_3 + \nabla^4 f_3 \quad ,$$
$$\delta^4 f_3 = \nabla^4 f_3 \quad , \tag{7.4-1}$$

so that (7.3-1) takes the form

$$y_3 = h^2 \left\{ \gamma_0^* \delta^{-2} f_3 + \gamma_2^* f_3 + \gamma_4^* \nabla^2 f_3 + \gamma_4^* \nabla^3 f_3 + (\gamma_4^* + \gamma_6^*) \nabla^4 f_3 \right\} \quad . \tag{7.4-2}$$

Applying (7.1-3) and collecting terms then yields the matrix form

$$y_3 = h^2 \left\{ \gamma_0^* \delta^{-2} f_3 + C^* f \right\} \quad , \tag{7.4-3}$$

where

$$C^* = \gamma^* E^* B \quad , \tag{7.4-4}$$

$$\gamma^* = (\gamma_2^*, \gamma_4^*, \gamma_6^*) \quad , \tag{7.4-5}$$

$$E^* = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{7.4-6}$$

$$f = f_3, f_2, f_1, f_0, f_{-1} \quad , \tag{7.4-7}$$

and B is the matrix of signed binomial coefficients defined by (7.1-19).

In expanded form (7.4-3) becomes the corrector formula

$$y_3 = h^2 \left\{ \gamma_0^* \delta^{-2} f_3 + \sum_{j=1}^{5} C_j^* f_{3-j+1} \right\} \quad , \tag{7.4-8}$$

in which the coefficients C_j^* are calculated from (7.4-4).

8. The Computer Program

8.1. The Initial Value Problem

The Cowell's method computer program outlined here is specifically adapted for tabulating certain periodic orbits associated with the restricted three-body problem. These orbits are solutions of the special second order system

$$\ddot{x} = x + 2\dot{y} - \mu' \frac{(x+\mu)}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{(x-\mu')}{((x-\mu')^2+y^2)^{3/2}} ,$$

$$\ddot{y} = y - 2\dot{x} - \mu' \frac{y}{((x+\mu)^2+y^2)^{3/2}} - \mu \frac{y}{((x-\mu')^2+y^2)^{3/2}} , \qquad (8.1\text{-}1)$$

where $\mu' = (1 - \mu)$, and μ is a parameter representing the relative mass of the Moon compared to that of the Earth plus Moon. Initial conditions peculiar to each orbit are given in the form

$$x(0) = x_0, \ \dot{x}(0) = \dot{x}_0 ,$$

$$y(0) = y_0, \ \dot{y}(0) = \dot{y}_0 , \qquad (8.1\text{-}2)$$

and it can be assumed that in each case the period is known with fair precision (15 to 20 decimal places). A typical set of initial values is listed below:

$$x_0 = 0.994 ,$$

$$y_0 = 0 ,$$

$$\dot{x}_0 = 0 ,$$

$$\dot{y}_0 = -2.031732629557336835 66 ,$$

μ = 0.012277471 ,

Period = 11.124340337266085135070 .

8.2. Operating Instructions

The subject program was written in Extended Algol 60 as implemented by the Burroughs B-5000 data processor. There are no unusual hardware requirements and the program will run on a minimum B-5000 system with no unusual accessories. It makes use of double precision floating point arithmetic which retains 23 decimal digits in all computations. Provision is made for the application of Cowell's method to the initial value problem (8.1-1) using predictor-corrector formulas corresponding to differences of even order

$$Q = 2M,\ M = 3,4,5,\ldots,10$$

where Q is an input parameter to be specified by the user.

The running of a single orbit requires a total of 11 input data cards which are simple to prepare. Data entries can be in any one of a choice of formats and separate data-sets can be sequenced indefinitely. This particular feature is implemented by a special double precision free field read procedure. A typical data-set is described below.

Card	Contains
1	$x_0 = 0.994$
2	$y_0 = 0$
3	$\dot{x}_0 = 0$
4	$\dot{y}_0 = -2.03173262955733683566$
5	$\mu = 0.012277471$

Card	Contains
6	period = 11.124340337266085135O7
7	Q = 10
8	ϵ_0 = 5.0@-16
9	ϵ_1 = 5.0@-14
10	ϵ_2 = 5.0@-9
11	0, or 1,

If it is desired to identify a card with comments, this can be done by putting an asterisk (*) after the number. Entries following the * are ignored by the card read routine. The quantities ϵ_0, ϵ_1, and ϵ_2 on cards 8-10 are concerned with error tolerances in the initial stepsize selection routine, step 6 in Figure 6; the predictor-corrector iteration, step 13; and the stepsize change criterion, steps 20, 21, and 22. These are discussed in more detail in subsequent sections. In Card 11 a 0, indicates that more data sets are to follow. The entry 1, indicates termination of the card read procedure, i.e., no more data sets will be processed.

As concerns the frequency and format of the line printer output, the current program provides for a print out of the integration results every Q steps. The tabulation includes current double precision values of the variables t, x, and y as well as integer values of certain counters designated by STEP, P, ITER, and START. The column headed STEP gives the number of accepted integration steps which have survived error tolerances and subsequent restarts. The column headed P indicates that the current stepsize is Period/2^P. ITER indicates the number of times the corrector was entered in order to stabilize the current values of the dependent variables. START indicates the number of restarts that have been necessary to reach the current stage of the

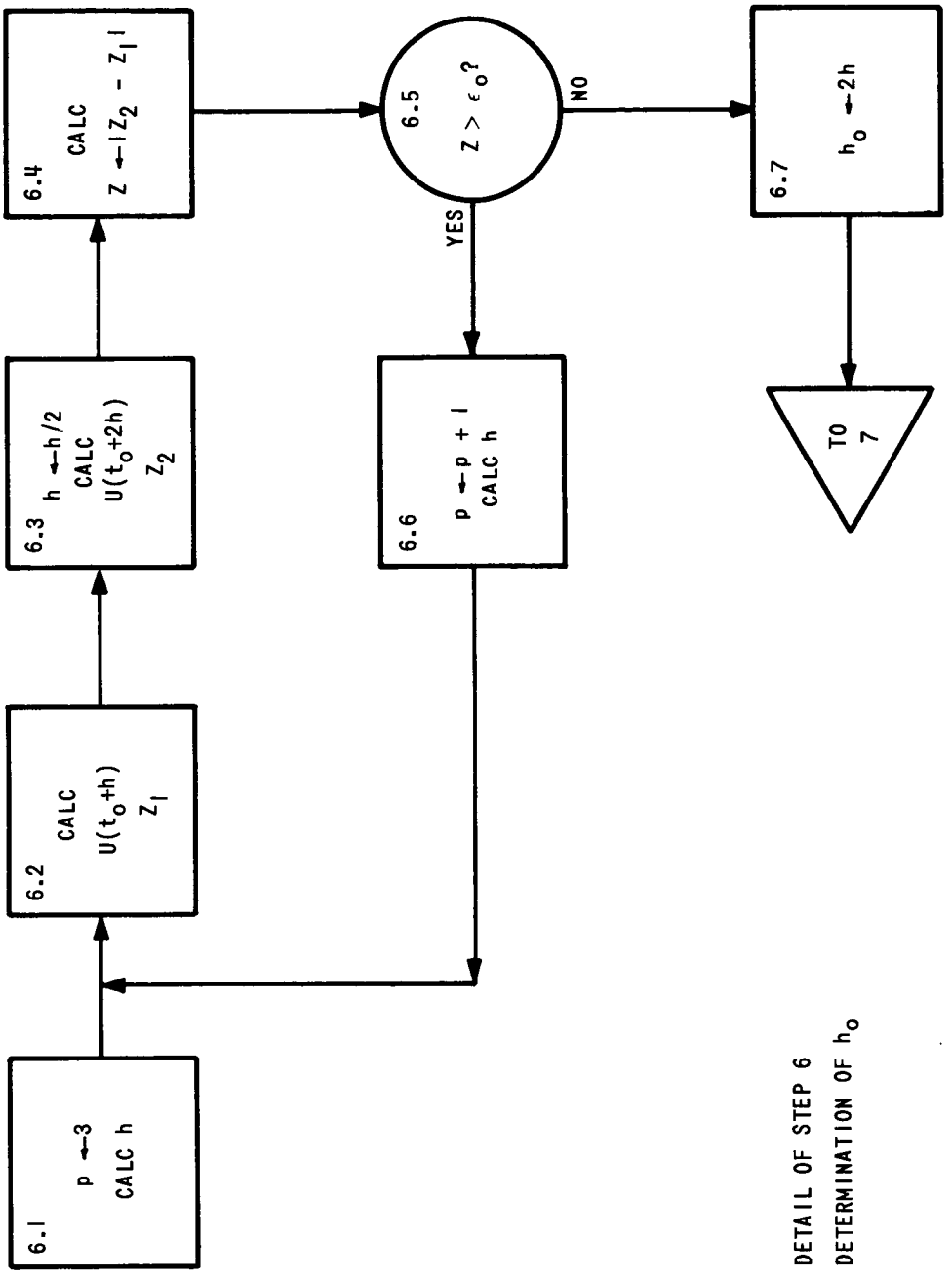

Figure 7. The Flow Diagram for the Initial Step Size Determination.

integration, i.e., it gives the number of times the stepsize change criterion has been exercised. A sample input is included in the accompanying program listing.

To operate the program place one or more data-sets at the end of the program deck provided. When compilation has been completed, runs will take place and continue until the file of data-sets is exhausted.

8.3. Error Control

The basic error control technique used in the subject program hinges upon reducing the second order system (8.1-1) to a first order system by the change in variables

$$x = u_1 \ , \quad y = u_2 \ ,$$

$$\dot{x} = u_3 \ , \quad \dot{y} = u_4 \ .$$

The first order system takes the form

$$\dot{u}_1 = u_3 = f_1(\vec{u}) \ ,$$

$$\dot{u}_2 = u_4 = f_2(\vec{u}) \ ,$$

$$\dot{u}_3 = f(u_1,u_2,u_4) = f_3(\vec{u}) \ ,$$

$$\dot{u}_4 = g(u_1,u_2,u_3) = f_4(\vec{u}) \ ,$$

where

$$\vec{u} = (u_1,u_2,u_3,u_4) = (x,y,\dot{x},\dot{y}) \ .$$

This must be done in order to make use of the Runge-Kutta starting procedure. Control is then applied to the length of the vector $\vec{u}$ given by

$$u = \sqrt{u_1^2 + u_2^2 + u_3^2 + u_4^2}$$

the process has already been described and flow diagrammed in Figure 7 for the initial stepsize determination routine. For the stepsize change criterion, the process runs as follows (see Figure 6).

20.1. Assume that the integration has been successfully carried on for $Q = 2M$ steps, the successive vectors u and the accelerations f_3, f_4 being stored sequentially. Continue the process for $M = Q/2$ additional steps.

20.2. Drop back M steps and compute a vector $\vec{\bar{u}}_{2M}$ using the mid-range formulas.

20.3. Calculate the relative difference

$$Z = \left| \frac{u_{2M} - \bar{u}_{2M}}{u_{2M}} \right|$$

20.4. If $Z \leq \epsilon_1$, or $\epsilon_1 < Z \leq \epsilon_2$ then continue the integration M more steps and retest in step 21. If $\epsilon_2 < Z$ go to step ω_3.

21. Retest. If $Z \leq \epsilon_1$, go to step ω_1, otherwise go to step 22.

22. If $\epsilon_1 < Z \leq \epsilon_2$, go to step ω_2, otherwise go to step ω_3.

Step ω_1

Build a new table of differences using the stored values

$$u_{2i},\ (f_3)_{2i},\ (f_4)_{2i} \quad i = 0,\ 2,\ 4,\ \ldots,\ Q = 2M$$

and continue the integration with doubled stepsize.

Step ω_2

Build a new table using the values

$$u_{2M+i},\ (f_3)_{2M+i},\ (f_4)_{2M+i} \quad i = 0,\ 1,\ 2,\ \ldots,\ Q = 2M$$

and continue the integration with stepsize unchanged.

Step ω_3

Drop back M steps and do a complete restart which includes a new initial stepsize determination and table construction.

In the predictor-corrector, iteration control is maintained on the acceleration vector

$$\vec{a} = (\dot{u}_3,\ \dot{u}_4) = (\ddot{x},\ \ddot{y}) \ .$$

This runs as follows (see Figure 6):

12.1 Predict $\vec{u}$

12.2 Calculate $\vec{a}$ and $Z_1 = \sqrt{\ddot{p}^2 + \ddot{p}^2}$

13.1 Correct $\vec{u}$

13.2 Calculate $\vec{a}$ and $Z_2 = \sqrt{\ddot{c}^2 + \ddot{c}^2}$

13.3 Test. If $|(Z_2 - Z_1)/Z_2| \leq \epsilon_0$, then go to step 14, otherwise set $Z_1 = Z_2$ and re-enter step 13.1.

9. Program Listing for Cowell's Method

The following 42 pages lists the program for Cowell's Method.

```
 LABEL   000000000XXXXXX0010000001
                                                                                00000000
BEGIN                                                                           00001000
  INTEGER CTR ,CROSS,I ,ITER ,J ,K ,L ,M ,MSK ,MASK ,N ,NU ,NV ,NT ,NZ          00002000
  ,P ,P0 ,P1 ,RSCTR ,S ,SN ,SNI ,SNR ,SNS ,T ,TAG ,ZZZI, SGN, SGV ;             00003000
  INTEGER ORBCTR, RUN ;                                                         00004000
  REAL C2H,C2L ,DAL ,DEL ,HH,HL ,H1,L1 ,H2,L2 ,H3,L3 ,H4,L4 ,H5,L5 ,H1H        00005000
  ,H1L ,H2H,H2L ,MPH,MPL ,MUH,MUL ,PIH,PIL ,PRDH,PRDL ,PRODH,PRODL ,           00006000
  SLOPEH,SLOPEL ,SUMH,SUML ,SUM1H,SUM1L ,SUM2H,SUM2L ,TH,TL ,TOH,TOL ,         00007000
  T1H,T1L ,T2H,T2L ,TMH,TML ,TM1 ,TM2 ,TM3 ,TWOPIH,TWOPIL ,TZH,TZL ,X ,Y       00008000
  ,Z ,ZH,ZL ,Z1H,Z1L ,Z2H,Z2L ,Z3H,Z3L ,RH, RL, MINH, MINL, DIL, DOL ;         00009000
  REAL R1H, R1L, R2H, R2L, F3H, F3L, F4H, F4L, WH, WL ;                         00010000
  REAL V1H,V1L,V2H,V2L,V3H,V3L ;                                                00011000
  BOOLEAN B, BL, BLO, BL1, FLAG, PRNT, DONE, TAB ;                              00012000
  BOOLEAN TAB1, TAB2, TAB3, TAB4, TAB5, TAB6 ;                                  00013000
  ALPHA ARRAY ALPHA1,ALPHA2,ALPHA3,ALPHA4,ALPHA5,ALPHA6,ALPHA7 , ALPHA8         00014000
  ,ALPHA9,ALPHA10[0:4], A[0:77] ;                                               00015000
  INTEGER ARRAY Q[0:25];                                                        00016000
  REAL ARRAY AH,AL[0:50,0:50],BZH,BZL[0:4,0:50],CON[0:7],DH,DL[0:15,0:50000017000
  ],DELH,DELL[0:10],FACTH,FACTL[0:50],FH,FL[0:50],FNH,FNL[0:10,0:50],           00018000
  FRSTH,FRSTL[0:10,0:50],GH,GL[0:2,0:50],PCCH,PCCL[0:7,0:50,0:6],PCMH,          00019000
  PCML[0:15,0:50],RKH,RKL[0:4],SCNDH,SCNDL[0:10,0:50],TNH,TNL[0:50],TSH         00020000
  ,TSL[0:50],UH,UL[0:10],U0H,U0L[0:10],U1H,U1L[0:10],U2H,U2L[0:10],UMH,         00021000
  UML[0:10],UPH,UPL[0:10],UZH,UZL[0:10],VNH,VNL[0:10,0:50];                     00022000
  REAL ARRAY ICH,ICL[0:20] ;                                                    00023000
  FILE IN CARD (3,10) ;                                                         00024000
  LABEL ASSIGN ,CONTROL ,COWELL ,GAMMA ,OUTSET ,PCCOEFF ,RET0 ,RET1 ,           00025000
  RET2 ,RET3 ,RET4 ,FINISH, LAST ;                                              00026000
  DEFINE INUDO =FOR I ←1 STEP 1 UNTIL NU DO #,INVDO =FOR I ←1 STEP 1            00027000
  UNTIL NV DO #,JNUDO =FOR J ←1 STEP 1 UNTIL NU DO #,JNVDO =FOR J ←1            00028000
  STEP 1 UNTIL NV DO #,I04DO =FOR I ←0 STEP 1 UNTIL 4 DO #,J04DO =FOR J         00029000
  ←0 STEP 1 UNTIL 4 DO #;                                                       00030000
  DEFINE J2PDO = FOR J ← 2 STEP 1 UNTIL P DO # ;                                00031000
  DEFINE POWER =                                                                00032000
  BEGIN                                                                         00033000
    IF P1 =0 THEN DOUBLE(,1.0,←,ZH,ZL)ELSE                                      00034000
    BEGIN                                                                       00035000
      DOUBLE(,2.0,←,ZH,ZL);                                                     00036000
      FOR I ←2 STEP 1 UNTIL P1 DO DOUBLE(ZH,ZL,,2.0,×,←,ZH,ZL)                  00037000
    END ;                                                                       00038000
```

111

```
                                                                                00039000
  END#;                                                                         00040000
  DEFINE DROOT =                                                                00041000
  BEGIN                                                                         00042000
    INTEGER T ;                                                                 00043000
    REAL X, Y ;                                                                 00044000
    LABEL EXIT ;                                                                00045000
    IF SUMH = 0 THEN                                                            00046000
    BEGIN                                                                       00047000
      RH ← RL ← 0 ;                                                             00048000
      GO TO EXIT ;                                                              00049000
                                                                                00050000
    END ;                                                                       00051000
    X ←SUMH ×1.0 ;                                                              00052000
    T ←0 ;                                                                      00053000
    T ←X.[8:2];                                                                 00054000
    T.[45:1]←X.[2:1];                                                           00055000
    Y ←X ;                                                                      00056000
    Y.[3:6]←Y.[2:6];                                                            00057000
    Y ←Y ×CON[T];                                                               00058000
    Y ←(X/Y +Y)×0.5 ;                                                           00059000
    Y ←(X/Y +Y)×0.5 ;                                                           00060000
    Y ←(X/Y +Y)×0.5 ;                                                           00061000
    Y ←(X/Y +Y)×0.5 ;                                                           00062000
    DOUBLE(SUMH,SUML,Y,0,/,Y,0,-,,0.5,×,Y,0,+,←,RH,RL);                         00063000
    EXIT:                                                                       00064000
  END#;                                                                         00065000
  FILE OUT PRINT 4(3,15);                                                       00066000
  FORMAT TITLE (X40,"COWELL METHOD INTEGRATION"//);                             00067000
  FORMAT CWHED ("STEP",X15,"T",X31,"X",X31,"Y",X18 ,"P ITER START"//);          00068000
  FORMAT FINAL ("INTEGRATION COMPLETED"//);                                     00069000
  FORMAT OUT CWFMT (I4,3(X2,5A6),I3,I5 //X36, 2(X2,5A6)//X36, 2(X2,5A6)/00070000
  /) ;                                                                          00071000
  FORMAT OUT STFMT (I4,3(X2,5A6),I3,I11 //X36,2(X2,5A6)//) ;                    00072000
  FORMAT OUT FMTM1 ("REAL TIME = ",F7.2,"  SEC."//);                            00073000
  FORMAT OUT FMTM2 ("PROC TIME = ",F7.2,"  SEC."//);                            00074000
  FORMAT OUT FMTM3 ("IPOP TIME = ",F7.2,"  SEC."//);                            00075000
  FORMAT OUT FOFMT ("COMPLETE ORBIT: PERIOD IS ", 5A6, " ." //) ;               00076000
  FORMAT OUT IOFMT (//"ORDER OF DIFFERENCES =",I3,X2,"DEL = ",E8.1,X2, "00077000
DIL = ",E8.1,X2, "DOL = ",E8.1//I5,3(X2,5A6)//X37,2(X2,5A6)//I5, 3(X2,  00078000
```

```
  5A6)//X37,2(X2,5A6),I11//X5, "RUN NO.",I3//), ORBFMT (//X5,"TO = ",5A600079000
  ,X2,"T = ",5A6//X5, " XO =",5A6,X2 ," X =",5A6//X5, " YO =",5A6,X2," Y00080000
 =",5A6//X5, "XOD =",5A6,X2 ,"XD =",5A6//X5, "YOD =",5A6,X2,"YD =",5A6) 00081000
  ; 00082000
  LIST CWOUT (CTR,J04DO ALPHA1[J],J04DO ALPHA2[J],J04DO ALPHA3[J],P,ITER00083000
  , J04DO ALPHA4[J], J04DO ALPHA5[J], J04DO ALPHA6[J],J04DO ALPHA7[J]); 00084000
  LIST STOUT (CTR,I04DO ALPHA1[I],I04DO ALPHA2[I],I04DO ALPHA3[I],P, 00085000
  RSCTR, I04DO ALPHA4[I], I04DO ALPHA5[I]) ; 00086000
  LIST MPOUT (J04DO ALPHA1[J]); 00087000
  LIST IOLST (Q[S],DEL,DIL,DOL,CTR-1, J04DO ALPHA1[J],J04DO ALPHA2[J], 00088000
  J04DO ALPHA3[J],J04DO ALPHA7[J], J04DO ALPHA8[J], CTR,J04DO ALPHA4[J] 00089000
  ,J04DO ALPHA5[J], J04DO ALPHA6[J] ,J04DO ALPHA9[J], J04DO ALPHA10[J], 00090000
  RSCTR, RUN), ORBLST (J04DO ALPHA1[J],J04DO ALPHA2[J],J04DO ALPHA3 [J],00091000
  J04DO ALPHA4[J],J04DO ALPHA5[J], J04DO ALPHA6[J],J04DO ALPHA7[J] , 00092000
  J04DO ALPHA8[J], J04DO ALPHA9[J],J04DO ALPHA10[J]) ; 00093000
  FORMAT CHED (//X39 , "TABLE 3" //X34 , "GAMMA COEFFICIENTS" //X35 , "(00094000
HIGH PRECISION)" //X8 , "2I", X13, "GAMMA(2I)", X23, "GAMMA*(2I)") ; 00095000
  FORMAT OUT CFMT (//X8, I2, 2(X2, 5A6)) ; 00096000
  LIST CLST (2*(I-1), J04DO ALPHA1[J], J04DO ALPHA2[J]) ; 00097000
  DEFINE DPOPG = 00098000
  BEGIN 00099000
    WRITE (PRINT[PAGE]) ; 00100000
    WRITE (PRINT,CHED) ; 00101000
    FOR I ← 1 STEP 1 UNTIL (Q[7]+2)/2 DO 00102000
    BEGIN 00103000
      DPOP(GH[1,I],GL[1,I],PRINT,ALPHA1,A) ; 00104000
      DPOP(GH[2,I],GL[2,I],PRINT,ALPHA2,A) ; 00105000
      WRITE (PRINT,CFMT,CLST) ; 00106000
 00107000
    END ; 00108000
 00109000
  END # ; 00110000
  LIST LST1 (I, J04DO ALPHA1[J], J04DO ALPHA2[J] ) , LST2 (I, J04DO 00111000
  ALPHA1[J], J04DO ALPHA2[J] ) , LST3 (I, J04DO ALPHA1[J], J04DO ALPHA2 00112000
  [J] ) , LST4 (Q[S]) ; 00113000
  FORMAT HED1 (//X35 , "FIRST ORDER CASE"//X22 , "PREDICTOR",X23,"CORREC00114000
TOR"//X9 , "I",X14,"P(I)",X28,"C(I)") , HED2 (//X34 , "SECOND ORDER CASE00115000
"//X22 , "PREDICTOR",X23,"CORRECTOR"//X9 , "I",X14,"P*(I)",X27,"C*(I)") 00116000
  , HED3 (//X38 , "MIDRANGE"//X18 , "FIRST ORDER CASE",X16,"SECOND ORDER00117000
 CASE"//X9 , "I",X14,"M(I)",X28,"M*(I)") ; 00118000
```

```
  FORMAT OUT FMT1 (//I10, 2(X2, 5A6)) , FMT2 (//I10, 2(X2, 5A6)) , FMT3 00119000
  (//I10, 2(X2, 5A6)) , FMT4 (//X38 , "TABLE 4"//X15 , "PREDICTOR-CORREC00120000
TOR AND MIDRANGE FORMULA",X1 , "COEFFICIENTS"//X22 , "CORRESPONDING TO D00121000
IFFERENCES OF ORDER ",I2) ;                                              00122000
  DEFINE DISPLAY =                                                       00123000
  BEGIN                                                                  00124000
    WRITE (PRINT[PAGE]) ;                                                00125000
    WRITE (PRINT,FMT4,LST4) ;                                            00126000
    WRITE (PRINT,HED1) ;                                                 00127000
    FOR I ← 1 STEP 1 UNTIL Q[S]+1 DO                                     00128000
    BEGIN                                                                00129000
      DPOP(PCCH[S,I,1],PCCL[S,I,1],PRINT,ALPHA1,A) ;                     00130000
      DPOP(PCCH[S,I,2],PCCL[S,I,2],PRINT,ALPHA2,A) ;                     00131000
      WRITE (PRINT,FMT1,LST1) ;                                          00132000
                                                                         00133000
    END ;                                                                00134000
    WRITE (PRINT[PAGE]) ;                                                00135000
    WRITE (PRINT,HED2) ;                                                 00136000
    FOR I ← 1 STEP 1 UNTIL Q[S]+1 DO                                     00137000
    BEGIN                                                                00138000
      DPOP(PCCH[S,I,3],PCCL[S,I,3],PRINT,ALPHA1,A) ;                     00139000
      DPOP(PCCH[S,I,4],PCCL[S,I,4],PRINT,ALPHA2,A) ;                     00140000
      WRITE (PRINT,FMT2,LST2) ;                                          00141000
                                                                         00142000
    END ;                                                                00143000
    WRITE (PRINT[PAGE]) ;                                                00144000
    WRITE (PRINT,HED3) ;                                                 00145000
    FOR I ← 1 STEP 1 UNTIL Q[S]+1 DO                                     00146000
    BEGIN                                                                00147000
      DPOP(PCCH[S,I,5],PCCL[S,I,5],PRINT,ALPHA1,A) ;                     00148000
      DPOP(PCCH[S,I,6],PCCL[S,I,6],PRINT,ALPHA2,A) ;                     00149000
      WRITE (PRINT,FMT3,LST3) ;                                          00150000
                                                                         00151000
    END ;                                                                00152000
                                                                         00153000
  END # ;                                                                00154000
  DEFINE D = DOUBLE # ;                                                  00155000
  DEFINE HOLD = FOR MASK ← 1 # ;                                         00156000
  DEFINE HACK1 = TM1 ← TIME(1) # ,HACK2 = TM2 ← TIME(2) # ,HACK3 = TM3 ←00157000
  TIME(3) # ,CHECK1 = WRITE (PRINT,FMTM1,(TIME(1)-TM1)/60) # ,CHECK2 =  00158000
```

```
WRITE (PRINT,FMTM2,(TIME(2)-TM2)/60) # ,CHECK3 = WRITE (PRINT,FMTM3,   00159000
(TIME(3)-TM3)/60) # ,CLOCKALL =                                       00160000
BEGIN                                                                 00161000
  CHECK1 ;                                                            00162000
  HACK1 ;                                                             00163000
  CHECK2 ;                                                            00164000
  HACK2 ;                                                             00165000
  CHECK3 ;                                                            00166000
  HACK3 ;                                                             00167000
                                                                      00168000
END # ;                                                               00169000
PROCEDURE DPOP(A1,A2,F,ALF,A);                                        00170000
VALUE A1,A2 ;                                                         00171000
REAL A1,A2 ;                                                          00172000
FILE F ;                                                              00173000
ALPHA ARRAY A,ALF[0];                                                 00174000
                                                                      00175000
BEGIN                                                                 00176000
  STREAM PROCEDURE PRINT(X,Y,N,F,A,ALF);                              00177000
  VALUE X,Y,N ;                                                       00178000
                                                                      00179000
  BEGIN                                                               00180000
    LOCAL V1,V11,V2,V22,V3,V4,ST,RP ;                                 00181000
    DI ←LOC V22;                                                      00182000
    SI ←LOC X ;                                                       00183000
    SI ←SI+1 ;                                                        00184000
    SKIP 3 SB ;                                                       00185000
    13(DS ←3 RESET ;                                                  00186000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                             00187000
    SKIP 1 SB));                                                      00188000
    SI ←LOC Y ;                                                       00189000
    SI ←SI+1 ;                                                        00190000
    SKIP 3 SB ;                                                       00191000
    13(DS ←3 RESET ;                                                  00192000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                             00193000
    SKIP 1 SB));                                                      00194000
    DI ←LOC RP ;                                                      00195000
    DS ←8 LIT"00000000";                                              00196000
    DI ←LOC ST ;                                                      00197000
    SI ←F ;                                                           00198000
```

```
DS ←WDS ;                                                              00199000
DI ←ST ;                                                               00200000
SI ←LOC X ;                                                            00201000
SKIP 1 SB ;                                                            00202000
IF SB THEN DS ←8 LIT"      -0."ELSE DS ←8 LIT"       0.";              00203000
3(DS ←8 LIT"00000000");                                                00204000
11(DS ←8 LIT"         ");                                              00205000
SI ←A ;                                                                00206000
V4 ←SI ;                                                               00207000
SI ←LOC V22;                                                           00208000
V3 ←SI ;                                                               00209000
26(DI ←LOC RP ;                                                        00210000
DI ←DI +7 ;                                                            00211000
DS ←CHR ;                                                              00212000
SI ←V4 ;                                                               00213000
DI ←ST ;                                                               00214000
DI ←DI+8 ;                                                             00215000
RP(DS ←24 ADD ;                                                        00216000
DI ←DI -24 ;                                                           00217000
SI ←SI -24);                                                           00218000
SI ←SI +24 ;                                                           00219000
V4 ←SI ;                                                               00220000
SI ←V3 ;                                                               00221000
SI ←SI+1 ;                                                             00222000
V3 ←SI);                                                               00223000
DI ←DI +24 ;                                                           00224000
SI ←LOC N ;                                                            00225000
SKIP 1 SB ;                                                            00226000
IF SB THEN DS ←2 LIT "@-"ELSE DS ←2 LIT "@+";                          00227000
SI ←LOC N ;                                                            00228000
DS ←2 DEC ;                                                            00229000
DI ←DI-1 ;                                                             00230000
2(DS ←RESET);                                                          00231000
DI ←ALF ;                                                              00232000
SI ←ST ;                                                               00233000
SI ←SI +5 ;                                                            00234000
DI ←DI +2 ;                                                            00235000
DS ←CHR ;                                                              00236000
SI ←SI +2 ;                                                            00237000
DS ←CHR ;                                                              00238000
```

```
    DS ←1 LIT".";                                                    00239000
    DS ←3 CHR ;                                                      00240000
    4(DI ←DI +2 ;                                                    00241000
    DS ←6 CHR);                                                      00242000
                                                                     00243000
  END ;                                                              00244000
  STREAM PROCEDURE SHIFT(A,B,C);                                     00245000
  VALUE A,B ;                                                        00246000
                                                                     00247000
  BEGIN                                                              00248000
    SI ←LOC A ;                                                      00249000
    DI ←C ;                                                          00250000
    DS ←CHR ;                                                        00251000
    SKIP 3 SB ;                                                      00252000
    DI ←DI +1 ;                                                      00253000
    36(IF SB THEN DS ←SET ELSE DS ←RESET ;                           00254000
    SKIP 1 SB);                                                      00255000
    SKIP 9 DB ;                                                      00256000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                            00257000
    SKIP 1 SB);                                                      00258000
    SI ←LOC B ;                                                      00259000
    SKIP 9 SB ;                                                      00260000
    36(IF SB THEN DS ←SET ELSE DS ←RESET ;                           00261000
    SKIP 1 SB);                                                      00262000
                                                                     00263000
  END ;                                                              00264000
  INTEGER Y,I ;                                                      00265000
  REAL T1,T2,HT ,LT;                                                 00266000
  ARRAY T[0:1];                                                      00267000
  LABEL L1,L2,L3 ;                                                   00268000
  IF A1 =0 THEN                                                      00269000
  BEGIN                                                              00270000
    T1 ←T2 ←0 ;                                                      00271000
    GO TO L2                                                         00272000
  END ;                                                              00273000
  HT ←1 ;                                                            00274000
  LT ←0 ;                                                            00275000
  Y ←0.90309 ×(IF BOOLEAN(A1.[2:1])THEN 13 -A1.[3:6]ELSE 13 +A1.[3:6])00276000
  ;                                                                  00277000
  FOR I ←1 STEP 1 UNTIL ABS(Y)DO DOUBLE(HT,LT,,10,×,←,HT,LT);        00278000
```

```
    IF Y>0 THEN DOUBLE(A1,A2,HT,LT,/,←,T1,T2)ELSE DOUBLE(A1,A2,HT,LT,×,←00279000
    ,T1,T2)};                                                          00280000
    Y ←Y-1 };                                                          00281000
    L1:IF T1.[3:6]<13 THEN                                             00282000
    BEGIN                                                              00283000
      DOUBLE(T1,T2,,10.0,/,←,T1,T2)};                                  00284000
      Y ←Y+1                                                           00285000
    END ELSE IF T1.[3:6]>13 THEN                                       00286000
    BEGIN                                                              00287000
      DOUBLE(T1,T2,,10.0,×,←,T1,T2)};                                  00288000
      Y ←Y-1                                                           00289000
    END ELSE GO TO L2 };                                               00290000
    IF T1.[3:6]=14 THEN                                                00291000
    BEGIN                                                              00292000
      SHIFT(T1,T2,T)};                                                 00293000
      PRINT(T[0],T[1],Y,F,A,ALF)};                                     00294000
      GO TO L3                                                         00295000
    END ELSE GO TO L1 };                                               00296000
    L2:PRINT(T1,T2,Y,F,A,ALF )};                                       00297000
    L3:                                                                00298000
  END };                                                               00299000
  INTEGER EVER,NEVER };                                                00300000
  INTEGER ARRAY NOTUSE1[0:80],NOTUSE2[0:9]};                           00301000
  REAL ARRAY NOTUSE3[0:68],NOTUSE4[0:68]};                             00302000
  PROCEDURE DREAD(AH,AL)};                                             00303000
  REAL AH,AL };                                                        00304000
                                                                       00305000
  BEGIN                                                                00306000
    INTEGER K,T1 };                                                    00307000
    REAL TH,TL };                                                      00308000
    BOOLEAN DF,MF };                                                   00309000
    LABEL ER,L1,L2,L3,L4,L5,L6,L7 };                                   00310000
    FORMAT OUT FR(/"DATA CARD ERROR",X105)};                           00311000
    STREAM PROCEDURE EC(S,D)};                                         00312000
                                                                       00313000
    BEGIN                                                              00314000
      SI :=S };                                                        00315000
      DI :=D };                                                        00316000
      DS :=8 LIT "0"};                                                 00317000
      2(40(DS :=7 LIT "0"};                                            00318000
```

```
    DS :=1 CHR))};                                                    00319000
                                                                      00320000
  END };                                                              00321000
  PROCEDURE GN };                                                     00322000
                                                                      00323000
  BEGIN                                                               00324000
    LABEL L1 };                                                       00325000
    IF NEVER ≤80 THEN GO TO L1 ELSE IF NEVER >81 THEN                 00326000
    BEGIN                                                             00327000
      READ (CARD,10,NOTUSE2[*])[LAST]};                               00328000
      EC(NOTUSE2,NOTUSE1)};                                           00329000
      NEVER :=1 };                                                    00330000
      L1:EVER :=NOTUSE1[NEVER]};                                      00331000
      IF EVER ="*"THEN                                                00332000
      BEGIN                                                           00333000
        NEVER :=81 };                                                 00334000
        EVER :=" "END ELSE                                            00335000
      END ELSE EVER :=" "};                                           00336000
      NEVER :=NEVER +1 };                                             00337000
                                                                      00338000
    END };                                                            00339000
    L1:GN };                                                          00340000
    IF EVER =" "THEN GO TO L1 };                                      00341000
    DF :=FALSE };                                                     00342000
    K :=0 };                                                          00343000
    TH :=0 };                                                         00344000
    TL :=0 };                                                         00345000
    IF EVER ="-"THEN MF :=TRUE ELSE                                   00346000
    BEGIN                                                             00347000
      MF :=FALSE };                                                   00348000
      IF EVER ≠"+"THEN GO TO L3 ELSE                                  00349000
    END };                                                            00350000
    L2:GN };                                                          00351000
    L3:IF EVER <10 THEN                                               00352000
    BEGIN                                                             00353000
      DF :=TRUE };                                                    00354000
      DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL)};                         00355000
      GO TO L2                                                        00356000
    END };                                                            00357000
    IF EVER =".."THEN                                                 00358000
```

```
  BEGIN                                                                    00359000
    L4:GN ;                                                                00360000
    IF EVER <10 THEN                                                       00361000
    BEGIN                                                                  00362000
      K :=K -1 ;                                                           00363000
      DF :=TRUE ;                                                          00364000
      DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL);                               00365000
      GO TO L4                                                             00366000
    END ELSE                                                               00367000
  END ;                                                                    00368000
  IF MF THEN DOUBLE(,0,TH,TL,-,:=,TH,TL);                                  00369000
  IF NOT DF THEN                                                           00370000
  BEGIN                                                                    00371000
    ER:WRITE(PRINT,FR);                                                    00372000
    GO TO LAST                                                             00373000
  END ;                                                                    00374000
  IF EVER ="" "THEN GO TO L7 ;                                             00375000
  IF EVER ≠"@"THEN GO TO ER ;                                              00376000
  GN ;                                                                     00377000
  T1 :=0 ;                                                                 00378000
  DF :=FALSE ;                                                             00379000
  IF EVER ="-"THEN MF :=TRUE ELSE                                          00380000
  BEGIN                                                                    00381000
    MF :=FALSE ;                                                           00382000
    IF EVER ≠"+"THEN GO TO L6 ELSE                                         00383000
  END ;                                                                    00384000
  L5:GN ;                                                                  00385000
  L6:IF EVER <10 THEN                                                      00386000
  BEGIN                                                                    00387000
    DF :=TRUE ;                                                            00388000
    T1 :=T1 ×10 +EVER ;                                                    00389000
    GO TO L5                                                               00390000
  END ;                                                                    00391000
  IF NOT DF THEN GO TO ER ;                                                00392000
  IF EVER ≠"" "THEN GO TO ER ;                                             00393000
  IF MF THEN K :=K -T1 ELSE K :=K +T1 ;                                    00394000
  L7:IF K <0 THEN DOUBLE(TH,TL,NOTUSE3[-K],NOTUSE4[-K],/,:=,AH,AL          00395000
  )ELSE DOUBLE(TH,TL,NOTUSE3[K],NOTUSE4[K],×,:=,AH,AL);                    00396000
                                                                           00397000
END ;                                                                      00398000
```

```
DEFINE CLEARA =                                                          00399000
BEGIN                                                                    00400000
  FOR I ← 0 STEP 1 UNTIL 25 DO FOR J ← 0 STEP 1 UNTIL 25 DO AH[I,J]      00401000
  ← AL[I,J] ← 0 ;                                                        00402000
                                                                         00403000
END # ;                                                                  00404000
DEFINE DPOPIO =                                                          00405000
BEGIN                                                                    00406000
  DPOP(TNH[M],TNL[M],PRINT,ALPHA1,A) ;                                   00407000
  DPOP(VNH[1,M],VNL[1,M],PRINT,ALPHA2,A) ;                               00408000
  DPOP(VNH[2,M],VNL[2,M],PRINT,ALPHA3,A) ;                               00409000
  DPOP(VNH[3,M],VNL[3,M],PRINT,ALPHA7,A) ;                               00410000
  DPOP(VNH[4,M],VNL[4,M],PRINT,ALPHA8,A) ;                               00411000
  DPOP(TNH[M+1],TNL[M+1],PRINT,ALPHA4,A) ;                               00412000
  DPOP(VNH[1,M+1],VNL[1,M+1],PRINT,ALPHA5,A) ;                           00413000
  DPOP(VNH[2,M+1],VNL[2,M+1],PRINT,ALPHA6,A) ;                           00414000
  DPOP(VNH[3,M+1],VNL[3,M+1],PRINT,ALPHA9 ,A) ;                          00415000
  DPOP(VNH[4,M+1],VNL[4,M+1],PRINT,ALPHA10,A) ;                          00416000
  WRITE (PRINT,IOFMT,IOLST) ;                                            00417000
  CHECK2 ;                                                               00418000
                                                                         00419000
END # ;                                                                  00420000
DEFINE DPOPORB =                                                         00421000
BEGIN                                                                    00422000
  DPOP(TOH,TOL,PRINT,ALPHA1,A) ;                                         00423000
  DPOP(TMH,TML,PRINT,ALPHA2,A) ;                                         00424000
  DPOP(UOH[1],UOL[1],PRINT,ALPHA3,A) ;                                   00425000
  DPOP(UMH[1],UML[1],PRINT,ALPHA4,A) ;                                   00426000
  DPOP(UOH[2],UOL[2],PRINT,ALPHA5,A) ;                                   00427000
  DPOP(UMH[2],UML[2],PRINT,ALPHA6,A) ;                                   00428000
  DPOP(UOH[3],UOL[3],PRINT,ALPHA7,A) ;                                   00429000
  DPOP(UMH[3],UML[3],PRINT,ALPHA8,A) ;                                   00430000
  DPOP(UOH[4],UOL[4],PRINT,ALPHA9,A) ;                                   00431000
  DPOP(UMH[4],UML[4],PRINT,ALPHA10,A) ;                                  00432000
  WRITE (PRINT, ORBFMT,ORBLST) ;                                         00433000
  CHECK2 ;                                                               00434000
                                                                         00435000
END # ;                                                                  00436000
DEFINE DPOPFO =                                                          00437000
BEGIN                                                                    00438000
```

```
    DPOP(PRDH,PRDL, PRINT, ALPHA1, A) ;                                          00439000
    WRITE (PRINT, FOFMT, MPOUT) ;                                                00440000
                                                                                 00441000
  END # ;                                                                        00442000
  DEFINE SETOUT =                                                                00443000
  BEGIN                                                                          00444000
    INTEGER I ;                                                                  00445000
    D(,0,←,TOH,TOL) ;                                                            00446000
    NV ← 2 ;                                                                     00447000
    NU ← 2×NV ;                                                                  00448000
    FOR I ← 1 STEP 1 UNTIL NU DO D(ICH[I],ICL[I],←,UOH[I],UOL[I]) ;              00449000
    D(ICH[5],ICL[5],←,MUH,MUL) ;                                                 00450000
    D(1,0,MUH,MUL,-,←,MPH,MPL) ;                                                 00451000
    D(ICH[6],ICL[6],←,PRDH,PRDL) ;                                               00452000
    S ← ENTIER((ICH[7]/2)-3) ;                                                   00453000
    DEL ← ICH[8] ;                                                               00454000
    DIL ← ICH[9] ;                                                               00455000
    DOL ← ICH[10] ;                                                              00456000
    DAL ← DEL ;                                                                  00457000
    D(TOH,TOL,←,TMH,TML) ;                                                       00458000
    FOR I ← 1 STEP 1 UNTIL NU DO D(UOH[I],UOL[I],←,UMH[I],UML[I]) ;              00459000
    FLAG ← TRUE ;                                                                00460000
    PRNT ← TRUE ;                                                                00461000
    TAB ← TRUE ;                                                                 00462000
    BL1 ← TRUE ;                                                                 00463000
    RUN ← (Q[S]-2)/4 ;                                                           00464000
    P ← 10;                                                                      00465000
                                                                                 00466000
  END # ;                                                                        00467000
  DEFINE READIN =                                                                00468000
  BEGIN                                                                          00469000
    INTEGER I, STOP ;                                                            00470000
    IF DONE THEN GO TO FINISH ;                                                  00471000
    FOR I ← 1 STEP 1 UNTIL 10 DO DREAD(ICH[I],ICL[I]) ;                          00472000
    READ (CARD,/,STOP) ;                                                         00473000
    IF STOP = 1 THEN DONE ← TRUE ELSE DONE ← FALSE ;                             00474000
    IF DONE THEN CLOSE(CARD,RELEASE);                                            00475000
    SETOUT ;                                                                     00476000
                                                                                 00477000
  END # ;                                                                        00478000
```

```
DEFINE REFINE =                                                          00479000
BEGIN                                                                    00480000
  LABEL HOME ;                                                           00481000
  IF TNH[M] ≥ PRDH THEN M ← M-1 ;                                        00482000
  IF VNH[NV,M] < 0 THEN                                                  00483000
  BEGIN                                                                  00484000
    FOR MSK ← 1 WHILE VNH[NV,M] < 0 DO M ← M-1 ;                         00485000
                                                                         00486000
  END ;                                                                  00487000
  D(TNH[M],TNL[M],←,TMH,TML) ;                                           00488000
  D(TMH,TML,←,TH,TL) ;                                                   00489000
  JNUDO                                                                  00490000
  BEGIN                                                                  00491000
    D(VNH[J,M],VNL[J,M],←,UMH[J],UML[J]) ;                               00492000
    D(UMH[J],UML[J],←,UH[J],UL[J]) ;                                     00493000
                                                                         00494000
  END ;                                                                  00495000
  IF ABS(PRDH-TH) ≤ MINH THEN GO TO HOME ELSE                            00496000
  BEGIN                                                                  00497000
    D(MINH,MINL,←,HH,HL) ;                                               00498000
    FOR MSK ← 1 WHILE TMH < (PRDH-HH) DO                                 00499000
    BEGIN                                                                00500000
      JNUDO D(UMH[J],UML[J],←,UH[J],UL[J]) ;                             00501000
      D(TMH,TML,←,TH,TL) ;                                               00502000
      D(TMH,TML,HH,HL,+,←,TMH,TML) ;                                     00503000
      RNKUT(NU,HH,HL,RKH,RKL,UMH,UML,FUNCTION) ;                         00504000
                                                                         00505000
    END ;                                                                00506000
                                                                         00507000
  END ;                                                                  00508000
  HOME: D(PRDH,PRDL,TH,TL,-,←,HH,HL) ;                                   00509000
  WH ← 1.0 ;                                                             00510000
  D(TH,TL,HH,HL,+,←,TH,TL) ;                                             00511000
  RNKUT(NU,HH,HL,RKH,RKL,UH,UL,FUNCTION) ;                               00512000
  D(TH,TL,←,TMH,TML) ;                                                   00513000
  JNUDO D(UH[J],UL[J],←,UMH[J],UML[J]) ;                                 00514000
  DPOPORB ;                                                              00515000
                                                                         00516000
END # ;                                                                  00517000
DEFINE HOMESAFE =                                                        00518000
```

```
BEGIN                                                                    00519000
  CHECK2 ;                                                               00520000
  REFINE ;                                                               00521000
  DPOPFD ;                                                               00522000
  CLOCKALL ;                                                             00523000
                                                                         00524000
END # ;                                                                  00525000
DEFINE HACKALL =                                                         00526000
BEGIN                                                                    00527000
  HACK1 ;                                                                00528000
  HACK2 ;                                                                00529000
  HACK3 ;                                                                00530000
                                                                         00531000
END # ;                                                                  00532000
DEFINE CHECKALL =                                                        00533000
BEGIN                                                                    00534000
  CHECK1 ;                                                               00535000
  CHECK2 ;                                                               00536000
  CHECK3 ;                                                               00537000
                                                                         00538000
END # ;                                                                  00539000
DEFINE PREP =                                                            00540000
BEGIN                                                                    00541000
  D(UZH[1],UZL[1], MUH, MUL, +, ←, H1,L1) ;                              00542000
  D(H1,L1, 1,0, -, ←, H2,L2) ;                                           00543000
  D(H1,L1, H1,L1, ×, UZH[2],UZL[2], UZH[2],UZL[2], ×, +, ←, SUMH,        00544000
  SUML) ;                                                                00545000
  DROOT ;                                                                00546000
  D(SUMH,SUML,RH,RL,×,←,R1H,R1L) ;                                       00547000
  D(H2,L2, H2,L2, ×, UZH[2],UZL[2], UZH[2],UZL[2], ×, +, ←, SUMH,        00548000
  SUML) ;                                                                00549000
  DROOT ;                                                                00550000
  D(SUMH,SUML,RH,RL,×,←,R2H,R2L) ;                                       00551000
  D(2,0, UZH[4],UZL[4], ×, MPH,MPL, H1,L1, ×, R1H,R1L, /,-, MUH,MUL      00552000
  ,H2,L2, ×, R2H,R2L, /, -, UZH[1],UZL[1], +, ←, F3H,F3L) ;              00553000
  D( -2,0, UZH[3],UZL[3], ×, MPH,MPL, UZH[2],UZL[2], ×, R1H,R1L, /,      00554000
  -, MUH,MUL, UZH[2],UZL[2], ×, R2H,R2L, /, -, UZH[2],UZL[2], +,←,       00555000
  F4H,F4L);                                                              00556000
                                                                         00557000
END # ;                                                                  00558000
```

```
DEFINE EVAL =                                                          00559000
BEGIN                                                                  00560000
  GO TO SW[J] ;                                                        00561000
  SW1: D(UZH[3],UZL[3], <- ZH,ZL) ;                                    00562000
  GO TO SW5 ;                                                          00563000
  SW2: D(UZH[4],UZL[4], <- ZH,ZL) ;                                    00564000
  GO TO SW5 ;                                                          00565000
  SW3: D(F3H,F3L, <- ZH,ZL) ;                                          00566000
  GO TO SW5 ;                                                          00567000
  SW4: D(F4H,F4L, <- ZH,ZL) ;                                          00568000
  GO TO SW5 ;                                                          00569000
  SW5:                                                                 00570000
END # ;                                                                00571000
PROCEDURE FUNCTION(S,UZH,UZL,ZH,ZL) ;                                  00572000
VALUE S ;                                                              00573000
INTEGER S ;                                                            00574000
REAL ZH,ZL ;                                                           00575000
REAL ARRAY UZH,UZL[0] ;                                                00576000
                                                                       00577000
BEGIN                                                                  00578000
  LABEL SW1, SW2, SW3, SW4, SW5 ;                                      00579000
  SWITCH SW <- SW1, SW2, SW3, SW4 ;                                    00580000
  INTEGER J ;                                                          00581000
  J <- S ;                                                             00582000
  IF J=3 THEN PREP ;                                                   00583000
  EVAL ;                                                               00584000
                                                                       00585000
END FUNCTION ;                                                         00586000
PROCEDURE RNKUT(NU,HH,HL,RKH,RKL,UH,UL,FUNCTION);                      00587000
VALUE NU,HH,HL ;                                                       00588000
INTEGER NU ;                                                           00589000
REAL HH,HL ;                                                           00590000
REAL ARRAY RKH,RKL,UH,UL[0];                                           00591000
PROCEDURE FUNCTION ;                                                   00592000
                                                                       00593000
BEGIN                                                                  00594000
  INTEGER I,J ;                                                        00595000
  REAL ZH, ZL ;                                                        00596000
  REAL ARRAY UZH,UZL[0:25],BZH,BZL[0:4,0:25];                          00597000
  DEFINE JNUDO =FOR J <-1 STEP 1 UNTIL NU DO #;                        00598000
```

```
    JNUDO DOUBLE(UH[J],UL[J],←,UZH[J],UZL[J]);                                  00599000
    JNUDO                                                                       00600000
    BEGIN                                                                       00601000
      FUNCTION(J,UZH,UZL,ZH,ZL) ;                                               00602000
      DOUBLE(ZH,ZL,←,BZH[1,J],BZL[1,J]) ;                                       00603000
                                                                                00604000
    END ;                                                                       00605000
    FOR I ←2 STEP 1 UNTIL 4 DO                                                  00606000
    BEGIN                                                                       00607000
      JNUDO DOUBLE(HH,HL,RKH[I],RKL[I],×,BZH[I-1,J],BZL[I-1,J],×,UH[J]00608000
      ,UL[J],+,←,UZH[J],UZL[J]);                                                00609000
      JNØDO                                                                     00610000
      BEGIN                                                                     00611000
        FUNCTION(J,UZH,UZL,ZH,ZL) ;                                             00612000
        DOUBLE(ZH,ZL,←,BZH[I,J],BZL[I,J]) ;                                     00613000
                                                                                00614000
      END ;                                                                     00615000
                                                                                00616000
    END ;                                                                       00617000
    JNUDO DOUBLE(BZH[2,J],BZL[2,J],BZH[3,J],BZL[3,J],+,2,0,×,BZH[1,J] 00618000
    ,BZL[1,J],BZH[4,J],BZL[4,J],+,+,HH,HL,×,6,0,/,UH[J],UL[J],+,←,UH[J00619000
    ],UL[J]);                                                                   00620000
                                                                                00621000
END RNKUT ;                                                                     00622000
CONTROL: TAB1 ← TRUE ;                                                          00623000
TAB2 ← TRUE ;                                                                   00624000
TAB3 ← TRUE ;                                                                   00625000
TAB4 ← TRUE ;                                                                   00626000
TAB5 ← TRUE ;                                                                   00627000
TAB6 ← TRUE ;                                                                   00628000
PRNT ← TRUE ;                                                                   00629000
DONE ← FALSE ;                                                                  00630000
HACKALL ;                                                                       00631000
GO TO ASSIGN ;                                                                  00632000
RET0: GO TO GAMMA ;                                                             00633000
RET1: READIN ;                                                                  00634000
RET2: GO TO PCCOEFF ;                                                           00635000
RET3: CHECKALL ;                                                                00636000
IF BL1 THEN GO TO COWELL ;                                                      00637000
RET4: GO TO RET1 ;                                                              00638000
```

```
    ASSIGN:CON[0]←0.0000026973988;                                          00639000
    CON[1]←0.000001603883;                                                  00640000
    CON[2]←0.0000076294;                                                    00641000
    CON[3]←0.000004536465;                                                  00642000
    CON[4]←2.137099@23;                                                     00643000
    CON[5]←1.270727@23;                                                     00644000
    CON[6]←7.555786@22;                                                     00645000
    CON[7]←4.482697@22;                                                     00646000
    FILL A[*]WITH "12500000",,"00000000","00000000",,"01562500",,"00000000"00647000
    ,,"00000000","00195312",,"50000000",,"00000000",,"00024414",,"06250000",,"00648000
00000000",,"00003051",,"75781250","00000000","00000381",,"46972656",,"25000000649000
00",,"00000047",,"68371582",,"03125000",,"00000005",,"96046447",,"75390625",,"000650000
0000000",,"74505805",,"96923828","00000000",,"09313225",,"74615479","000000000651000
0",,"01164153",,"21826935",,"00000000",,"00145519",,"15228367",,"00000000",,"0000652000
018189",,"89403546",,"00000000",,"00002273",,"73675443",,"00000000",,"000028400653000
",,"21709430",,"00000000",,"00000035",,"52713679",,"00000000",,"00000004",,"44000654000
89210","00000000",,"00000000","55511151",,"00000000",,"00000000",,"06938894"00655000
    ,,"00000000",,"00000000",,"00867362",,"00000000",,"00000000",,"00108420",,"00656000
00000000",,"00000000",,"00013553",,"00000000",,"00000000",,"00001694",,"000000006570000
00","00000000",,"00000212",,"00000000",,"00000000",,"00000026",,"00000000",,"000658000
0000000",,"00000003";                                                       00659000
                                                                            00660000
    BEGIN                                                                   00661000
      INTEGER I ;                                                           00662000
      DOUBLE(,1,:=,NOTUSE3[0],NOTUSE4[0]) ;                                 00663000
      FOR I := 1 STEP 1 UNTIL 33 DO DOUBLE(,10,NOTUSE3[I - 1],NOTUSE4[I     00664000
      - 1],×,:=,NOTUSE3[I], NOTUSE4[I]) ;                                   00665000
      FOR I := 1 STEP 1 UNTIL 35 DO DOUBLE(NOTUSE3[33],NOTUSE4[33],         00666000
      NOTUSE3[I],NOTUSE4[I],×,:=, NOTUSE3[I + 33],NOTUSE4[I + 33]) ;        00667000
      NEVER := 82 ;                                                         00668000
                                                                            00669000
    END ;                                                                   00670000
    FILL Q[*] WITH 6, 8, 10, 12, 14, 16, 18, 20 ;                           00671000
    DOUBLE(1,0,←,FH[0],FL[0]);                                              00672000
    DOUBLE(1,0,←,FH[1],FL[1]);                                              00673000
    FOR I ← 2 STEP 1 UNTIL 25 DO DOUBLE(I,0,FH[I-1],FL[I-1],×,←,FH[I],FL00674000
    [I]) ;                                                                  00675000
    DOUBLE(,3.14159265358979323846264,←,PIH,PIL);                           00676000
    DOUBLE(,0,←,RKH[1],RKL[1]);                                             00677000
    DOUBLE(,0.5,←,RKH[2],RKL[2]);                                           00678000
```

```
DOUBLE(0.5,RKH[3],RKL[3]);                                                    00679000
DOUBLE(1.0,RKH[4],RKL[4]);                                                    00680000
GO TO RET0 ;                                                                  00681000
OUTSET: GAMMA:                                                                00682000
BEGIN                                                                         00683000
  INTEGER MSK ;                                                               00684000
  LABEL PP0 ;                                                                 00685000
  N ←Q[7];                                                                    00686000
  CLEARA ;                                                                    00687000
  DOUBLE(1.0,AH[2,1],AL[2,1]);                                                00688000
  DOUBLE(0,1.0,=,AH[2,2],AL[2,2]);                                            00689000
  FOR I ←4 STEP 2 UNTIL N+2 DO                                                00690000
  BEGIN                                                                       00691000
    DOUBLE(1.0,AH[I,1],AL[I,1]);                                              00692000
    DOUBLE(0,I,0,-,2,0,/,AH[I,2],AL[I,2]);                                    00693000
    FOR J ←3 STEP 1 UNTIL I DO DOUBLE(AH[I-2,J],AL[I-2,J],AH[I-2,J-100694000
    ],AL[I-2,J-1],-,I,0,2,0 ,-,I,0,×,4,0,/,AH[I-2,J-2],AL[I-2,J-2],×00695000
    ,-,AH[I,J],AL[I,J])                                                       00696000
  END ;                                                                       00697000
  DOUBLE(1.0,GH[1,1],GL[1,1]);                                                00698000
  DOUBLE(1.0,GH[2,1],GL[2,1]);                                                00699000
  FOR I ←2 STEP 2 UNTIL N+2 DO                                                00700000
  BEGIN                                                                       00701000
    DOUBLE(0,SUMH,SUML);                                                      00702000
    FOR J ←1 STEP 1 UNTIL I DO DOUBLE(1.0,AH[I,J],AL[I,J],×,I,0,J,000703000
    ,-,2,0,+,/,SUMH,SUML,+,SUMH,SUML);                                        00704000
    DOUBLE(SUMH,SUML,FH[I],FL[I],/,GH[1,I+1],GL[1,I+1])                       00705000
  END ;                                                                       00706000
  PP0:DOUBLE(2.0,GH[1,3],GL[1,3],×,,0.25,+,GH[2,3],GL[2,3]);                  00707000
  FOR I ←4 STEP 2 UNTIL N+2 DO                                                00708000
  BEGIN                                                                       00709000
    DOUBLE(GH[1,I+1],GL[1,I+1],SUM1H,SUM1L);                                  00710000
    DOUBLE(GH[1,I-1],GL[1,I-1],SUM2H,SUM2L);                                  00711000
    IF (I/2)MOD 2 =0 THEN                                                     00712000
    BEGIN                                                                     00713000
      FOR J ←1 STEP 1 UNTIL (I-4)/4 DO                                        00714000
      BEGIN                                                                   00715000
        DOUBLE(GH[1,2×J+1],GL[1,2×J+1],GH[1,I+1-2×J],GL[1,I+1-2×J],×00716000
        ,SUM1H,SUM1L,+,SUM1H,SUM1L);                                          00717000
        DOUBLE(GH[1,2×J+1],GL[1,2×J+1],GH[1,I-1-2×J],GL[1,I-1-2×J],×00718000
```

```
      ,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                                    00719000
    END ;                                                              00720000
    DOUBLE(,2.0,SUM1H,SUM1L,×,SUM2H,SUM2L,2,0,/,+,GH[1,(I+2)/2],GL00721000
    [1,(I+2)/2],GH[1,(I+2)/2],GL[1,(I+2)/2],×,+,←,GH[2,I+1],GL[2,I00722000
    +1])                                                               00723000
  END ELSE                                                             00724000
  BEGIN                                                                00725000
    FOR J ←1 STEP 1 UNTIL (I-2)/4 DO DOUBLE(GH[1,2×J+1],GL[1,2×J+100726000
    ],GH[1,I+1-2×J],GL[1,I+1-2×J],×,SUM1H,SUM1L,+,←,SUM1H,SUM1L); 00727000
    FOR J ←1 STEP 1 UNTIL (I-6)/4 DO DOUBLE(GH[1,2×J+1],GL[1,2×J+100728000
    ],GH[1,I-1-2×J],GL[1,I-1-2×J],×,SUM2H,SUM2L,+,←,SUM2H,SUM2L); 00729000
    DOUBLE(,2.0,SUM1H,SUM1L,×,SUM2H,SUM2L,2,0,/,+,GH[1,I/2],GL[1,I00730000
    /2],GH[1,I/2],GL[1,I/2],×,4,0,/,+,←,GH[2,I+1],GL[2,I+1])       00731000
  END ;                                                                00732000
                                                                       00733000
END ;                                                                  00734000
FOR I ←1 STEP 1 UNTIL (N+4)/2 DO                                       00735000
BEGIN                                                                  00736000
  DOUBLE(GH[1,2×I-1],GL[1,2×I-1],←,GH[1,I],GL[1,I]);                   00737000
  DOUBLE(GH[2,2×I-1],GL[2,2×I-1],←,GH[2,I],GL[2,I])                    00738000
END ;                                                                  00739000
IF PRNT THEN DPOPG ;                                                   00740000
GO TO RET1 ;                                                           00741000
                                                                       00742000
END GAMMA ;                                                            00743000
PCCOEFF:                                                               00744000
BEGIN                                                                  00745000
  INTEGER MSK ;                                                        00746000
  LABEL BINOM,CLEAR,MATRX,VECTR ;                                      00747000
  LABEL PP1,PP2,PP3,PP4,PP5,PP6,PP7,PP8,PP9,PP10 ;                     00748000
  LABEL PP11,PP12,PP13,PP14,PP15 ;                                     00749000
  N ←Q[S];                                                             00750000
  SN ←1 ;                                                              00751000
  CLEARA ;                                                             00752000
  GO TO BINOM ;                                                        00753000
  PP1:GO TO CLEAR ;                                                    00754000
  PP2:DOUBLE(,1.0,←,DH[1,1],DL[1,1]);                                  00755000
  FOR J ←1 STEP 1 UNTIL N DO                                           00756000
  BEGIN                                                                00757000
    DOUBLE(,1.0,←,DH[1,J+1],DL[1,J+1]);                                00758000
```

```
  DOUBLE(J,0,1,0,+,←,DH[2,J+1],DL[2,J+1])                                    00759000
END ;                                                                        00760000
FOR I ←2 STEP 1 UNTIL N/2 DO                                                 00761000
BEGIN                                                                        00762000
  FOR J ←2×I-1 STEP 1 UNTIL N DO                                             00763000
  BEGIN                                                                      00764000
    DOUBLE(J,0,1,0,+,←,PRODH,PRODL);                                         00765000
    FOR M ←I STEP 1 UNTIL 2×I-2 DO DOUBLE(J,0,M,0,-,PRODH,PRODL,×,00766000
    ←,PRODH,PRODL);                                                          00767000
    DOUBLE(PRODH,PRODL,FH[I],FL[I],/,←,DH[I+1,J+1],DL[I+1,J+1])              00768000
  END ;                                                                      00769000
                                                                             00770000
END ;                                                                        00771000
GO TO MATRX ;                                                                00772000
PP3:GO TO VECTR ;                                                            00773000
PP4:N ←Q[S];                                                                 00774000
SN ←2 ;                                                                      00775000
GO TO CLEAR ;                                                                00776000
PP5:DOUBLE(,1,0,←,DH[1,1],DL[1,1]);                                          00777000
DOUBLE(,2,0,←,DH[2,2],DL[2,2]);                                              00778000
DOUBLE(,1,0,←,DH[2,3],DL[2,3]);                                              00779000
FOR J ←3 STEP 1 UNTIL N DO                                                   00780000
BEGIN                                                                        00781000
  DOUBLE(,1,0,←,DH[2,J+1],DL[2,J+1]);                                        00782000
  DOUBLE(J,0,1,0,-,←,DH[3,J+1],DL[3,J+1])                                    00783000
END ;                                                                        00784000
FOR I ←3 STEP 1 UNTIL N/2 DO                                                 00785000
BEGIN                                                                        00786000
  FOR J ←2×I-1 STEP 1 UNTIL N DO                                             00787000
  BEGIN                                                                      00788000
    DOUBLE(J,0,1,0,-,←,PRODH,PRODL);                                         00789000
    FOR M ←I+1 STEP 1 UNTIL 2×I-2 DO DOUBLE(J,0,M,0,-,PRODH,PRODL,00790000
    ×,←,PRODH,PRODL);                                                        00791000
    DOUBLE(PRODH,PRODL,FH[I-1],FL[I-1],/,←,DH[I+1,J+1],DL[I+1,J+1]00792000
    )                                                                        00793000
  END ;                                                                      00794000
                                                                             00795000
END ;                                                                        00796000
GO TO MATRX ;                                                                00797000
PP6:GO TO VECTR ;                                                            00798000
```

```
PP7:N ←Q[S];                                                                    00799000
SN ←3 ;                                                                         00800000
GO TO CLEAR ;                                                                   00801000
PP8:DOUBLE(␣1.0␣←␣DH[1␣1]␣DL[1␣1]);                                             00802000
DOUBLE(␣1.0␣←␣DH[1␣2]␣DL[1␣2]);                                                 00803000
FOR J ←2 STEP 1 UNTIL N DO                                                      00804000
BEGIN                                                                           00805000
  DOUBLE(␣1.0␣←␣DH[1␣J+1]␣DL[1␣J+1]);                                           00806000
  DOUBLE(J␣0␣1␣0␣=␣←␣DH[2␣J+1]␣DL[2␣J+1])                                       00807000
END ;                                                                           00808000
FOR I ←2 STEP 1 UNTIL N/2 DO                                                    00809000
BEGIN                                                                           00810000
  FOR J ←2×I STEP 1 UNTIL N DO                                                  00811000
  BEGIN                                                                         00812000
    DOUBLE(␣1.0␣←␣PRODH␣PRODL);                                                 00813000
    FOR M ←I STEP 1 UNTIL 2×I-1 DO DOUBLE(J␣0␣M␣0␣=␣PRODH␣PRODL␣×␣             00814000
    ←␣PRODH␣PRODL);                                                             00815000
    DOUBLE(PRODH␣PRODL␣FH[I]␣FL[I]␣/␣←␣DH[I+1␣J+1]␣DL[I+1␣J+1])                 00816000
  END ;                                                                         00817000
                                                                                00818000
END ;                                                                           00819000
GO TO MATRX ;                                                                   00820000
PP9:GO TO VECTR ;                                                               00821000
PP10:N ←Q[S];                                                                   00822000
SN ←4 ;                                                                         00823000
GO TO CLEAR ;                                                                   00824000
PP11:DOUBLE(␣1.0␣←␣DH[1␣1]␣DL[1␣1]);                                            00825000
DOUBLE(␣1.0␣←␣DH[2␣3]␣DL[2␣3]);                                                 00826000
DOUBLE(␣1.0␣←␣DH[2␣4]␣DL[2␣4]);                                                 00827000
FOR J ←4 STEP 1 UNTIL N DO                                                      00828000
BEGIN                                                                           00829000
  DOUBLE(␣1.0␣←␣DH[2␣J+1]␣DL[2␣J+1]);                                           00830000
  DOUBLE(J␣0␣3␣0␣=␣←␣DH[3␣J+1]␣DL[3␣J+1])                                       00831000
END ;                                                                           00832000
FOR I ←3 STEP 1 UNTIL N/2 DO                                                    00833000
BEGIN                                                                           00834000
  FOR J ←2×I STEP 1 UNTIL N DO                                                  00835000
  BEGIN                                                                         00836000
    DOUBLE(␣1.0␣←␣PRODH␣PRODL);                                                 00837000
    FOR M ←I+1 STEP 1 UNTIL 2×I-1 DO DOUBLE(J␣0␣M␣0␣=␣PRODH␣PRODL␣00838000
```

```
      x,←,PRODH,PRODL)}                                                    00839000
      DOUBLE(PRODH,PRODL,FH[I-1],FL[I-1],/,←,DH[I+1,J+1],DL[I+1,J+1]00840000
      )                                                                    00841000
    END }                                                                  00842000
                                                                           00843000
  END }                                                                    00844000
  GO TO MATRX }                                                            00845000
  PP12:GO TO VECTR }                                                       00846000
  PP13:N ←Q[S]}                                                            00847000
  SN ←2 }                                                                  00848000
  CLEARA }                                                                 00849000
  GO TO BINOM }                                                            00850000
  PP14:FOR I ←2 STEP 2 UNTIL N DO FOR J ←1 STEP 1 UNTIL I DO DOUBLE 00851000
  (AH[I,J],AL[I,J],GH[1,I/2+1],GL[1,I/2+1],x,←,AH[I,J],AL[I,J])}           00852000
  FOR I ←2 STEP 2 UNTIL N DO                                               00853000
  BEGIN                                                                    00854000
    K ←I }                                                                 00855000
    J ←1 }                                                                 00856000
    DOUBLE(AH[K,J],AL[K,J],2,0,/,←,SUMH,SUML)}                             00857000
    FOR MSK ←1 WHILE K <N DO                                               00858000
    BEGIN                                                                  00859000
      K ←K+2 }                                                             00860000
      J ←J+1 }                                                             00861000
      DOUBLE(AH[K,J],AL[K,J],2,0,/,SUMH,SUML,+,←,SUMH,SUML)                00862000
    END }                                                                  00863000
    DOUBLE(SUMH,SUML,←,PCCH[S,(N-I+2)/2,5],PCCL[S,(N-I+2)/2,5])}           00864000
    DOUBLE(,0,SUMH,SUML,-,←,PCCH[S,(N+I+2)/2,5],PCCL[S,(N+I+2)/2,5] 00865000
    )                                                                      00866000
  END }                                                                    00867000
  FOR I ←4 STEP 2 UNTIL N DO                                               00868000
  BEGIN                                                                    00869000
    K ←I }                                                                 00870000
    J ←1 }                                                                 00871000
    DOUBLE(AH[K,J],AL[K,J],2,0,/,←,SUMH,SUML)}                             00872000
    FOR MSK ←1 WHILE K <N DO                                               00873000
    BEGIN                                                                  00874000
      K ←K+2 }                                                             00875000
      J ←J+1 }                                                             00876000
      DOUBLE(AH[K,J],AL[K,J],2,0,/,SUMH,SUML,+,←,SUMH,SUML)                00877000
    END }                                                                  00878000
```

```
  DOUBLE(PCCH[S,(N-I+4)/2,5],PCCL[S,(N-I+4)/2,5],SUMH,SUML,+,←,    00879000
  PCCH[S,(N-I+4)/2,5],PCCL[S,(N-I+4)/2,5]);                        00880000
  DOUBLE(PCCH[S,(N+I)/2,5],PCCL[S,(N+I)/2,5],SUMH,SUML,-,←,PCCH[S,00881000
  (N+I)/2,5],PCCL[S,(N+I)/2,5])                                    00882000
END ;                                                              00883000
DOUBLE(,0.5,←,PCCH[S,(N+2)/2,5],PCCL[S,(N+2)/2,5]);                00884000
SN ←3 ;                                                            00885000
CLEARA ;                                                           00886000
GO TO BINOM ;                                                      00887000
PP15:FOR I ←1 STEP 2 UNTIL N+1 DO FOR J ←1 STEP 1 UNTIL I DO       00888000
DOUBLE(AH[I,J],AL[I,J],GH[2,(I+3)/2],GL[2,(I+3)/2],×,←,AH[I,J],AL 00889000
[I,J]);                                                            00890000
FOR I ←1 STEP 2 UNTIL N+1 DO                                       00891000
BEGIN                                                              00892000
  K ←I ;                                                           00893000
  J ←1 ;                                                           00894000
  DOUBLE(AH[I,J],AL[I,J],←,SUMH,SUML);                             00895000
  FOR MSK ←1 WHILE K <N+1 DO                                       00896000
  BEGIN                                                            00897000
    K ←K+2 ;                                                       00898000
    J ←J+1 ;                                                       00899000
    DOUBLE(AH[K,J],AL[K,J],SUMH,SUML,+,←,SUMH,SUML)                00900000
  END ;                                                            00901000
  DOUBLE(SUMH,SUML,←,PCCH[S,(N-I+3)/2,6],PCCL[S,(N-I+3)/2,6]);     00902000
  DOUBLE(SUMH,SUML,←,PCCH[S,(N+I+1)/2,6],PCCL[S,(N+I+1)/2,6])      00903000
END ;                                                              00904000
IF PRNT THEN DISPLAY ;                                             00905000
GO TO RET3 ;                                                       00906000
BINOM:                                                             00907000
BEGIN                                                              00908000
  LABEL SW1,SW2,SW3 ;                                              00909000
  SWITCH SW ←SW1,SW2,SW3 ;                                         00910000
  DOUBLE(,1.0,←,AH[1,1],AL[1,1]);                                  00911000
  FOR I ←2 STEP 1 UNTIL N+1 DO                                     00912000
  BEGIN                                                            00913000
    DOUBLE(,1.0,←,AH[I,1],AL[I,1]);                                00914000
    FOR J ←2 STEP 1 UNTIL I DO DOUBLE(AH[I-1,J],AL[I-1,J],AH[I-1,J00915000
    -1],AL[I-1,J-1],-,←,AH[I,J],AL[I,J])                           00916000
  END ;                                                            00917000
  GO TO SW[SN];                                                    00918000
```

```
  SW1:GO TO PP1 ;                                                    00919000
  SW2:GO TO PP14 ;                                                   00920000
  SW3:GO TO PP15 ;                                                   00921000
                                                                     00922000
END BINOM ;                                                          00923000
CLEAR:                                                               00924000
BEGIN                                                                00925000
  LABEL SW1,SW2,SW3,SW4 ;                                            00926000
  SWITCH SW ←SW1,SW2,SW3,SW4 ;                                       00927000
  FOR I ←1 STEP 1 UNTIL (N+2)/2 DO FOR J ←1 STEP 1 UNTIL (N+2)DO     00928000
  DOUBLE(,0,←,DH[I,J],DL[I,J]);                                      00929000
  GO TO SW[SN];                                                      00930000
  SW1:GO TO PP2 ;                                                    00931000
  SW2:GO TO PP5 ;                                                    00932000
  SW3:GO TO PP8 ;                                                    00933000
  SW4:GO TO PP11 ;                                                   00934000
                                                                     00935000
END CLEAR ;                                                          00936000
MATRX:                                                               00937000
BEGIN                                                                00938000
  LABEL SW1,SW2,SW3,SW4 ;                                            00939000
  SWITCH SW ←SW1,SW2,SW3,SW4 ;                                       00940000
  FOR I ←1 STEP 1 UNTIL (N+2)/2 DO FOR J ←1 STEP 1 UNTIL (N+1)DO     00941000
  BEGIN                                                              00942000
    DOUBLE(,0,←,SUMH,SUML);                                          00943000
    FOR M ←1 STEP 1 UNTIL (N+1)DO DOUBLE(DH[I,M],DL[I,M],AH[M,J],    00944000
    AL[M,J],×,SUMH,SUML,+,←,SUMH,SUML);                              00945000
    DOUBLE(SUMH,SUML,←,PCMH[I,J],PCML[I,J])                          00946000
  END ;                                                              00947000
  GO TO SW[SN];                                                      00948000
  SW1:GO TO PP3 ;                                                    00949000
  SW2:GO TO PP6 ;                                                    00950000
  SW3:GO TO PP9 ;                                                    00951000
  SW4:GO TO PP12 ;                                                   00952000
                                                                     00953000
END MATRX ;                                                          00954000
VECTR:                                                               00955000
BEGIN                                                                00956000
  LABEL SW1,SW2,SW3,SW4 ;                                            00957000
  SWITCH SW ←SW1,SW2,SW3,SW4 ;                                       00958000
```

```
    FOR J ←1 STEP 1 UNTIL (N+1)DO                                          00959000
    BEGIN                                                                  00960000
      DOUBLE(0,0,←,SUMH,SUML);                                             00961000
      IF SN <3 THEN                                                        00962000
      BEGIN                                                                00963000
        FOR I ←1 STEP 1 UNTIL (N+2)/2 DO DOUBLE(GH[1,I],GL[1,I],PCMH       00964000
        [I,J],PCML[I,J],×,SUMH,SUML,+,←,SUMH,SUML);                        00965000
        DOUBLE(SUMH,SUML,2,0,/,←,SUMH,SUML)                                00966000
      END ELSE                                                             00967000
      BEGIN                                                                00968000
        FOR I ←1 STEP 1 UNTIL (N+2)/2 DO DOUBLE(GH[2,I+1],GL[2,I+1]        00969000
        ,PCMH[I,J],PCML[I,J],×,SUMH,SUML,+,←,SUMH,SUML)                    00970000
      END ;                                                                00971000
      DOUBLE(SUMH,SUML,←,PCCH[S,J,SN],PCCL[S,J,SN])                        00972000
    END ;                                                                  00973000
    GO TO SW[SN];                                                          00974000
    SW1:GO TO PP4 ;                                                        00975000
    SW2:GO TO PP7 ;                                                        00976000
    SW3:GO TO PP10 ;                                                       00977000
    SW4:GO TO PP13 ;                                                       00978000
                                                                           00979000
  END VECTR ;                                                              00980000
                                                                           00981000
END PCCOEFF ;                                                              00982000
COWELL:                                                                    00983000
BEGIN                                                                      00984000
  INTEGER MSK1, MSK2 ;                                                     00985000
  LABEL INITIATE, INTERPOLATE2, INTEGRATE ;                                00986000
  LABEL PP16, PP17, PP18, PP19, PP20, PP21 ;                               00987000
  LABEL EXIT1, EXIT2, LAST1, LAST2 ;                                       00988000
                                                                           00989000
  BEGIN                                                                    00990000
    WRITE (PRINT[PAGE]) ;                                                  00991000
    WRITE (PRINT,TITLE) ;                                                  00992000
    WRITE (PRINT,CWHED) ;                                                  00993000
                                                                           00994000
  END ;                                                                    00995000
  SNI ← 1 ;                                                                00996000
  GO TO INITIATE ;                                                         00997000
  PP16: D(HH,HL,←,MINH,MINL) ;                                             00998000
```

```
FOR SNI ← 1 WHILE TNH[M+1] ≤ PRDH DO                                    00999000
BEGIN                                                                   01000000
  GO TO INTEGRATE ;                                                     01001000
  PP17:                                                                 01002000
END ;                                                                   01003000
GO TO PP18 ;                                                            01004000
PP18: DPOPIO ;                                                          01005000
PP20: GO TO INTERPOLATE2 ;                                              01006000
PP21: PP19: GO TO RET4 ;                                                01007000
INITIATE:                                                               01008000
BEGIN                                                                   01009000
  INTEGER MASK ;                                                        01010000
  LABEL STEPSIZE1,START1,TABLE1 ;                                       01011000
  LABEL PT1,PT2,PT3 ;                                                   01012000
  LABEL SW1,SW2 ;                                                       01013000
  SWITCH SW ←SW1,SW2 ;                                                  01014000
  DOUBLE(TMH,TML,←,TNH[1],TNL[1]);                                      01015000
  JNUDO DOUBLE(UMH[J],UML[J],←,UH[J],UL[J]);                            01016000
  L ←CTR ←RSCTR ←0 ;                                                    01017000
  M ←Q[S];                                                              01018000
  SN ←1 ;                                                               01019000
  GO TO STEPSIZE1 ;                                                     01020000
  PT1:GO TO START1 ;                                                    01021000
  PT2:GO TO TABLE1 ;                                                    01022000
  PT3:DOUBLE(Q[S],0,HH,HL,×,TNH[1],TNL[1],+,←,T1H,T1L);                 01023000
  DOUBLE(T1H,T1L,←,TH,TL);                                              01024000
  GO TO SW[SNI];                                                        01025000
  SW1:GO TO PP16 ;                                                      01026000
  SW2:GO TO PP19 ;                                                      01027000
  STEPSIZE1:                                                            01028000
  BEGIN                                                                 01029000
    INTEGER MASK ;                                                      01030000
    LABEL CYCLE ;                                                       01031000
    LABEL PP26,PP27 ;                                                   01032000
    P1 ←P ;                                                             01033000
    DOUBLE(TNH[1],TNL[1],←,T2H,T2L);                                    01034000
    JNUDO                                                               01035000
    BEGIN                                                               01036000
      DOUBLE(UH[J],UL[J],←,U1H[J],U1L[J]);                              01037000
      DOUBLE(UH[J],UL[J],←,U2H[J],U2L[J])                               01038000
```

```
END ;                                                                    01039000
SNS ←1 ;                                                                 01040000
POWER ;                                                                  01041000
DOUBLE(PRDH,PRDL,ZH,ZL,/,←,H1H,H1L);                                     01042000
GO TO CYCLE ;                                                            01043000
PP26:SNS ←2 ;                                                            01044000
FOR MASK ←1 WHILE ABS(Z3H)≥DEL DO                                        01045000
BEGIN                                                                    01046000
  P1 ←P1+1 ;                                                             01047000
  D(H2H,H2L,←,H1H,H1L) ;                                                 01048000
  JNUDO                                                                  01049000
  BEGIN                                                                  01050000
    DOUBLE(UH[J],UL[J],←,U1H[J],U1L[J]);                                 01051000
    DOUBLE(UH[J],UL[J],←,U2H[J],U2L[J])                                  01052000
  END ;                                                                  01053000
  GO TO CYCLE ;                                                          01054000
  PP27:                                                                  01055000
END ;                                                                    01056000
P ←P1 ;                                                                  01057000
DOUBLE(H1H,H1L,←,HH,HL);                                                 01058000
GO TO PT1 ;                                                              01059000
CYCLE:                                                                   01060000
BEGIN                                                                    01061000
  INTEGER K ;                                                            01062000
  REAL ZH,ZL,TZH,TZL ;                                                   01063000
  LABEL CY1,CY2,CY3,RNKUT1,RNKUT2,RADICAL ;                              01064000
  LABEL SW1,SW2 ;                                                        01065000
  SWITCH SW ←SW1,SW2 ;                                                   01066000
  DOUBLE(H1H,H1L,2,0,/,←,H2H,H2L);                                       01067000
  RNKUT(NU,H1H,H1L,RKH,RKL,U1H,U1L,FUNCTION) ;                           01068000
  CY1:FOR K ←1 STEP 1 UNTIL 2 DO                                         01069000
  BEGIN                                                                  01070000
    DOUBLE(K,0,1,0,=,H2H,H2L,×,T2H,T2L,+,←,TZH,TZL);                     01071000
    RNKUT(NU,H2H,H2L,RKH,RKL,U2H,U2L,FUNCTION) ;                         01072000
    CY2:                                                                 01073000
  END ;                                                                  01074000
  GO TO RADICAL ;                                                        01075000
  CY3: GO TO SW[SNS];                                                    01076000
  SW1:GO TO PP26 ;                                                       01077000
  SW2:GO TO PP27 ;                                                       01078000
```

```
      RADICAL:                                                          01079000
      BEGIN                                                             01080000
        INTEGER J ;                                                     01081000
        REAL SUMH,SUML,RH,RL ;                                          01082000
        SUMH ←SUML ←0 ;                                                 01083000
        FOR J ← 1 STEP 1 UNTIL NU DO                                    01084000
        BEGIN                                                           01085000
          DOUBLE(U2H[J],U2L[J],U1H[J],U1L[J],-,←,Z2H,Z2L) ;             01086000
          DOUBLE(Z2H,Z2L,Z2H,Z2L,×,SUMH,SUML,+,←,SUMH,SUML) ;           01087000
                                                                        01088000
        END ;                                                           01089000
        DROOT ;                                                         01090000
        DOUBLE(RH,RL,←,Z3H,Z3L) ;                                       01091000
        GO TO CY3 ;                                                     01092000
                                                                        01093000
      END RADICAL;                                                      01094000
                                                                        01095000
    END CYCLE ;                                                         01096000
                                                                        01097000
  END STEPSIZE1 ;                                                       01098000
  START1:                                                               01099000
  BEGIN                                                                 01100000
    DOUBLE(TNH[1],TNL[1],←,TZH,TZL);                                    01101000
    JNUDO DOUBLE(UH[J],UL[J],←,VNH[J,1],VNL[J,1]);                      01102000
    INVDO                                                               01103000
    BEGIN                                                               01104000
      K ← I + NV ;                                                      01105000
      FUNCTION(K,UH,UL,ZH,ZL) ;                                         01106000
      DOUBLE(ZH,ZL,←,FNH[I,1],FNL[I,1]) ;                               01107000
                                                                        01108000
    END ;                                                               01109000
    FOR J ←1 STEP 1 UNTIL Q[S]DO                                        01110000
    BEGIN                                                               01111000
      DOUBLE(J,0,1,0,-,HH,HL,×,TNH[1],TNL[1],+,←,TZH,TZL);              01112000
      DOUBLE(J,0,HH,HL,×,TNH[1],TNL[1],+,←,TNH[J+1],TNL[J+1]);          01113000
      RNKUT(NU,HH,HL,RKH,RKL,UH,UL,FUNCTION);                           01114000
      INUDO DOUBLE(UH[I],UL[I],←,VNH[I,J+1],VNL[I,J+1]);                01115000
      INVDO                                                             01116000
      BEGIN                                                             01117000
        K ← I + NV ;                                                    01118000
```

```
      FUNCTION(K,UH,UL,ZH,ZL) ;                                          01119000
      DOUBLE(ZH,ZL,←,FNH[I,J+1],FNL[I,J+1]) ;                            01120000
                                                                         01121000
    END ;                                                                01122000
                                                                         01123000
  END ;                                                                  01124000
  FLAG ← TRUE ;                                                          01125000
  CTR ←CTR+Q[S]  ;                                                       01126000
  GO TO PT2 ;                                                            01127000
                                                                         01128000
END START1 ;                                                             01129000
TABLE1;                                                                  01130000
BEGIN                                                                    01131000
  N ←Q[S];                                                               01132000
  INVDO                                                                  01133000
  BEGIN                                                                  01134000
    DOUBLE(,0,←,SUM1H,SUM1L);                                            01135000
    DOUBLE(,0,←,SUM2H,SUM2L);                                            01136000
    FOR J ←0 STEP 1 UNTIL N DO                                           01137000
    BEGIN                                                                01138000
      DOUBLE(PCCH[S,J+1,5],PCCL[S,J+1,5],FNH[I,N-J+1],FNL[I,N-J          01139000
      +1],×,SUM1H,SUM1L,+,←,SUM1H,SUM1L);                                01140000
      DOUBLE(PCCH[S,J+1,6],PCCL[S,J+1,6],FNH[I,N-J+1],FNL[I,N-J          01141000
      +1],×,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                                 01142000
    END ;                                                                01143000
    DOUBLE(VNH[I+NV,N/2+1],VNL[I+NV,N/2+1],HH,HL,/,SUM1H,SUM1L,-         01144000
    ,←,FRSTH[I,N/2+1],FRSTL[I,N/2+1]);                                   01145000
    DOUBLE(VNH[I,N/2+1],VNL[I,N/2+1],HH,HL,/,HH,HL,/,SUM2H ,             01146000
    SUM2L,-,←,SCNDH[I,N/2+1],SCNDL[I,N/2+1]);                            01147000
    DOUBLE(FRSTH[I,(N+2)/2],FRSTL[I,(N+2)/2],FNH[I,(N+2)/2],FNL          01148000
    [I,(N+2)/2],+,←,FRSTH[I,(N+4)/2],FRSTL[I,(N+4)/2]);                  01149000
    FOR J ←(N+6)/2 STEP 1 UNTIL N+2 DO DOUBLE(FRSTH[I,J-1],FRSTL         01150000
    [I,J-1],FNH[I,J-1],FNL[I,J-1],+,←,FRSTH[I,J],FRSTL[I,J]);            01151000
    FOR J ←N/2 STEP -1 UNTIL 1 DO DOUBLE(FRSTH[I,J+1],FRSTL[I,J          01152000
    +1],FNH[I,J],FNL[I,J],-,←,FRSTH[I,J],FRSTL[I,J]);                    01153000
    FOR J ←(N+4)/2 STEP 1 UNTIL N+2 DO DOUBLE(SCNDH[I,J-1],SCNDL         01154000
    [I,J-1],FRSTH[I,J],FRSTL[I,J],+,←,SCNDH[I,J],SCNDL[I,J]);            01155000
    FOR J ←N/2 STEP -1 UNTIL 1 DO DOUBLE(SCNDH[I,J+1],SCNDL[I,J          01156000
    +1],FRSTH[I,J+1],FRSTL[I,J+1],-,←,SCNDH[I,J],SCNDL[I,J]);            01157000
                                                                         01158000
```

```
    END ;                                                                    01159000
    GO TO PT3 ;                                                              01160000
                                                                             01161000
  END TABLE1 ;                                                               01162000
                                                                             01163000
END INITIATE ;                                                               01164000
INTEGRATE:                                                                   01165000
BEGIN                                                                        01166000
  INTEGER MASK ;                                                             01167000
  LABEL PREDICT,CORRECT,TEST ;                                               01168000
  LABEL PP21,PP22,PP23,AHEAD ;                                               01169000
  LABEL SW1,SW2 ;                                                            01170000
  SWITCH SW ←SW1,SW2 ;                                                       01171000
  NT←Q[S]/2 ;                                                                01172000
  CTR ←CTR+1 ;                                                               01173000
  L ←L+1 ;                                                                   01174000
  M ←M+1 ;                                                                   01175000
  DOUBLE(L,0,HH,HL,×,T1H,T1L,+,←,TNH[M+1],TNL[M+1]);                         01176000
  DOUBLE(L,0,1,0,-,HH,HL,×,T1H,T1L,+,←,TH,TL);                               01177000
  GO TO PREDICT ;                                                            01178000
  PP21:SN ←1 ;                                                               01179000
  GO TO CORRECT ;                                                            01180000
  PP22:SN ←2 ;                                                               01181000
  ITER ←1 ;                                                                  01182000
  FOR MASK ←1 WHILE ABS(Z3H)≥DAL DO                                          01183000
  BEGIN                                                                      01184000
    ITER ←ITER+1 ;                                                           01185000
    FOR J ← 1 STEP 1 UNTIL NV DO DOUBLE(U2H[J],U2L[J],←,U1H[J],U1L01186000
    [J]) ;                                                                   01187000
    GO TO CORRECT ;                                                          01188000
    PP23:                                                                    01189000
  END ;                                                                      01190000
  TAB6 ← FALSE ;                                                             01191000
  IF TAB THEN                                                                01191010
  BEGIN                                                                      01191020
    DPOP(TNH[M+1],TNL[M+1],PRINT,ALPHA1,A);                                  01191040
    DPOP(VNH[1,M+1],VNL[1,M+1],PRINT,ALPHA2,A);                              01191050
    DPOP(VNH[2,M+1],VNL[2,M+1],PRINT,ALPHA3,A);                              01191060
    DPOP(VNH[3,M+1],VNL[3,M+1],PRINT,ALPHA4,A) ;                             01191070
    DPOP(VNH[4,M+1],VNL[4,M+1],PRINT,ALPHA5,A) ;                             01191080
```

```
    DPOP(FNH[1,M+1],FNL[1,M+1],PRINT,ALPHA6,A) ;                    01191090
    DPOP(FNH[2,M+1],FNL[2,M+1],PRINT,ALPHA7,A) ;                    01191100
    WRITE (PRINT,CWFMT,CWOUT);                                      01191110
    CHECK2 ;                                                        01191120
                                                                    01191130
  END ;                                                             01191140
  JNVDO                                                             01192000
  BEGIN                                                             01193000
    DOUBLE(FRSTH[J,M+1],FRSTL[J,M+1],FNH[J,M+1],FNL[J,M+1],+,←      01194000
    FRSTH[J,M+2],FRSTL[J,M+2]);                                     01195000
    DOUBLE(SCNDH[J,M+1],SCNDL[J,M+1],FRSTH[J,M+2],FRSTL[J,M+2],+,←01196000
    ,SCNDH[J,M+2],SCNDL[J,M+2])                                     01197000
  END ;                                                             01198000
  GO TO TEST ;                                                      01199000
  IF L MOD NT ≠0 THEN GO TO AHEAD ELSE GO TO TEST ;                 01200000
  AHEAD:GO TO SW[SNI];                                              01201000
  SW1:GO TO PP17 ;                                                  01202000
  SW2:GO TO PP20 ;                                                  01203000
  PREDICT:                                                          01204000
  BEGIN                                                             01205000
    LABEL PR1,RADICAL ;                                             01206000
    INVDO                                                           01207000
    BEGIN                                                           01208000
      DOUBLE(FRSTH[I,M+1],FRSTL[I,M+1],←,SUM1H,SUM1L);              01209000
      DOUBLE(SCNDH[I,M+1],SCNDL[I,M+1],←,SUM2H,SUM2L);              01210000
      FOR J ←0 STEP 1 UNTIL Q[S]DO                                  01211000
      BEGIN                                                         01212000
        DOUBLE(PCCH[S,J+1,1],PCCL[S,J+1,1],FNH[I,M-J],FNL[I,M-J],×01213000
        ,SUM1H,SUM1L,+,←,SUM1H,SUM1L);                              01214000
        DOUBLE(PCCH[S,J+1,3],PCCL[S,J+1,3],FNH[I,M-J],FNL[I,M-J],×01215000
        ,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                               01216000
      END ;                                                         01217000
      DOUBLE(SUM1H,SUM1L,HH,HL,×,←,VNH[I+NV,M+1],VNL[I+NV,M+1]) ;  01218000
      DOUBLE(SUM2H,SUM2L,HH,HL,×,HH,HL,×,←,VNH[I,M+1],VNL[I,M+1]);01219000
      DOUBLE(VNH[I,M+1],VNL[I,M+1],←,UZH[I],UZL[I]);                01220000
      DOUBLE(VNH[I+NV,M+1],VNL[I+NV,M+1],←,UZH[I+NV] ,UZL[I+NV] ) 01221000
                                                                    01222000
    END ;                                                           01223000
    INVDO                                                           01224000
    BEGIN                                                           01225000
```

```
      K ← I + NV ;                                                    01226000
      FUNCTION(K,UZH,UZL,ZH,ZL) ;                                     01227000
      DOUBLE(ZH,ZL,←,FNH[I,M+1],FNL[I,M+1]) ;                         01228000
                                                                      01229000
    END ;                                                             01230000
    INVDO DOUBLE(FNH[I,M+1],FNL[I,M+1],←,U1H[I],U1L[I]);              01231000
    PR1:GO TO PP21 ;                                                  01232000
    RADICAL:                                                          01233000
    BEGIN                                                             01234000
      INTEGER J ;                                                     01235000
      REAL SUMH,SUML,RH,RL ;                                          01236000
      SUMH ←SUML ←0 ;                                                 01237000
      FOR J ←1 STEP 1 UNTIL NV DO DOUBLE(UZH[J],UZL[J],UZH[J],UZL     01238000
      [J],×,SUMH,SUML,+,←,SUMH,SUML);                                 01239000
      DROOT ;                                                         01240000
      DOUBLE(RH,RL,←,Z1H,Z1L);                                        01241000
      GO TO PR1 ;                                                     01242000
                                                                      01243000
    END RADICAL;                                                      01244000
                                                                      01245000
  END PREDICT ;                                                       01246000
  CORRECT:                                                            01247000
  BEGIN                                                               01248000
    INTEGER MASK ;                                                    01249000
    LABEL CR1,RADICAL ;                                               01250000
    LABEL SW1,SW2 ;                                                   01251000
    SWITCH SW ←SW1,SW2 ;                                              01252000
    INVDO                                                             01253000
    BEGIN                                                             01254000
      DOUBLE(FRSTH[I,M+1],FRSTL[I,M+1],←,SUM1H,SUM1L);                01255000
      DOUBLE(SCNDH[I,M+1],SCNDL[I,M+1],←,SUM2H,SUM2L);                01256000
      FOR J ←0 STEP 1 UNTIL Q[S]DO                                    01257000
      BEGIN                                                           01258000
        ;                                                             01259000
        DOUBLE(PCCH[S,J+1,2],PCCL[S,J+1,2],FNH[I,M+1-J],FNL[I,M+1     01260000
        -J],×,SUM1H,SUM1L,+,←,SUM1H,SUM1L);                           01261000
        DOUBLE(PCCH[S,J+1,4],PCCL[S,J+1,4],FNH[I,M+1-J],FNL[I,M+1     01262000
        -J],×,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                            01263000
      END ;                                                           01264000
      DOUBLE(SUM1H,SUM1L,HH,HL,×,←,VNH[I+NV,M+1],VNL[I+NV,M+1]) ;     01265000
```

```
      DOUBLE(SUM2H,SUM2L,HH,HL,×,HH,HL,×,←,VNH[I,M+1],VNL[I,M+1]);01266000
      DOUBLE(VNH[I,M+1],VNL[I,M+1],←,UZH[I],UZL[I]);                01267000
      DOUBLE(VNH[I+NV,M+1],VNL[I+NV,M+1],←,UZH[I+NV],UZL[I+NV])     01268000
    END ;                                                           01269000
    INVDO                                                           01270000
    BEGIN                                                           01271000
      K ← I + NV ;                                                  01272000
      FUNCTION(K,UZH,UZL,ZH,ZL) ;                                   01273000
      DOUBLE(ZH,ZL,←,FNH[I,M+1],FNL[I,M+1]) ;                       01274000
                                                                    01275000
    END ;                                                           01276000
    INVDO DOUBLE(FNH[I,M+1],FNL[I,M+1],←,U2H[I],U2L[I]);            01277000
    GO TO RADICAL ;                                                 01278000
    CR1: GO TO SW[SN];                                              01279000
    SW1:GO TO PP22 ;                                                01280000
    SW2:GO TO PP23 ;                                                01281000
    RADICAL:                                                        01282000
    BEGIN                                                           01283000
      INTEGER J ;                                                   01284000
      REAL SUMH,SUML,RH,RL ;                                        01285000
      SUMH ←SUML ←0 ;                                               01286000
      FOR J ← 1 STEP 1 UNTIL NV DO                                  01287000
      BEGIN                                                         01288000
        DOUBLE(U2H[J],U2L[J],U1H[J],U1L[J],-,←,Z2H,Z2L) ;           01289000
        DOUBLE(Z2H,Z2L,Z2H,Z2L,×,SUMH,SUML,+,←,SUMH,SUML) ;         01290000
                                                                    01291000
      END ;                                                         01292000
      DROOT ;                                                       01293000
      DOUBLE(RH,RL,←,Z3H,Z3L) ;                                     01294000
      GO TO CR1 ;                                                   01295000
                                                                    01296000
    END RADICAL;                                                    01297000
                                                                    01298000
  END CORRECT ;                                                     01299000
  TEST:                                                             01300000
  BEGIN                                                             01301000
    INTEGER NZ ;                                                    01302000
    LABEL BYPASS ;                                                  01303000
    REAL Z ;                                                        01304000
    LABEL TS1,RADICAL ;                                             01305000
```

```
LABEL SHIFT1,SHIFT2,RESTART ;                                          01306000
NZ ←M+1-Q[S]/2 ;                                                       01307000
INVDO                                                                  01308000
BEGIN                                                                  01309000
  DOUBLE(SCNDH[I,NZ],SCNDL[I,NZ],←,SUMH,SUML);                         01310000
  FOR J ←0 STEP 1 UNTIL Q[S]DO DOUBLE(PCCH[S,J+1,6],PCCL[S,J+1         01311000
  ,6],FNH[I,M+1-J],FNL[I,M+1-J],×,SUMH,SUML,+,←,SUMH,SUML);            01312000
  DOUBLE(HH,HL,HH,HL,×,SUMH,SUML,×,←,TSH[I],TSL[I]);                   01313000
  DOUBLE(VNH[I,NZ],VNL[I,NZ],←,UZH[I],UZL[I])                          01314000
END ;                                                                  01315000
GO TO RADICAL ;                                                        01316000
TS1: Z ←ABS(Z3H);                                                      01317000
IF (Z <DIL)AND (L MOD Q[S]≠0)THEN GO TO AHEAD ELSE IF (Z <DIL          01318000
)AND (L MOD Q[S]=0)THEN GO TO SHIFT2 ELSE IF (DIL ≤Z)AND (Z ≤          01319000
DOL)AND (L MOD Q[S]≠0)THEN GO TO AHEAD ELSE IF (DIL ≤Z)AND (Z          01320000
≤ DOL)AND (L MOD Q[S]=0)THEN GO TO SHIFT1 ELSE                         01321000
BEGIN                                                                  01322000
  CTR ←CTR-Q[S]/2 ;                                                    01323000
  RSCTR ←RSCTR+1 ;                                                     01324000
  GO TO RESTART ;                                                      01325000
                                                                       01326000
END ;                                                                  01327000
BYPASS: IF (ITER = 1) AND (L MOD Q[S] ≠ 0) THEN GO TO AHEAD            01328000
ELSE IF (ITER = 1) AND (L MOD Q[S] = 0) THEN GO TO SHIFT2 ELSE         01329000
IF (ITER = 2) AND (L MOD Q[S] ≠ 0) THEN GO TO AHEAD ELSE IF            01330000
(ITER = 2) AND (L MOD Q[S] = 0) THEN GO TO SHIFT1 ELSE IF              01331000
(ITER = 3) AND (L MOD Q[S] ≠ 0) THEN GO TO AHEAD ELSE IF (ITER         01332000
= 3) AND (L MOD Q[S] = 0) THEN GO TO SHIFT1 ELSE CTR ← CTR-1 ;         01333000
RSCTR ← RSCTR+1 ;                                                      01334000
GO TO RESTART ;                                                        01335000
RADICAL:                                                               01336000
BEGIN                                                                  01337000
  INTEGER J ;                                                          01338000
  REAL SUMH,SUML,RH,RL ;                                               01339000
  SUMH ←SUML ←0 ;                                                      01340000
  FOR J ← 1 STEP 1 UNTIL NV DO                                         01341000
  BEGIN                                                                01342000
    DOUBLE(UZH[J],UZL[J],TSH[J],TSL[J],-,←,Z2H,Z2L) ;                  01343000
    DOUBLE(Z2H,Z2L,Z2H,Z2L,×,SUMH,SUML,+,←,SUMH,SUML) ;                01344000
                                                                       01345000
```

```
    END ;                                                              01346000
    DROOT ;                                                            01347000
    DOUBLE(RH,RL,←,Z3H,Z3L) ;                                          01348000
    GO TO TS1 ;                                                        01349000
                                                                       01350000
  END RADICAL;                                                         01351000
  SHIFT1:                                                              01352000
  BEGIN                                                                01353000
    INTEGER MASK ;                                                     01354000
    LABEL TABLE2 ;                                                     01355000
    DEFINE NZ =M+1+J-Q[S]#;                                            01356000
    FLAG ← FALSE ;                                                     01357000
    FOR J ←0 STEP 1 UNTIL Q[S]DO                                       01358000
    BEGIN                                                              01359000
      INUDO DOUBLE(VNH[I,NZ],VNL[I,NZ],←,VNH[I,J+1],VNL[I,J+1]);       01360000
      INVDO DOUBLE(FNH[I,NZ],FNL[I,NZ],←,FNH[I,J+1],FNL[I,J+1]);       01361000
      INVDO                                                            01362000
      BEGIN                                                            01363000
        D(FRSTH[I,NZ],FRSTL[I,NZ],←,FRSTH[I,J+1],FRSTL[I,J+1]) ;       01364000
        D(SCNDH[I,NZ],SCNDL[I,NZ],←,SCNDH[I,J+1],SCNDL[I,J+1]) ;       01365000
                                                                       01366000
      END ;                                                            01367000
      DOUBLE(TNH[NZ],TNL[NZ],←,TNH[J+1],TNL[J+1])                      01368000
    END ;                                                              01369000
    INVDO                                                              01370000
    BEGIN                                                              01371000
      D(FRSTH[I,M+2],FRSTL[I,M+2],←,FRSTH[I,Q[S]+2],FRSTL[I,Q[S]       01372000
      +2]) ;                                                           01373000
      D(SCNDH[I,M+2],SCNDL[I,M+2],←,SCNDH[I,Q[S]+2],SCNDL[I,Q[S]       01374000
      +2]) ;                                                           01375000
                                                                       01376000
    END ;                                                              01377000
    L ←0 ;                                                             01378000
    M ←Q[S];                                                           01379000
    DOUBLE(Q[S],0,HH,HL,×,TNH[1],TNL[1],+,←,T1H,T1L);                  01380000
    DOUBLE(T1H,T1L,←,TH,TL);                                           01381000
    GO TO AHEAD ;                                                      01382000
                                                                       01383000
  END SHIFT1 ;                                                         01384000
  SHIFT2:                                                              01385000
```

```
BEGIN                                                                    01386000
  INTEGER MASK ;                                                         01387000
  REAL ZH,ZL ;                                                           01388000
  LABEL TABLE3 ;                                                         01389000
  FLAG ← FALSE ;                                                         01390000
  P1 ←P ←P-1 ;                                                           01391000
  D(2,0,HH,HL,×,←,HH,HL) ;                                               01392000
  FOR J ←0 STEP 1 UNTIL Q[S]DO                                           01393000
  BEGIN                                                                  01394000
    INUDO DOUBLE(VNH[I,2×J+1],VNL[I,2×J+1],←,VNH[I,J+1],VNL[I            01395000
    ,J+1]);                                                              01396000
    INVDO DOUBLE(FNH[I,2×J+1],FNL[I,2×J+1],←,FNH[I,J+1],FNL[I            01397000
    ,J+1]);                                                              01398000
    DOUBLE(TNH[2×J+1],TNL[2×J+1],←,TNH[J+1],TNL[J+1])                    01399000
  END ;                                                                  01400000
  L ←0 ;                                                                 01401000
  M ←Q[S];                                                               01402000
  DOUBLE(Q[S],0,HH,HL,×,TNH[1],TNL[1],+,←,T1H,T1L);                      01403000
  DOUBLE(T1H,T1L,←,TH,TL);                                               01404000
  GO TO TABLE3 ;                                                         01405000
  TABLE3:                                                                01406000
  BEGIN                                                                  01407000
    N ←Q[S];                                                             01408000
    INVDO                                                                01409000
    BEGIN                                                                01410000
      DOUBLE(,0,←,SUM1H,SUM1L);                                          01411000
      DOUBLE(,0,←,SUM2H,SUM2L);                                          01412000
      FOR J ←0 STEP 1 UNTIL N DO                                         01413000
      BEGIN                                                              01414000
        DOUBLE(PCCH[S,J+1,5],PCCL[S,J+1,5],FNH[I,N-J+1],FNL[I            01415000
        ,N-J+1],×,SUM1H,SUM1L,+,←,SUM1H,SUM1L);                          01416000
        DOUBLE(PCCH[S,J+1,6],PCCL[S,J+1,6],FNH[I,N-J+1],FNL[I            01417000
        ,N-J+1],×,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                           01418000
      END ;                                                              01419000
      DOUBLE(VNH[I+NV,N/2+1],VNL[I+NV,N/2+1],HH,HL,/,SUM1H,              01420000
      SUM1L,-,←,FRSTH[I,N/2+1],FRSTL[I,N/2+1]);                          01421000
      DOUBLE(VNH[I,N/2+1],VNL[I,N/2+1],HH,HL,/,HH,HL,/,SUM2H             01422000
      ,SUM2L,-,←,SCNDH[I,N/2+1],SCNDL[I,N/2+1]);                         01423000
      DOUBLE(FRSTH[I,(N+2)/2],FRSTL[I,(N+2)/2],FNH[I,(N+2)/2]            01424000
      ,FNL[I,(N+2)/2],+,←,FRSTH[I,(N+4)/2],FRSTL[I,(N+4)/2]);            01425000
```

```
      FOR J ←(N+6)/2 STEP 1 UNTIL N+2 DO DOUBLE(FRSTH[I,J-1], 01426000
      FRSTL[I,J-1],FNH[I,J-1],FNL[I,J-1],+,←,FRSTH[I,J],FRSTL 01427000
      [I,J]); 01428000
      FOR J ←N/2 STEP -1 UNTIL 1 DO DOUBLE(FRSTH[I,J+1],FRSTL 01429000
      [I,J+1],FNH[I,J],FNL[I,J],-,←,FRSTH[I,J],FRSTL[I,J]); 01430000
      FOR J ←(N+4)/2 STEP 1 UNTIL N+2 DO DOUBLE(SCNDH[I,J-1], 01431000
      SCNDL[I,J-1],FRSTH[I,J],FRSTL[I,J],+,←,SCNDH[I,J],SCNDL 01432000
      [I,J]); 01433000
      FOR J ←N/2 STEP -1 UNTIL 1 DO DOUBLE(SCNDH[I,J+1],SCNDL 01434000
      [I,J+1],FRSTH[I,J+1],FRSTL[I,J+1],-,←,SCNDH[I,J],SCNDL[I01435000
      ,J]); 01436000
                                                              01437000
    END ; 01438000
    GO TO AHEAD ; 01439000
                                                              01440000
  END TABLE3 ; 01441000
                                                              01442000
END SHIFT2 ; 01443000
RESTART: 01444000
BEGIN 01445000
  INTEGER NZ ; 01446000
  LABEL BYPASS ; 01447000
  REAL ZH,ZL ; 01448000
  LABEL STEPSIZE2,START2,TABLE4 ; 01449000
  LABEL PP24,PP25 ; 01450000
  NZ ← M ; 01451000
  GO TO BYPASS ; 01452000
  IF FLAG THEN 01453000
  BEGIN 01454000
    CTR ← CTR-Q[S] ; 01455000
    NZ ← NZ-Q[S] ; 01456000
                                                              01457000
  END ; 01458000
  BYPASS: P1 ← P ← P+1 ; 01459000
  DOUBLE(TNH[NZ],TNL[NZ],←,TNH[1],TNL[1]); 01460000
  JNUDO DOUBLE(VNH[J,NZ],VNL[J,NZ],←,UH[J],UL[J]); 01461000
  SN ←2 ; 01462000
  GO TO STEPSIZE2; 01463000
  PP24:GO TO START2 ; 01464000
  PP25:L ←0 ; 01465000
```

```
M ←Q[S];                                                          01466000
DOUBLE(Q[S],0,HH,HL,×,TNH[1],TNL[1],+,←,T1H,T1L);                 01467000
DOUBLE(T1H,T1L,←,TH,TL);                                          01468000
GO TO TABLE4 ;                                                    01469000
STEPSIZE2:                                                        01470000
BEGIN                                                             01471000
  INTEGER MASK ;                                                  01472000
  LABEL CYCLE ;                                                   01473000
  LABEL PP26,PP27 ;                                               01474000
  P1 ←P ;                                                         01475000
  DOUBLE(TNH[1],TNL[1],←,T2H,T2L);                                01476000
  JNUDO                                                           01477000
  BEGIN                                                           01478000
    DOUBLE(UH[J],UL[J],←,U1H[J],U1L[J]);                          01479000
    DOUBLE(UH[J],UL[J],←,U2H[J],U2L[J])                           01480000
  END ;                                                           01481000
  SNS ←1 ;                                                        01482000
  POWER ;                                                         01483000
  DOUBLE(PRDH,PRDL,ZH,ZL,/,←,H1H,H1L) ;                           01484000
  GO TO CYCLE ;                                                   01485000
  PP26:SNS ←2 ;                                                   01486000
  FOR MASK ←1 WHILE ABS(Z3H)≥DEL DO                               01487000
  BEGIN                                                           01488000
    P1 ←P1+1 ;                                                    01489000
    D(H2H,H2L,←,H1H,H1L) ;                                        01490000
    JNUDO                                                         01491000
    BEGIN                                                         01492000
      DOUBLE(UH[J],UL[J],←,U1H[J],U1L[J]);                        01493000
      DOUBLE(UH[J],UL[J],←,U2H[J],U2L[J])                         01494000
    END ;                                                         01495000
    GO TO CYCLE ;                                                 01496000
    PP27:                                                         01497000
  END ;                                                           01498000
  P ←P1 ;                                                         01499000
  DOUBLE(H1H,H1L,←,HH,HL);                                        01500000
  GO TO PP24 ;                                                    01501000
  CYCLE:                                                          01502000
  BEGIN                                                           01503000
    INTEGER K ;                                                   01504000
    REAL ZH,ZL,TZH,TZL ;                                          01505000
```

```
      LABEL CY1,CY2,CY3,RNKUT1,RNKUT2,RADICAL ;                          01506000
      LABEL SW1,SW2 ;                                                    01507000
      SWITCH SW ←SW1,SW2 ;                                               01508000
      DOUBLE(H1H,H1L,2,0,/,←,H2H,H2L);                                   01509000
      RNKUT(NU,H1H,H1L,RKH,RKL,U1H,U1L,FUNCTION) ;                       01510000
      CY1:FOR K ←1 STEP 1 UNTIL 2 DO                                     01511000
      BEGIN                                                              01512000
        DOUBLE(K,0,1,0,-,H2H,H2L,×,T2H,T2L,+,←,TZH,TZL);                 01513000
        RNKUT(NU,H2H,H2L,RKH,RKL,U2H,U2L,FUNCTION) ;                     01514000
        CY2:                                                             01515000
      END ;                                                              01516000
      GO TO RADICAL ;                                                    01517000
      CY3: GO TO SW[SNS];                                                01518000
      SW1:GO TO PP26 ;                                                   01519000
      SW2:GO TO PP27 ;                                                   01520000
      RADICAL:                                                           01521000
      BEGIN                                                              01522000
        INTEGER J ;                                                      01523000
        REAL SUMH,SUML,RH,RL ;                                           01524000
        SUMH ←SUML ←0 ;                                                  01525000
        FOR J ← 1 STEP 1 UNTIL NU DO                                     01526000
        BEGIN                                                            01527000
          DOUBLE(U2H[J],U2L[J],U1H[J],U1L[J],-,←,Z2H,Z2L) ;              01528000
          DOUBLE(Z2H,Z2L,Z2H,Z2L,×,SUMH,SUML,+,←,SUMH,SUML) ;            01529000
                                                                         01530000
        END ;                                                            01531000
        DROOT ;                                                          01532000
        DOUBLE(RH,RL,←,Z3H,Z3L) ;                                        01533000
        GO TO CY3 ;                                                      01534000
                                                                         01535000
      END RADICAL;                                                       01536000
                                                                         01537000
    END CYCLE ;                                                          01538000
                                                                         01539000
  END STEPSIZE2 ;                                                        01540000
  START2:                                                                01541000
  BEGIN                                                                  01542000
    DOUBLE(TNH[1],TNL[1],←,TZH,TZL);                                     01543000
    JNUDO DOUBLE(UH[J],UL[J],←,VNH[J,1],VNL[J,1]);                       01544000
    FLAG ← TRUE ;                                                        01545000
```

```
      INVDO                                                          01546000
      BEGIN                                                          01547000
        K ← I + NV ;                                                 01548000
        FUNCTION(K,UH,UL,ZH,ZL) ;                                    01549000
        DOUBLE(ZH,ZL,←,FNH[I,1],FNL[I,1]) ;                          01550000
                                                                     01551000
      END ;                                                          01552000
      FOR J ←1 STEP 1 UNTIL Q[S]DO                                   01553000
      BEGIN                                                          01554000
        DOUBLE(J,0,1,0,-,HH,HL,×,TNH[1],TNL[1],+,←,TZH,TZL);         01555000
        DOUBLE(J,0,HH,HL,×,TNH[1],TNL[1],+,←,TNH[J+1],TNL[J+1]);     01556000
        RNKUT(NU,HH,HL,RKH,RKL,UH,UL,FUNCTION);                      01557000
        INUDO DOUBLE(UH[I],UL[I],←,VNH[I,J+1],VNL[I,J+1]);           01558000
        INVDO                                                        01559000
        BEGIN                                                        01560000
          K ← I + NV ;                                               01561000
          FUNCTION(K,UH,UL,ZH,ZL) ;                                  01562000
          DOUBLE(ZH,ZL,←,FNH[I,J+1],FNL[I,J+1]) ;                    01563000
                                                                     01564000
        END ;                                                        01565000
                                                                     01566000
      END ;                                                          01567000
  CTR ←CTR+Q[S]+2;                                                   01568000
      GO TO PP25 ;                                                   01569000
                                                                     01570000
    END START2 ;                                                     01571000
    TABLE4;                                                          01572000
    BEGIN                                                            01573000
      N ←Q[S];                                                       01574000
      INVDO                                                          01575000
      BEGIN                                                          01576000
        DOUBLE(,0,←,SUM1H,SUM1L);                                    01577000
        DOUBLE(,0,←,SUM2H,SUM2L);                                    01578000
        FOR J ←0 STEP 1 UNTIL N DO                                   01579000
        BEGIN                                                        01580000
          DOUBLE(PCCH[S,J+1,5],PCCL[S,J+1,5],FNH[I,N-J+1],FNL[I      01581000
          ,N-J+1],×,SUM1H,SUM1L,+,←,SUM1H,SUM1L);                    01582000
          DOUBLE(PCCH[S,J+1,6],PCCL[S,J+1,6],FNH[I,N-J+1],FNL[I      01583000
          ,N-J+1],×,SUM2H,SUM2L,+,←,SUM2H,SUM2L)                     01584000
        END ;                                                        01585000
```

```
                DOUBLE(VNH[I+NV,N/2+1],VNL[I+NV,N/2+1],HH,HL,/,SUM1H,    01586000
                SUM1L,-,+,FRSTH[I,N/2+1],FRSTL[I,N/2+1]);                01587000
                DOUBLE(VNH[I,N/2+1],VNL[I,N/2+1],HH,HL,/,HH,HL,/,SUM2H   01588000
                ,SUM2L,-,+,SCNDH[I,N/2+1],SCNDL[I,N/2+1]);               01589000
                DOUBLE(FRSTH[I,(N+2)/2],FRSTL[I,(N+2)/2],FNH[I,(N+2)/2]  01590000
                ,FNL[I,(N+2)/2],+,+,FRSTH[I,(N+4)/2],FRSTL[I,(N+4)/2]);  01591000
                FOR J +(N+6)/2 STEP 1 UNTIL N+2 DO DOUBLE(FRSTH[I,J-1],  01592000
                FRSTL[I,J-1],FNH[I,J-1],FNL[I,J-1],+,+,FRSTH[I,J],FRSTL  01593000
                [I,J]);                                                  01594000
                FOR J +N/2 STEP -1 UNTIL 1 DO DOUBLE(FRSTH[I,J+1],FRSTL  01595000
                [I,J+1],FNH[I,J],FNL[I,J],-,+,FRSTH[I,J],FRSTL[I,J]);    01596000
                FOR J +(N+4)/2 STEP 1 UNTIL N+2 DO DOUBLE(SCNDH[I,J-1],  01597000
                SCNDL[I,J-1],FRSTH[I,J],FRSTL[I,J],+,+,SCNDH[I,J],SCNDL  01598000
                [I,J]);                                                  01599000
                FOR J +N/2 STEP -1 UNTIL 1 DO DOUBLE(SCNDH[I,J+1],SCNDL  01600000
                [I,J+1],FRSTH[I,J+1],FRSTL[I,J+1],-,+,SCNDH[I,J],SCNDL[I 01601000
                ,J]);                                                    01602000
                                                                         01603000
              END ;                                                      01604000
              GO TO AHEAD ;                                              01605000
                                                                         01606000
            END TABLE4 ;                                                 01607000
                                                                         01608000
          END RESTART ;                                                  01609000
                                                                         01610000
        END TEST ;                                                       01611000
                                                                         01612000
      END INTEGRATE ;                                                    01613000
      INTERPOLATE2;                                                      01614000
      BEGIN                                                              01615000
        HOMESAFE ;                                                       01616000
        GO TO PP21 ;                                                     01617000
                                                                         01618000
      END INTERPOLATE2 ;                                                 01619000
                                                                         01620000
    END COWELL ;                                                         01621000
    FINISH;WRITE (PRINT,FINAL);                                          01622000
    LAST;                                                                01623000
  END .                                                                  01624000
30
```

```
LABEL    00000000CARD    001      01
     1.2                                                          *    IC  1
    0.0                                                           *    IC  2
    0.0                                                           *    IC  3
   -1.049357509830319907 26                                       *    IC  4
    0.012128562765312310491206792 00                              *    IC  5
    6.192169331319639706 74                                       *    IC  6
    6.0                                                           *    IC  7
    5.0@-10                                                       *    IC  8
    5.0@-14                                                       *    IC  9
    5.0@-09                                                       *    IC 10
    1,                                                            /     STOP
```

C. The Adams Method

1. Description of Method

The method of Adams [16] is independent of the differential equations being integrated except in that a second order differential must be considered as a pair of coupled first order equations. In this report a pair of coupled second order equations are being dealt with and by the Adams method these are dealt with as if one had four first order equations.

In general, then one deals with the system of equations

$$\ddot{\vec{u}}(t) = \frac{d}{dt}\,\dot{\vec{u}}(t) = \vec{f}(\vec{u}(t),\, t) \quad , \tag{1-1}$$

where $\vec{u}$ and $\vec{f}$ can be thought of as vectors (4 components for this work), and

$$\vec{u} = \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} \quad .$$

This differential equation is equivalent to the integral equation

$$\dot{\vec{u}}(t_0+\Delta t) = \dot{\vec{u}}(t_0) + \int_{t_0}^{t_0+\Delta t} \vec{f}(\vec{u}(t),t)\, dt \quad . \tag{1-2}$$

This integral equation is then approximated by

$$\dot{\vec{u}}_{N+1} = \dot{\vec{u}}_N + \Delta t \sum_{\rho=0}^{q} \beta_{q\rho}\, \vec{f}_{N-\rho} \quad , \tag{1-3a}$$

or

$$\dot{\vec{u}}_{N+1} = \dot{\vec{u}}_N + \Delta t \sum_{\rho=0}^{q} \beta^*_{q\rho}\, \vec{f}_{N-\rho+1} \quad . \tag{1-3b}$$

Formula (1-3a) is the Adams-Bashforth predictor equation and (1-3b) is called the Adams-Moulton corrector. The coefficients $\beta_{q\rho}$ and $\beta^*_{q\rho}$ are chosen so

that N, the order of the method, is q + 1 . Explicit formulas for β and β^* are

$$\beta_{q\rho} = (-1)^{\rho}\left\{\binom{\rho}{\rho}\gamma_{\rho} + \binom{\rho+1}{\rho}\gamma_{\rho+1} + \cdots \binom{q}{\rho}\gamma_{q}\right\},$$

where $\binom{m}{n}$ represents the binomial coefficients, and the γ_ρ are found by the recursion relation

$$\gamma_{\rho} + \frac{1}{2}\gamma_{\rho-1} + \frac{1}{3}\gamma_{\rho-2} + \cdots \frac{1}{\rho+1}\gamma_{0} = 1 \ ,$$

$$\beta^*_{q\rho} = (-1)^{\rho}\left\{\binom{\rho}{\rho}\gamma^*_{\rho} + \binom{\rho+1}{\rho}\gamma^*_{\rho+1} + \cdots \binom{q}{\rho}\gamma^*_{q}\right\} \ ,$$

and the γ^*_ρ are found by the recursion relation

$$\gamma^*_{\rho} + \frac{1}{2}\gamma_{\rho-1} + \frac{1}{3}\gamma^*_{\rho-2} + \cdots \frac{1}{\rho+1}\gamma^*_{0} = \begin{cases} 1 \text{ if } \rho = 0 \\ 0 \text{ if } \rho = 1,2,\ldots \end{cases}$$

The error term for each is

$$\gamma_{q+1}(\Delta t)^{q+2}\frac{d^{q+2}\vec{u}}{dt^{q+2}} \qquad \text{(Adams-Bashforth)} \ ,$$

and

$$\gamma^*_{q+1}(\Delta t)^{q+2}\frac{d^{q+2}\vec{u}}{dt^{q+2}} \qquad \text{(Adams-Moulton)}$$

The Adams method is equivalent to approximating the integral in (1-2) by fitting $\vec{f}$ by a polynomial of degree q using the q+1 most recent points.

The coefficients $\beta_{q\rho}$ and $\beta^*_{q\rho}$ were calculated by a separate program up to q = 16 and are considered to be accurate to at least 24 decimal places. The coefficients $\beta_{q\rho}$ and $\beta^*_{q\rho}$ are given in decimal form at the end of the

program listing for the Adams method.

2. The Adams Computer Program

The program itself is written in double precision for the Burroughs B-5000. The language is close to Algol-60 except that all double precision arithmetic is written in Polish prefix notation. There are no unusual hardware requirements. The program may be executed on a minimum B-5000 system having only one card reader and one line printer.

2.1. Starting Procedure

Since the Adams method is a multistep method it cannot start itself but must rely on a starting procedure that will supply the first q+1 points at least as a history on which it can build. The starting procedure used here is the classical Runge-Kutta 4th order single step method.

2.2. Orders Used

The order, N = q+1 was variable and can range from 5 to 17 in this program. Not only can the order of the predictor and corrector be set separately for each orbit but the order changes during the run as dictated by the needs for accuracy. A maximum and minimum q value are set, called q_{max} and q_{min}, for the corrector and the order used at each step is determined by the estimated error in the last step. A parameter Δq is entered and the q value of the predictor is always $q - \Delta q$ where q+1 is the order of the corrector. The parameter Δq can be set to zero making the predictor and corrector of the same order and experience shows that $0 \leq \Delta q \leq 2$ works well.

2.3. Error Estimates and Step Size Control

In the Runge-Kutta starting procedure the error is estimated by taking two steps of size Δt and then a single step of $2\Delta t$ (starting at the same point) and comparing these two. If the difference in the velocities as

calculated in these two ways is greater than E_{max} then Δt is halved and a new attempt is made at starting. If this difference is smaller than E_{max} then the Runge-Kutta program proceeds with step size Δt until enough points are accumulated to start the Adams method. The value of E_{max} and the initial trial value of Δt are read in as program parameters with the initial data (see operating instructions of Adams method below).

In the Adams method proper the error is estimated by the difference between the predictor and the corrector values of the velocity. The velocity is calculated first by the Adams-Bashforth and then by the Adams-Moulton formula. If this difference is greater than E_{max} then an effort is made to increase the accuracy, and if this difference is smaller than E_{min} an effort is made to decrease the accuracy.

A decrease in the accuracy is achieved by either decreasing q, or increasing the step size, Δt. If $q > q_{min}$ and the error is too small, q is decreased by one. If the error is too small and $q = q_{min}$ and if there is sufficient history available, the step size is doubled. Otherwise the same q and Δt are continued.

An increase in the accuracy is achieved by either increasing q, or decreasing the step size. If $q < q_{max}$ and the error is too large, q is increased by one. If the error is too large and $q = q_{max}$, then the step size is halved. E_{max}, q_{min} and q_{max} are input parameters and E_{min} is determined by the empirical relation $E_{min} = 2^{-(2q_{min} - q_{max} + 3)} E_{max}$. Note that if the velocities are of the order of magnitude of unity, E_{min} cannot be smaller than about 10^{-22} when doing 23 decimal place arithmetic. This places a lower bound on E_{max} of $10^{-22} \times 2^{(2 q_{min} - q_{max} + 3)}$.

2.4. Reducing the Step Size

In multistep methods it is most convenient to reduce the step size by halving through interpolation. An interpolation polynomial of degree q_{int} is used where q_{int} is an input parameter. For maximum accuracy, mid point interpolation (Gaussian) was used and this requires that $[q_{int}/2]$ points be abandoned each time a halving of the step size takes place. While this seems wasteful, the interpolation procedure that kept all points so far calculated proved not accurate enough at the orders $q > 9$.

Explicitly, the interpolation formula is:

$$g_{N-1-L+\frac{1}{2}} = \sum_{s=L-q_h}^{q_{int}+L-q_h} A^{q_h}_{s-L+q_h} \, g_{N-1-s} \quad , \quad L = q_h, q_h+1, \ldots \quad ,$$

where

$$A^{q_h}_{k} = \frac{\prod_{J=0}^{q_{int}} (2q_h - 2J - 1)}{\prod_{(J\neq k)\, J=0}^{q_{int}} (2k - 2J)} \quad ,$$

and

$$q_h = [q_{int}/2] \text{ (i.e., the integer part of } q_{int}/2) \; .$$

2.5. Closing an Orbit

As the end of the period is approached the time interval, Δt is halved until it is as small as the original Δt used by the Runge-Kutta starting procedure. When the time is within $2\Delta t_{start}$ of the period (as determined by input data card No. 12), a single Runge-Kutta 4th order step

is taken which closes the period. Then an interpolation is made to a time that would give y = 0. The error in closing the orbit can then be taken as either the errors in x, y, $\dot{x}$ and $\dot{y}$ or as the errors in x, $\dot{x}$, $\dot{y}$ and the period.

2.6. Searching for Improved Initial Conditions

At a time close to the half period (as given by data input) a search is made for a crossing of the x-axis in a direction specified by data input card no. 10. The time interval Δt, is then successively halved as the crossing is approached until the magnitude of the change in y for each step is less than E_{min}. Steps are then taken until the x-axis is crossed. The value of $\dot{x}$ at this point and the value of $\dot{x}$ at the last half orbit are used together with the initial $\dot{y}$ of this and the previous orbit to do a linear interpolation for a new initial $\dot{y}$ that will further reduce the magnitude of $\dot{x}$. Half orbits are run until $|\dot{x}| < E_{max}$.

2.7. Data Output

The following is a description of the data output:

The first two lines printed give the information on the first 10 cards of the data input. Here q_{min} is referred to as QMIN, q_{max} as QMAX, Δq the difference between the order of the predictor and the corrector as DQ, q_{int} as QINT, E_{max} as EMAX, iteration as ITER, print on double as PRD, print on halve as PRH, print every step, PRE, and search for initial conditions as SEARCH. Following this is printed the suggested starting value of Δt (labeled DT), the period of this orbit (labeled PERIOD), and the value of μ (labeled MU).

Then at each point that orbit information is called to be printed the following is printed:

the number of function evaluations so far (labeled N),

the number of steps taken so far (labeled NS),

the q value to be used for the next step (labeled Q),

the processor time used so far (labeled T2),

the time interval to be used for the next step (labeled DT),

the time of this orbit point (labeled T),

the position and velocity coordinates (labeled X, Y, VX, and VY).

This orbit information is always printed out at the initial point and at the end of the period as given on data input card no. 12. At the end of the orbit, a value of Δt is estimated that will give $y = 0$ ($\Delta t = - y/\dot{y}$) and an interpolation is then made for $\dot{x}$ ($\dot{x}_{int} = \dot{x}_{final} + \Delta t\ \ddot{x}$). The errors for the orbit closure are then printed out in this order:

the difference between the initial and final value of x (labeled DX),

the value of y at the end of the period given on input data card no. 12 (labeled DY),

the value of $\dot{x}$ at the end of the period given on input data card no. 12 (labeled DVX),

the difference between the initial and final value of $\dot{y}$ (labeled DVY),

the time interval necessary to interpolate y to zero (labeled DT),

the value of $\dot{x}$ interpolated to the time corresponding to $y = 0$.

If a search for improved initial conditions is called for by the input data, this additional information is printed.

the orbit information at half orbit,

the new period based on this half orbit (labeled PERIOD),

the new initial value of $\dot{y}$ (labeled NEW VY [0]).

2.8. Data Input

Immediately following the program proper and called for under the file name FILEIN are the constants $\beta^*_{q\rho}$ and $\beta_{q\rho}$, for q from 0 to 16,

written one to the card ($17 \times \frac{17+1}{2} \times 2 = 306$ cards of data plus two comment cards). All other input data cards follow immediately behind these.

The data input cards are as follows:

Card No. 1:

The value of q_{min}, the smallest q value to be used by the corrector. This number must be written as a free field integer followed by a comma and lie in the range $4 \leq q_{min} \leq 16$.

Card No. 2:

The value of q_{max}, the largest q value to be used by the corrector. This number must be written as a free field integer, followed by a comma, and lie in the range $q_{min} \leq q_{max} \leq 16$.

Card No. 3:

The value of Δq, difference between the order of the corrector and the order of the predictor. If the predictor and corrector are to be of the same order enter 0, if the corrector is to be one greater than the predictor enter 1, etc. This number must be a free field integer, followed by a comma, and lie in the interval $0 \leq \Delta q \leq q_{min} - 4$.

Card No. 4:

The value of E_{max}. The maximum acceptable difference between the predicted value of the velocity and the corrected value of the velocity. This number must be a free field single precision real number, followed by a comma, and satisfy the inequality $E_{max} \geq 10^{-22} \times 2^{(2\ q_{min} - q_{max} + 3)}$.

Card No. 5:

The literal phrase "YES" or " NO" to denote whether the corrector is to be iterated at each point until two successive values for the velocity are smaller than E_{min}, or not. If " NO" here the corrector will be applied just

once. This entry must be a free field literal of 3 characters followed by a comma.

Card No. 6:

The literal phrase "YES" or " NO" to denote whether printing of the orbit information is to take place after doubling the step size, or not. This entry must be a free field literal of 3 characters followed by a comma.

Card No. 7:

The literal phrase "YES" or " NO" to denote whether printing of orbit information is to take place after halving of the step size, or not. This entry is a free field literal of 3 characters, followed by a comma.

Card No. 8:

The literal phrase "YES" or " NO" to denote whether printing of orbit information is to take place at every step, or not. This entry must be a free field literal of 3 characters followed by a comma.

Card No. 9:

The literal phrase "YES" or " NO" to denote whether a search for improved initial conditions is to be conducted. This entry must be a free field literal of 3 characters followed by a comma.

Card No. 10:

A 1 or -1 to indicate the direction of the axis crossing at half orbit. A 1 if $\dot{y} > 0$ or a -1 if $\dot{y} < 0$ at half orbit. Even though this information is used only if a search for improved initial conditions is requested, this card must always be included in the data input. This number must be a free field integer followed by a comma.

Card No. 11:

The value of Δt, the trial value for the initial step size. (If this step size is too large it will be halved until small enough to satisfy the

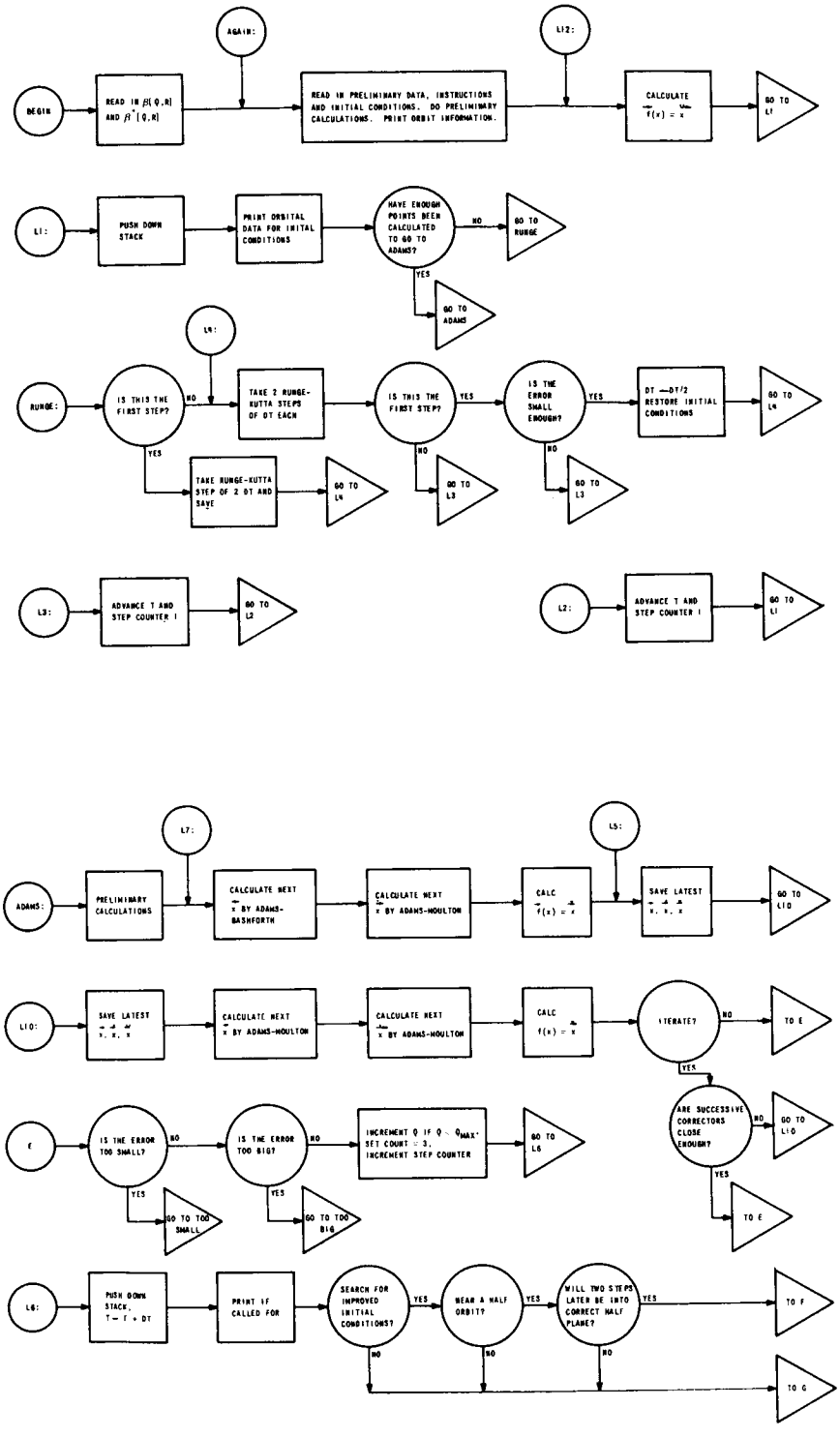

Figure 8. The Flow Diagram for the Adams Method.

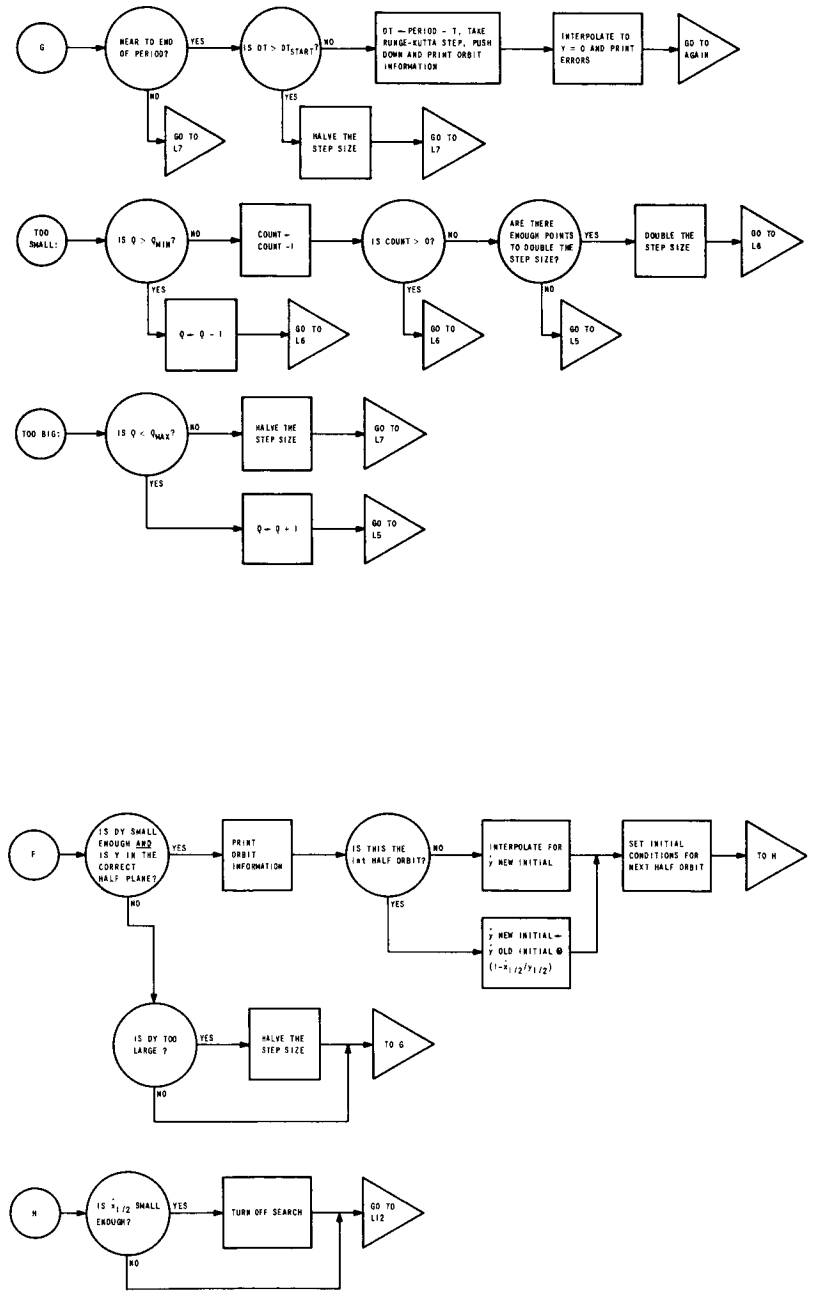

Figure 8 (Continued). The Flow Diagram for the Adams Method.

error requirements). This number must be a free field double precision real number not followed by a comma.

Card No. 12:

The value of the period for this orbit. This number must be a free field double precision real number not followed by a comma.

Card No. 13:

The value of μ. This number must be a free field double precision real number not followed by a comma.

Card No. 14:

The starting value of the x coordinate. This number must be a free field double precision real number not followed by a comma.

Card No. 15:

The starting value of the y coordinate. (This number must be present even though it is zero.) This number must be a free field double precision real number not followed by a comma.

Card No. 16:

The starting value of the x component of the velocity, $\dot{x}$. (This number must be present even though it is zero). This number must be a free field double precision real number not followed by a comma.

Card No. 17:

The starting value of the y component of the velocity, $\dot{y}$. This number must be a free field double precision real number not followed by a comma. If several orbits are to be run the sets of starting values are stacked behind each other.

It is also possible to put comments on the data cards by observing the followwing rules. For free field integers and single precision real numbers a slash, /, indicates that all the field on this card following the slash is

to be ignored. For double precision real numbers an asterisk, *, indicates that all the field on this card following the asterisk is to be ignored. Comments can be written on the data cards following the slash or asterisk.

3. Flow Diagram for the Adams Method

Figure 8 gives a detailed flow diagram of the computer program for the Adams method. The symbols and terminology used in the flow diagram correspond to those in the above discussion and in the program listing.

4. The Program Listing for the Adams Method

The following 31 pages list the program for the Adams method. Also listed at the end of the program are the Adams coefficients $\beta^*[q,\rho]$ and $\beta[q,\rho]$ from $q = 0$ to 16 and $\rho = 0$ to q.

```
 LABEL   000000000XXXXXX0010000001
                                                                                00000000
BEGIN                                                                           00001000
  COMMENT DECLARATIONS;                                                         00002000
  ALPHA ARRAY ALEPH1,ALEPH2,ALEPH3[0:4];                                        00003000
  ALPHA SEARCH,ITER,PRDUBL,PRHALV,PREVRY;                                       00004000
  LABEL AGAIN,L8,L9,THEEND;                                                     00005000
  ARRAY BETAH,BETAL[0:16,0:17];                                                 00006000
  REAL K1VXH,K1VXL,K1VYH,K1VYL,K1XH,K1XL,K1YH,K1YL,K2VXH,K2VXL,K2VYH,           00007000
  K2VYL,K2XH,K2XL ,K2YH,K2YL,K3VXH,K3VXL,K3VYH,K3VYL,K3XH,K3XL,K3YH,K3YL00008000
  ,K4VXH,K4VXL,K4VYH,K4VYL,K4XH,K4XL,K4YH,K4YL,XTH,XTL,X2DTH,X2DTL,YTH,         00009000
  YTL,Y2DTH,Y2DTL,VXTH,VXTL,VX2DTH,VX2DTL,PERIODH,X,Y,VYTH,VYTL,VY2DTH,         00010000
  VY2DTL,PERIODL,FXTH,FXTL,FX2DTH,FX2DTL,X1,H1Y,TXH,TYH,FYTH,FYTL,FY2DTH00011000
  ,FY2DTL,X2,L1Y,TXL,TYL,DNOMPH,DNOMMH,XOPMUH,XOMMUPH,YO2H,TH,DTH,MUH,          00012000
  MUPH,DOFRKH,DNOMPL,DNOMML,XOPMUL,XOMMUPL,YO2L,TL,DTL,MUL,MUPL,DOFRKL,         00013000
  T2,EMIN,EMAX,ETEST,TX,TY;                                                     00014000
  DEFINE XPH =X2DTH#,XPL =X2DTL#,YPH =Y2DTH#,YPL =Y2DTL#,VXPH =VX2DTH#,         00015000
  VXPL =VX2DTL#,VYPH =VY2DTH#,VYPL =VY2DTL#,FXPH =FX2DTH#,FXPL =FX2DTL#        00016000
  ,FYPH =FY2DTH#,FYPL =FY2DTL#;                                                 00017000
  INTEGER FLAG,NS,QPRIME,QMP,QS,M,N,Q,QMIN,QMAX,J,R;                            00018000
  FILE IN FILEIN(2,10);                                                         00019000
  FILE OUT FILEOUT 1(3,15);                                                     00020000
  DEFINE BSHQR =BETAH[16-Q,17-R]#,BSLQR =BETAL[16-Q,17-R]#;                     00021000
  LIST LIST1(QMIN,QMAX,QMP,QS,EMAX,ITER,PRDUBL,PRHALV,PREVRY,SEARCH,M );00022000
  FORMAT FRMT1(" QMIN"," QMAX","   DQ"," QINT","   EMAX   "," ITER","  P00023000
RD","  PRH","  PRE","  SEARCH  ",/,4I5,E10.2,5(X2,A3),I3,/);                     00024000
  LIST LISTA(FOR J :=0 STEP 1 UNTIL 4 DO ALEPH1[J],FOR J :=0 STEP 1             00025000
  UNTIL 4 DO ALEPH2[J],FOR J :=0 STEP 1 UNTIL 4 DO ALEPH3[J]);                  00026000
  FORMAT FORMATA(" DT =",5A6,"  PERIOD =",5A6,X5," MU =",5A6,/);                00027000
  LIST LISTXY1(N,NS,Q,(TIME(2)-T2)/60);                                         00028000
  FORMAT FMXY1("   N =",I5,X5," NS =",I5,X5,"  Q =",I3,X7," T2 =",F7.2,"00029000
 SEC "),FMXY2(X1,"DT = ",5A6,X5," X = ",5A6,X5," Y = ",5A6),FMXY3(X1," T00030000
 = ",5A6,X5,"VX = ",5A6,X5,"VY = ",5A6,/);                                      00031000
  DEFINE CARD =FILEIN #,PRINT =FILEOUT #,LAST=THEEND #;                         00032000
  INTEGER L;                                                                    00033000
  REAL ARRAY CON[0:7];                                                          00034000
  DEFINE DOSQRT =                                                               00035000
  BEGIN                                                                         00036000
    IF X1≠0 THEN                                                                00037000
    BEGIN                                                                       00038000
```

```
      CON[0]←0.00000269739885;                                          00039000
      CON[1]←0.0000016038835;                                           00040000
      CON[2]←0.000007629405;                                            00041000
      CON[3]←0.0000045364655;                                           00042000
      CON[4]←2.137099@23;                                               00043000
      CON[5]←1.270727@23;                                               00044000
      CON[6]←7.555786@22;                                               00045000
      CON[7]←4.482697@22;                                               00046000
      X←X1×1.0;                                                         00047000
      L←0;                                                              00048000
      L←X.[8:2];                                                        00049000
      L.[45:1]←X.[2:1];                                                 00050000
      Y←X;                                                              00051000
      Y.[3:6]←Y.[2:6];                                                  00052000
      Y←Y×CON[L];                                                       00053000
      Y←(X/Y+Y)×0.5;                                                    00054000
      Y←(X/Y+Y)×0.5;                                                    00055000
      Y←(X/Y+Y)×0.5;                                                    00056000
      Y←(X/Y+Y)×0.5;                                                    00057000
      DOUBLE(X1,X2,Y,0,/,Y,0,-,,0.5,×,Y,0,+,,←,H1Y,L1Y);                00058000
                                                                        00059000
    END ELSE H1Y←L1Y←0;                                                 00060000
                                                                        00061000
END#;                                                                   00062000
INTEGER EVER,NEVER ;                                                    00063000
INTEGER ARRAY NOTUSE1[0:80],NOTUSE2[0:9];                               00064000
REAL ARRAY NOTUSE3[0:68],NOTUSE4[0:68];                                 00065000
PROCEDURE DREAD(AH,AL);                                                 00066000
REAL AH,AL ;                                                            00067000
                                                                        00068000
BEGIN                                                                   00069000
  INTEGER K,T1 ;                                                        00070000
  REAL TH,TL ;                                                          00071000
  BOOLEAN DF,MF ;                                                       00072000
  LABEL ER,L1,L2,L3,L4,L5,L6,L7 ;                                       00073000
  FORMAT OUT FR(/"DATA CARD ERROR",X105);                               00074000
  STREAM PROCEDURE EC(S,D);                                             00075000
                                                                        00076000
  BEGIN                                                                 00077000
    SI :=S ;                                                            00078000
```

```
    DI :=D ;                                                          00079000
    DS :=8 LIT "0";                                                   00080000
    2(40(DS :=7 LIT "0";                                              00081000
    DS :=1 CHR));                                                     00082000
                                                                      00083000
  END ;                                                               00084000
  PROCEDURE GN ;                                                      00085000
                                                                      00086000
  BEGIN                                                               00087000
    LABEL L1 ;                                                        00088000
    IF NEVER ≤80 THEN GO TO L1 ELSE IF NEVER >81 THEN                 00089000
    BEGIN                                                             00090000
      READ (CARD,10,NOTUSE2[*])[LAST];                                00091000
      EC(NOTUSE2,NOTUSE1);                                            00092000
      NEVER :=1 ;                                                     00093000
      L1:EVER :=NOTUSE1[NEVER];                                       00094000
      IF EVER ="*"THEN                                                00095000
      BEGIN                                                           00096000
        NEVER :=81 ;                                                  00097000
        EVER :=" "END ELSE                                            00098000
      END ELSE EVER :=" ";                                            00099000
      NEVER :=NEVER +1 ;                                              00100000
                                                                      00101000
    END ;                                                             00102000
    L1:GN ;                                                           00103000
    IF EVER =" "THEN GO TO L1 ;                                       00104000
    DF :=FALSE ;                                                      00105000
    K :=0 ;                                                           00106000
    TH :=0 ;                                                          00107000
    TL :=0 ;                                                          00108000
    IF EVER ="-"THEN MF :=TRUE ELSE                                   00109000
    BEGIN                                                             00110000
      MF :=FALSE ;                                                    00111000
      IF EVER ≠"+"THEN GO TO L3 ELSE                                  00112000
    END ;                                                             00113000
    L2:GN ;                                                           00114000
    L3:IF EVER <10 THEN                                               00115000
    BEGIN                                                             00116000
      DF :=TRUE ;                                                     00117000
      DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL);                          00118000
```

```
    GO TO L2                                                                    00119000
  END ;                                                                         00120000
  IF EVER ="."THEN                                                              00121000
  BEGIN                                                                         00122000
    L4:GN ;                                                                     00123000
    IF EVER <10 THEN                                                            00124000
    BEGIN                                                                       00125000
      K :=K -1 ;                                                                00126000
      DF :=TRUE ;                                                               00127000
      DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL);                                    00128000
      GO TO L4                                                                  00129000
    END ELSE                                                                    00130000
  END ;                                                                         00131000
  IF MF THEN DOUBLE(,0,TH,TL,-,:=,TH,TL);                                       00132000
  IF NOT DF THEN                                                                00133000
  BEGIN                                                                         00134000
    ER:WRITE(PRINT,FR);                                                         00135000
    GO TO LAST                                                                  00136000
  END ;                                                                         00137000
  IF EVER =" "THEN GO TO L7 ;                                                   00138000
  IF EVER ≠"@"THEN GO TO ER ;                                                   00139000
  GN ;                                                                          00140000
  T1 :=0 ;                                                                      00141000
  DF :=FALSE ;                                                                  00142000
  IF EVER ="-"THEN MF :=TRUE ELSE                                               00143000
  BEGIN                                                                         00144000
    MF :=FALSE ;                                                                00145000
    IF EVER ≠"+"THEN GO TO L6 ELSE                                              00146000
  END ;                                                                         00147000
  L5:GN ;                                                                       00148000
  L6:IF EVER <10 THEN                                                           00149000
  BEGIN                                                                         00150000
    DF :=TRUE ;                                                                 00151000
    T1 :=T1 ×10 +EVER ;                                                         00152000
    GO TO L5                                                                    00153000
  END ;                                                                         00154000
  IF NOT DF THEN GO TO ER ;                                                     00155000
  IF EVER ≠" "THEN GO TO ER ;                                                   00156000
  IF MF THEN K :=K -T1 ELSE K :=K +T1 ;                                         00157000
  L7:IF K <0 THEN DOUBLE(TH,TL,NOTUSE3[-K],NOTUSE4[-K],/,:=,AH,AL               00158000
```

```
    )ELSE DOUBLE(TH,TL,NOTUSE3[K],NOTUSE4[K],X,:=,AH,AL);                    00159000
                                                                             00160000
  END ;                                                                      00161000
  PROCEDURE DPOP(A1,A2,F,ALF,A);                                             00162000
  VALUE A1,A2 ;                                                              00163000
  REAL A1,A2 ;                                                               00164000
  FILE F ;                                                                   00165000
  ALPHA ARRAY A,ALF[0];                                                      00166000
                                                                             00167000
  BEGIN                                                                      00168000
    STREAM PROCEDURE PRINT(X,Y,N,F,A,ALF);                                   00169000
    VALUE X,Y,N ;                                                            00170000
                                                                             00171000
    BEGIN                                                                    00172000
      LOCAL V1,V11,V2,V22,V3,V4,ST,RP ;                                      00173000
      DI ←LOC V22;                                                           00174000
      SI ←LOC X ;                                                            00175000
      SI ←SI+1 ;                                                             00176000
      SKIP 3 SB ;                                                            00177000
      13(DS ←3 RESET ;                                                       00178000
      3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                  00179000
      SKIP 1 SB));                                                           00180000
      SI ←LOC Y ;                                                            00181000
      SI ←SI+1 ;                                                             00182000
      SKIP 3 SB ;                                                            00183000
      13(DS ←3 RESET ;                                                       00184000
      3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                  00185000
      SKIP 1 SB));                                                           00186000
      DI ←LOC RP ;                                                           00187000
      DS ←8 LIT"00000000";                                                   00188000
      DI ←LOC ST ;                                                           00189000
      SI ←F ;                                                                00190000
      DS ←WDS ;                                                              00191000
      DI ←ST ;                                                               00192000
      SI ←LOC X ;                                                            00193000
      SKIP 1 SB ;                                                            00194000
      IF SB THEN DS ←8 LIT"     -0."ELSE DS ←8 LIT"      0.";                00195000
      3(DS ←8 LIT"00000000");                                                00196000
      11(DS ←8 LIT"        ");                                               00197000
      SI ←A ;                                                                00198000
```

```
    V4 ←SI ;                                                          00199000
    SI ←LOC V22;                                                      00200000
    V3 ←SI ;                                                          00201000
    26(DI ←LOC RP ;                                                   00202000
    DI ←DI +7 ;                                                       00203000
    DS ←CHR ;                                                         00204000
    SI ←V4 ;                                                          00205000
    DI ←ST ;                                                          00206000
    DI ←DI+8 ;                                                        00207000
    RP(DS ←24 ADD ;                                                   00208000
    DI ←DI -24 ;                                                      00209000
    SI ←SI -24);                                                      00210000
    SI ←SI +24 ;                                                      00211000
    V4 ←SI ;                                                          00212000
    SI ←V3 ;                                                          00213000
    SI ←SI+1 ;                                                        00214000
    V3 ←SI);                                                          00215000
    DI ←DI +24 ;                                                      00216000
    SI ←LOC N ;                                                       00217000
    SKIP 1 SB ;                                                       00218000
    IF SB THEN DS ←2 LIT "@-"ELSE DS ←2 LIT "@+";                     00219000
    SI ←LOC N ;                                                       00220000
    DS ←2 DEC ;                                                       00221000
    DI ←DI-1 ;                                                        00222000
    2(DS ←RESET);                                                     00223000
    DI ←ALF ;                                                         00224000
    SI ←ST ;                                                          00225000
    SI ←SI +5 ;                                                       00226000
    DI ←DI +2 ;                                                       00227000
    DS ←CHR ;                                                         00228000
    SI ←SI +2 ;                                                       00229000
    DS ←CHR ;                                                         00230000
    DS ←1 LIT".";                                                     00231000
    DS ←3 CHR ;                                                       00232000
    4(DI ←DI +2 ;                                                     00233000
    DS ←6 CHR);                                                       00234000
                                                                      00235000
END ;                                                                 00236000
STREAM PROCEDURE SHIFT(A,B,C);                                        00237000
VALUE A,B ;                                                           00238000
```

```
                                                                        00239000
BEGIN                                                                   00240000
  SI ←LOC A ;                                                           00241000
  DI ←C ;                                                               00242000
  DS ←CHR ;                                                             00243000
  SKIP 3 SB ;                                                           00244000
  DI ←DI +1 ;                                                           00245000
  36(IF SB THEN DS ←SET ELSE DS ←RESET ;                                00246000
  SKIP 1 SB);                                                           00247000
  SKIP 9 DB ;                                                           00248000
  3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                 00249000
  SKIP 1 SB);                                                           00250000
  SI ←LOC B ;                                                           00251000
  SKIP 9 SB ;                                                           00252000
  36(IF SB THEN DS ←SET ELSE DS ←RESET ;                                00253000
  SKIP 1 SB);                                                           00254000
                                                                        00255000
END ;                                                                   00256000
INTEGER Y,I ;                                                           00257000
REAL T1,T2,HT ,LT;                                                      00258000
ARRAY T[0:1];                                                           00259000
LABEL L1,L2,L3 ;                                                        00260000
IF A1 =0 THEN                                                           00261000
BEGIN                                                                   00262000
  T1 ←T2 ←0 ;                                                           00263000
  GO TO L2                                                              00264000
END ;                                                                   00265000
HT ←1 ;                                                                 00266000
LT ←0 ;                                                                 00267000
Y ←0.90309 ×(IF BOOLEAN(A1.[2:1])THEN 13 -A1.[3:6]ELSE 13 +A1.[3:600268000
]);                                                                     00269000
FOR I ←1 STEP 1 UNTIL ABS(Y)DO DOUBLE(HT,LT,,10,×,←,HT,LT);            00270000
IF Y>0 THEN DOUBLE(A1,A2,HT,LT,/,←,T1,T2)ELSE DOUBLE(A1,A2,HT,LT,×00271000
,←,T1,T2);                                                              00272000
Y ←Y-1 ;                                                                00273000
L1:IF T1.[3:6]<13 THEN                                                  00274000
BEGIN                                                                   00275000
  DOUBLE(T1,T2,,10.0,/,←,T1,T2);                                        00276000
  Y ←Y+1                                                                00277000
END ELSE IF T1.[3:6]>13 THEN                                            00278000
```

```
      BEGIN                                                                    00279000
        DOUBLE(T1,T2,,10.0,×,←,T1,T2);                                         00280000
        Y ←Y-1                                                                 00281000
      END ELSE GO TO L2 ;                                                      00282000
      IF T1.[3:6]=14 THEN                                                      00283000
      BEGIN                                                                    00284000
        SHIFT(T1,T2,T);                                                        00285000
        PRINT(T[0],T[1],Y,F,A,ALF);                                            00286000
        GO TO L3                                                               00287000
      END ELSE GO TO L1 ;                                                      00288000
      L2:PRINT(T1,T2,Y,F,A,ALF );                                              00289000
      L3:                                                                      00290000
    END ;                                                                      00291000
    ALPHA ARRAY A[0:77];                                                       00292000
    FILL A[*]WITH "12500000","00000000","00000000","01562500","00000000"00293000
    ,"00000000","00195312","50000000","00000000","00024414","06250000","00294000
00000000","00003051","75781250","00000000","00000381","46972656","25000000295000
00","00000047","68371582","03125000","00000005","96046447","75390625","000296000
0000000","74505805","96923828","00000000","09313225","74615479","000000000297000
0","01164153","21826935","00000000","00145519","15228367","00000000","0000298000
018189","89403546","00000000","00002273","73675443","00000000","000028400299000
","21709430","00000000","00000035","52713679","00000000","00000004","44000300000
89210","00000000","00000000","55511151","00000000","00000000","06938894"00301000
    ,"00000000","00000000","00867362","00000000","00000000","00108420","00302000
00000000","00000000","00013553","00000000","00000000","00001694","00000000303000
00","00000000","0 000212","00000000","00000000","00000026","00000000","000304000
0000000","00000003";                                                          00305000
                                                                              00306000
    BEGIN                                                                     00307000
      INTEGER I ;                                                             00308000
      DOUBLE(,1,:=,NOTUSE3[0],NOTUSE4[0]);                                    00309000
      FOR I :=1 STEP 1 UNTIL 33 DO DOUBLE(,10,NOTUSE3[I -1],NOTUSE4[I -100310000
      ],×,:=,NOTUSE3[I],NOTUSE4[I]);                                          00311000
      FOR I :=1 STEP 1 UNTIL 35 DO DOUBLE(NOTUSE3[33],NOTUSE4[33],            00312000
      NOTUSE3[I],NOTUSE4[I],×,:=,NOTUSE3[I +33],NOTUSE4[I +33]);              00313000
      NEVER :=82 ;                                                            00314000
                                                                              00315000
    END ;                                                                     00316000
    COMMENT INSERT BETAS HERE;                                                00317000
                                                                              00318000
```

```
BEGIN                                                                    00319000
  FOR Q :=0 STEP 1 UNTIL 16 DO FOR R :=0 STEP 1 UNTIL Q DO               00320000
  BEGIN                                                                  00321000
    DREAD(BSHQR,BSLQR);                                                  00322000
                                                                         00323000
  END;                                                                   00324000
  FOR Q :=0 STEP 1 UNTIL 16 DO FOR R :=0 STEP 1 UNTIL Q DO               00325000
  BEGIN                                                                  00326000
    DREAD(BETAH[Q,R],BETAL[Q,R]);                                        00327000
                                                                         00328000
  END;                                                                   00329000
                                                                         00330000
END;                                                                     00331000
AGAIN:WRITE(FILEOUT[DBL]);                                               00332000
T2 :=TIME(2);                                                            00333000
READ(FILEIN,/,LIST1)[THEEND];                                            00334000
WRITE(FILEOUT,FRMT1,LIST1);                                              00335000
DREAD(DTH,DTL);                                                          00336000
DREAD(PERIODH,PERIODL);                                                  00337000
DREAD(MUH,MUL);                                                          00338000
DPOP(DTH,DTL,FILEOUT,ALEPH1,A);                                          00339000
DPOP(PERIODH,PERIODL,FILEOUT,ALEPH2,A);                                  00340000
DPOP(MUH,MUL,FILEOUT,ALEPH3,A);                                          00341000
WRITE(FILEOUT,FORMATA,LISTA);                                            00342000
ETEST :=EMIN :=EMAX/2*(QMIN +3 -QMAX +QMIN );                            00343000
IF PREVRY ="YES"THEN FLAG :=1 ELSE FLAG :=0;                             00344000
QPRIME :=4 ×(QMAX +1);                                                   00345000
NS :=0;                                                                  00346000
                                                                         00347000
BEGIN                                                                    00348000
  ARRAY XH,XL,YH,YL,VXH,VXL,VYH,VYL,FYH,FYL,FXH,FXL[0:QPRIME];            00349000
  DEFINE RESTOREINITIAL =                                                00350000
  BEGIN                                                                  00351000
    X2DTH :=XH[1];                                                       00352000
    X2DTL :=XL[1];                                                       00353000
    Y2DTH :=YH[1];                                                       00354000
    Y2DTL :=YL[1];                                                       00355000
    VX2DTH :=VXH[1];                                                     00356000
    VX2DTL :=VXL[1];                                                     00357000
    VY2DTH :=VYH[1];                                                     00358000
```

```
    VY2DTL :=VYL[1];                                                    00359000
    FX2DTH :=FXH[1];                                                    00360000
    FX2DTL :=FXL[1];                                                    00361000
    FY2DTH :=FYH[1];                                                    00362000
    FY2DTL :=FYL[1];                                                    00363000
    XH[1]:=XH[2];                                                       00364000
    XL[1]:=XL[2];                                                       00365000
    YH[1]:=YH[2];                                                       00366000
    YL[1]:=YL[2];                                                       00367000
    VXH[1]:=VXH[2];                                                     00368000
    VXL[1]:=VXL[2];                                                     00369000
    VYH[1]:=VYH[2];                                                     00370000
    VYL[1]:=VYL[2];                                                     00371000
    FXH[1]:=FXH[2];                                                     00372000
    FXL[1]:=FXL[2];                                                     00373000
    FYH[1]:=FYH[2];                                                     00374000
    FYL[1]:=FYL[2];                                                     00375000
                                                                        00376000
  END #;                                                                00377000
  DEFINE FXFYCALC =                                                     00378000
  BEGIN                                                                 00379000
    DOUBLE(XH[0],XL[0],MUH,MUL,+,←,XOPMUH,XOPMUL);                      00380000
    DOUBLE(XH[0],XL[0],MUPH,MUPL,-,←,XOMMUPH,XOMMUPL);                  00381000
    DOUBLE(YH[0],YL[0],YH[0],YL[0],×,←,Y02H,Y02L);                      00382000
    DOUBLE(XOPMUH,XOPMUL,XOPMUH,XOPMUL,×,Y02H,Y02L,+,←,X1,X2);          00383000
    DDSQRT;                                                             00384000
    DOUBLE(X1,X2,H1Y,L1Y,×,MUPH,MUPL,/,←,DNOMPH,DNOMPL);                00385000
    DOUBLE(XOMMUPH,XOMMUPL,XOMMUPH,XOMMUPL,×,Y02H,Y02L,+,←,X1,X2);      00386000
    DDSQRT;                                                             00387000
    DOUBLE(X1,X2,H1Y,L1Y,×,MUH,MUL,/,←,DNOMMH,DNOMML);                  00388000
    DOUBLE(XH[0],XL[0],XOMMUPH,XOMMUPL,DNOMMH,DNOMML,/,-,XOPMUH,        00389000
    XOPMUL,DNOMPH,DNOMPL,/,-,VYH[0],VYL[0],+,VYH[0],VYL[0],+,←,FXH[000390000
    ],FXL[0]);                                                          00391000
    DOUBLE(YH[0],YL[0],YH[0],YL[0],DNOMPH,DNOMPL,/,-,YH[0],YL[0],       00392000
    DNOMMH,DNOMML,/,-,VXH[0],VXL[0],-,VXH[0],VXL[0],-,←,FYH[0],FYL[000393000
    ]);                                                                 00394000
    N :=N +1;                                                           00395000
                                                                        00396000
  END #;                                                                00397000
  DEFINE PUSHDOWN =                                                     00398000
```

```
BEGIN                                                              00399000
  FOR J :=QPRIME -1 STEP -1 UNTIL 0 DO                             00400000
  BEGIN                                                            00401000
    XH[J+1]:=XH[J];                                                00402000
    XL[J+1]:=XL[J];                                                00403000
    YH[J+1]:=YH[J];                                                00404000
    YL[J+1]:=YL[J];                                                00405000
    VXH[J+1]:=VXH[J];                                              00406000
    VXL[J+1]:=VXL[J];                                              00407000
    VYH[J+1]:=VYH[J];                                              00408000
    VYL[J+1]:=VYL[J];                                              00409000
    FXH[J+1]:=FXH[J];                                              00410000
    FXL[J+1]:=FXL[J];                                              00411000
    FYH[J+1]:=FYH[J];                                              00412000
    FYL[J+1]:=FYL[J];                                              00413000
                                                                   00414000
  END;                                                             00415000
                                                                   00416000
END #;                                                             00417000
DEFINE STORETOP =                                                  00418000
BEGIN                                                              00419000
  X2DTH :=XH[0];                                                   00420000
  X2DTL :=XL[0];                                                   00421000
  Y2DTH :=YH[0];                                                   00422000
  Y2DTL :=YL[0];                                                   00423000
  VX2DTH :=VXH[0];                                                 00424000
  VX2DTL :=VXL[0];                                                 00425000
  VY2DTH :=VYH[0];                                                 00426000
  VY2DTL :=VYL[0];                                                 00427000
  FX2DTH :=FXH[0];                                                 00428000
  FX2DTL :=FXL[0];                                                 00429000
  FY2DTH :=FYH[0];                                                 00430000
  FY2DTL :=FYL[0];                                                 00431000
                                                                   00432000
END #;                                                             00433000
DEFINE PRINT =IF FLAG =1 THEN                                      00434000
BEGIN                                                              00435000
  INTEGER J;                                                       00436000
  WRITE(FILEOUT,FMXY1,LISTXY1);                                    00437000
  DPOP(DTH,DTL,FILEOUT,ALEPH1,A);                                  00438000
```

```
    DPOP(XH[1],XL[1],FILEOUT,ALEPH2,A);                                  00439000
    DPOP(YH[1],YL[1],FILEOUT,ALEPH3,A);                                  00440000
    WRITE(FILEOUT,FMXY2,LISTA );                                         00441000
    DPOP(TH,TL,FILEOUT,ALEPH1,A);                                        00442000
    DPOP(VXH[1],VXL[1],FILEOUT,ALEPH2,A);                                00443000
    DPOP(VYH[1],VYL[1],FILEOUT,ALEPH3,A);                                00444000
    WRITE(FILEOUT,FMXY3,LISTA );                                         00445000
    IF PREVRY ≠"YES"THEN FLAG :=0;                                       00446000
                                                                         00447000
  END #;                                                                 00448000
  PROCEDURE RK;                                                          00449000
                                                                         00450000
  BEGIN                                                                  00451000
    DOUBLE(DOFRKH,DOFRKL,FXH[1],FXL[1],×,K1VXH,K1VXL);                   00452000
    DOUBLE(DOFRKH,DOFRKL,FYH[1],FYL[1],×,K1VYH,K1VYL);                   00453000
    DOUBLE(DOFRKH,DOFRKL,VXH[1],VXL[1],×,K1XH,K1XL );                    00454000
    DOUBLE(DOFRKH,DOFRKL,VYH[1],VYL[1],×,K1YH,K1YL );                    00455000
    DOUBLE(K1VXH,K1VXL,2,0,/,VXH[1],VXL[1],+,←,VXH[0],VXL[0]);           00456000
    DOUBLE(K1VYH,K1VYL,2,0,/,VYH[1],VYL[1],+,←,VYH[0],VYL[0]);           00457000
    DOUBLE(K1XH,K1XL,2,0,/,XH[1],XL[1],+,←,XH[0],XL[0]);                 00458000
    DOUBLE(K1YH,K1YL,2,0,/,YH[1],YL[1],+,←,YH[0],YL[0]);                 00459000
    FXFYCALC;                                                            00460000
    DOUBLE(DOFRKH,DOFRKL,FXH[0],FXL[0],×,K2VXH,K2VXL);                   00461000
    DOUBLE(DOFRKH,DOFRKL,FYH[0],FYL[0],×,K2VYH,K2VYL);                   00462000
    DOUBLE(DOFRKH,DOFRKL,VXH[0],VXL[0],×,K2XH ,K2XL );                   00463000
    DOUBLE(DOFRKH,DOFRKL,VYH[0],VYL[0],×,K2YH ,K2YL );                   00464000
    DOUBLE(K2VXH,K2VXL,2,0,/,VXH[1],VXL[1],+,←,VXH[0],VXL[0]);           00465000
    DOUBLE(K2VYH,K2VYL,2,0,/,VYH[1],VYL[1],+,←,VYH[0],VYL[0]);           00466000
    DOUBLE(K2XH,K2XL,2,0,/,XH[1],XL[1],+,←,XH[0],XL[0]);                 00467000
    DOUBLE(K2YH,K2YL,2,0,/,YH[1],YL[1],+,←,YH[0],YL[0]);                 00468000
    FXFYCALC;                                                            00469000
    DOUBLE(DOFRKH,DOFRKL,FXH[0],FXL[0],×,K3VXH,K3VXL);                   00470000
    DOUBLE(DOFRKH,DOFRKL,FYH[0],FYL[0],×,K3VYH,K3VYL);                   00471000
    DOUBLE(DOFRKH,DOFRKL,VXH[0],VXL[0],×,K3XH,K3XL);                     00472000
    DOUBLE(DOFRKH,DOFRKL,VYH[0],VYL[0],×,K3YH,K3YL);                     00473000
    DOUBLE(K3VXH,K3VXL,VXH[1],VXL[1],+,←,VXH[0],VXL[0]);                 00474000
    DOUBLE(K3VYH,K3VYL,VYH[1],VYL[1],+,←,VYH[0],VYL[0]);                 00475000
    DOUBLE(K3XH,K3XL,XH[1],XL[1],+,←,XH[0],XL[0]);                       00476000
    DOUBLE(K3YH,K3YL,YH[1],YL[1],+,←,YH[0],YL[0]);                       00477000
    FXFYCALC;                                                            00478000
```

```
  DOUBLE(DOFRKH,DOFRKL,FXH[0],FXL[0],×,←,K4VXH,K4VXL);                    00479000
  DOUBLE(DOFRKH,DOFRKL,FYH[0],FYL[0],×,←,K4VYH,K4VYL);                    00480000
  DOUBLE(DOFRKH,DOFRKL,VXH[0],VXL[0],×,←,K4XH,K4XL);                      00481000
  DOUBLE(DOFRKH,DOFRKL,VYH[0],VYL[0],×,←,K4YH,K4YL);                      00482000
  DOUBLE(K1VXH,K1VXL,K2VXH,K2VXL,+,K2VXH,K2VXL,+,K3VXH,K3VXL,+,          00483000
  K3VXH,K3VXL,+,K4VXH,K4VXL,+,6,0,/,VXH[1],VXL[1],+,←,VXH[0],VXL[000484000
  ]);                                                                     00485000
  DOUBLE(K1VYH,K1VYL,K2VYH,K2VYL,+,K2VYH,K2VYL,+,K3VYH,K3VYL,+,          00486000
  K3VYH,K3VYL,+,K4VYH,K4VYL,+,6,0,/,VYH[1],VYL[1],+,←,VYH[0],VYL[000487000
  ]);                                                                     00488000
  DOUBLE(K1XH ,K1XL ,K2XH ,K2XL ,+,K2XH ,K2XL ,+,K3XH ,K3XL ,+,          00489000
  K3XH ,K3XL ,+,K4XH ,K4XL ,+,6,0,/,XH[1],XL[1],+,←,XH[0],XL[0]);        00490000
  DOUBLE(K1YH ,K1YL ,K2YH ,K2YL ,+,K2YH ,K2YL ,+,K3YH ,K3YL ,+,          00491000
  K3YH ,K3YL ,+,K4YH ,K4YL ,+,6,0,/,YH[1],YL[1],+,←,YH[0],YL[0]);        00492000
  FXFYCALC;                                                               00493000
                                                                          00494000
END ;                                                                     00495000
REAL DVXH,DYH,XINH,YINH,VXINH,VYINNEWH,DSTRTH,DVXL,DYL,XINL,YINL,         00496000
VXINL,VYINNEWL,DSTRTL,VYINLASTH,VYINTHISH,VYINNEXTH,VXHALVELASTH,         00497000
VYINLASTL,VYINTHISL,VYINNEXTL,VXHALVELASTL;                               00498000
DEFINE VYINH =VYINTHISH #,VYINL =VYINTHISL #;                             00499000
FORMAT FMHALVE("PERIOD =",5A6,"   NEW VY[0] =",5A6,/,//);                00500000
LABEL L12;                                                                00501000
DEFINE SEARCHHALVEPOINT =                                                 00502000
BEGIN                                                                     00503000
  IF ABS(DYH)≤EMIN AND M×YH[1]≥0 THEN                                     00504000
  BEGIN                                                                   00505000
    INTEGER J;                                                            00506000
    LIST LSTHALVE(FOR J :=0 STEP 1 UNTIL 4 DO ALEPH1[J],FOR J :=0         00507000
    STEP 1 UNTIL 4 DO ALEPH2[J]);                                         00508000
    FLAG :=1;                                                             00509000
    PRINT;                                                                00510000
    IF VYINLASTH ≠0 THEN                                                  00511000
    BEGIN                                                                 00512000
      DOUBLE(VXH[1],VXL[1],VXHALVELASTH,VXHALVELASTL,-,←,DVXH,DVXL00513000
      );                                                                  00514000
      DOUBLE(VYINTHISH,VYINTHISL,VYINLASTH,VYINLASTL,-,                   00515000
      -VXHALVELASTH,VXHALVELASTL,×,DVXH,DVXL,/,VYINLASTH,VYINLASTL00516000
      ,+,←,VYINNEWH,VYINNEWL);                                            00517000
                                                                          00518000
```

```
      END ELSE DOUBLE(VYINTHISH,VYINTHISL,,1,VXH[1]/VYH[1],0,-,×,+, 00519000
      VYINNEWH,VYINNEWL);                                           00520000
      DOUBLE(TH,TL,TH,TL,+,+,PERIODH,PERIODL);                      00521000
      DPOP(PERIODH,PERIODL,FILEOUT,ALEPH1,A);                       00522000
      DPOP(VYINNEWH,VYINNEWL,FILEOUT,ALEPH2,A);                     00523000
      WRITE(FILEOUT,FMHALVE,LSTHALVE);                              00524000
      VXHALVELASTH :=VXH[1];                                        00525000
      VXHALVELASTL :=VXL[1];                                        00526000
      VYINLASTH :=VYINTHISH;                                        00527000
      VYINLASTL :=VYINTHISL;                                        00528000
      VYH[0]:=VYINTHISH :=VYINNEWH;                                 00529000
      VYL[0]:=VYINTHISL :=VYINNEWL;                                 00530000
      VXH[0]:=VXINH;                                                00531000
      VXL[0]:=VXINL;                                                00532000
      YH[0]:=YINH ;                                                 00533000
      YL[0]:=YINL ;                                                 00534000
      XH[0]:=XINH ;                                                 00535000
      XL[0]:=XINL ;                                                 00536000
      DTH :=DSTRTH;                                                 00537000
      DTL :=DSTRTL;                                                 00538000
      IF ABS(VXH[1])≤EMAX THEN SEARCH :=" NO";                      00539000
      GO TO L12;                                                    00540000
                                                                    00541000
    END;                                                            00542000
    IF ABS(DYH)≥EMIN THEN HALVESTEPSIZE;                            00543000
                                                                    00544000
  END #;                                                            00545000
  DREAD(XH[0],XL[0]);                                               00546000
  XINH :=XH[0];                                                     00547000
  XINL :=XL[0];                                                     00548000
  DREAD(YH[0],YL[0]);                                               00549000
  YINH :=YH[0];                                                     00550000
  YINL :=YL[0];                                                     00551000
  DREAD(VXH[0],VXL[0]);                                             00552000
  VXINH :=VXH[0];                                                   00553000
  VXINL :=VXL[0];                                                   00554000
  DREAD(VYH[0],VYL[0]);                                             00555000
  VYINTHISH:=VYH[0];                                                00556000
  VYINTHISL:=VYL[0];                                                00557000
  DOUBLE(1,0,MUH,MUL,-,+,MUPH,MUPL);                               00558000
```

```
VYINLASTH :=VYINLASTL :=0;                                                    00559000
L12:                                                                          00560000
BEGIN                                                                         00561000
  LABEL L1,L2,L3,L4,ADAMS,RUNGAK;                                             00562000
  N :=Q :=TH :=TL :=0;                                                        00563000
  FLAG :=1;                                                                   00564000
  FXFYCALC;                                                                   00565000
  GO TO L1;                                                                   00566000
  L2:DOUBLE(TH,TL,DTH,DTL,+,←,TH,TL);                                         00567000
  NS :=NS +1;                                                                 00568000
  L1:PUSHDOWN;                                                                00569000
  PRINT;                                                                      00570000
  IF TH/DTH >QMIN -0.5 THEN GO TO ADAMS ELSE GO TO RUNGAK;                    00571000
  RUNGAK:IF TH =0 THEN                                                        00572000
  BEGIN                                                                       00573000
    DOUBLE(DTH,DTL,DTH,DTL,+,←,DOFRKH,DOFRKL);                                00574000
    RK;                                                                       00575000
    STORETOP;                                                                 00576000
                                                                              00577000
  END;                                                                        00578000
  L4:DOFRKH :=DTH;                                                            00579000
  DOFRKL :=DTL;                                                               00580000
  RK;                                                                         00581000
  PUSHDOWN;                                                                   00582000
  RK;                                                                         00583000
  IF TH ≠0 THEN GO TO L3;                                                     00584000
  DOUBLE(VX2DTH,VX2DTL,VXH[0],VXL[0],-,←,TXH,TXL);                            00585000
  DOUBLE(VY2DTH,VY2DTL,VYH[0],VYL[0],-,←,TYH,TYL);                            00586000
  IF ABS(TXH)≤ETEST AND ABS(TYH)≤ETEST THEN                                   00587000
  BEGIN                                                                       00588000
    L3:DOUBLE(TH,TL,DTH,DTL,+,←,TH,TL);                                       00589000
    NS :=NS +1;                                                               00590000
    GO TO L2;                                                                 00591000
                                                                              00592000
  END;                                                                        00593000
  DOUBLE(DTH,DTL,2,0,/,←,DTH,DTL);                                            00594000
  RESTOREINITIAL;                                                             00595000
  GO TO L4;                                                                   00596000
  ADAMS:                                                                      00597000
END;                                                                          00598000
```

```
                                                                        00599000
BEGIN                                                                   00600000
  LABEL L5,L6,L7,L8,L9,L10,FINISH,TOOSMALL,TOOBIG;                      00601000
  INTEGER COUNT,QH,P,QD,K,G,QM,QT,J,J2,L,S,L2;                          00602000
  REAL TEMPH,TEMPL,TEMPXH,TEMPXL,TEMPYH,TEMPYL,XPPH,YPPH,VXPPH,         00603000
  VYPPH,FXPPH,FYPPH,XPPL,YPPL,VXPPL,VYPPL,FXPPL,FYPPL;                  00604000
  ARRAY XXH,XXL,YYH,YYL,VVXH,VVYH,FFXH,FFYH,VVXL,VVYL,FFXL,FFYL[0      00605000
  :QPRIME],AH,AL[0:16,0:16];                                            00606000
  DEFINE ADAMSBASHV =                                                   00607000
  BEGIN                                                                 00608000
    TEMPXH :=TEMPXL :=TEMPYH :=TEMPYL :=0;                              00609000
    P :=Q -QMP;                                                         00610000
    FOR R :=0 STEP 1 UNTIL P DO                                         00611000
    BEGIN                                                               00612000
      DOUBLE(FXH[R+1],FXL[R+1],BETAH[P,R],BETAL[P,R],×,TEMPXH,          00613000
      TEMPXL,+,←,TEMPXH,TEMPXL);                                        00614000
      DOUBLE(FYH[R+1],FYL[R+1],BETAH[P,R],BETAL[P,R],×,TEMPYH,          00615000
      TEMPYL,+,←,TEMPYH,TEMPYL);                                        00616000
                                                                        00617000
    END;                                                                00618000
    DOUBLE(TEMPXH,TEMPXL,DTH,DTL,×,VXH[1],VXL[1],+,←,VXH[0],VXL[0]      00619000
    );                                                                  00620000
    DOUBLE(TEMPYH,TEMPYL,DTH,DTL,×,VYH[1],VYL[1],+,←,VYH[0],VYL[0]      00621000
    );                                                                  00622000
                                                                        00623000
  END #;                                                                00624000
  DEFINE COMPUTEA =                                                     00625000
  BEGIN                                                                 00626000
    INTEGER J,K;                                                        00627000
    ARRAY EH,EL[0:QT],CH,CL[0:QMAX];                                    00628000
    REAL TEMPH,TEMPL;                                                   00629000
    FOR L :=QH DO                                                       00630000
    BEGIN                                                               00631000
      TEMPH :=1;                                                        00632000
      TEMPL :=0;                                                        00633000
      FOR J :=0 STEP 1 UNTIL QS DO DOUBLE(L,0,L,0,+,J,0,-,J,0,-,1       00634000
      ,0,-,TEMPH,TEMPL,×,←,TEMPH,TEMPL);                                00635000
      EH[L]:=TEMPH;                                                     00636000
      EL[L]:=TEMPL;                                                     00637000
                                                                        00638000
```

```
    END;                                                                    00639000
    FOR S :=0 STEP 1 UNTIL QS DO                                            00640000
    BEGIN                                                                   00641000
      TEMPH :=1;                                                            00642000
      TEMPL :=0;                                                            00643000
      FOR J :=0 STEP 1 UNTIL S -1,S+1 STEP 1 UNTIL QS DO DOUBLE(S           00644000
      ,0,S,0,+,J,0,-,J,0,-,TEMPH,TEMPL,x,+,TEMPH,TEMPL);                    00645000
      CH[S]:=TEMPH;                                                         00646000
      CL[S]:=TEMPL;                                                         00647000
                                                                            00648000
    END;                                                                    00649000
    FOR S :=0 STEP 1 UNTIL QS DO FOR L :=QH DO                              00650000
    BEGIN                                                                   00651000
      DOUBLE(EH[L],EL[L],L,0,L,0,+,1,0,-,S,0,-,S,0,-,CH[S],CL[S],x00652000
      ,/,+,AH[L,S],AL[L,S]);                                                00653000
                                                                            00654000
    END;                                                                    00655000
                                                                            00656000
END #;                                                                      00657000
DEFINE ADAMSMOULV =                                                         00658000
BEGIN                                                                       00659000
  TEMPXH :=TEMPXL :=TEMPYH :=TEMPYL :=0;                                    00660000
  FOR R :=0 STEP 1 UNTIL Q DO                                               00661000
  BEGIN                                                                     00662000
    DOUBLE(FXH[R],FXL[R],BSHQR,BSLQR,x,TEMPXH,TEMPXL,+,+,TEMPXH 00663000
    ,TEMPXL);                                                               00664000
    DOUBLE(FYH[R],FYL[R],BSHQR,BSLQR,x,TEMPYH,TEMPYL,+,+,TEMPYH 00665000
    ,TEMPYL);                                                               00666000
                                                                            00667000
  END;                                                                      00668000
  DOUBLE(TEMPXH,TEMPXL,DTH,DTL,x,VXH[1],VXL[1],+,+,VXH[0],VXL[0]00669000
  );                                                                        00670000
  DOUBLE(TEMPYH,TEMPYL,DTH,DTL,x,VYH[1],VYL[1],+,+,VYH[0],VYL[0]00671000
  );                                                                        00672000
                                                                            00673000
END #;                                                                      00674000
DEFINE ADAMSMOULXY =                                                        00675000
BEGIN                                                                       00676000
  TEMPXH :=TEMPXL :=TEMPYH :=TEMPYL :=0;                                    00677000
  FOR R :=0 STEP 1 UNTIL Q DO                                               00678000
```

```
    BEGIN                                                                    00679000
      DOUBLE(VXH[R],VXL[R],BSHQR,BSLQR,×,TEMPXH,TEMPXL,+,←,TEMPXH            00680000
      ,TEMPXL);                                                              00681000
      DOUBLE(VYH[R],VYL[R],BSHQR,BSLQR,×,TEMPYH,TEMPYL,+,←,TEMPYH            00682000
      ,TEMPYL);                                                              00683000
                                                                             00684000
    END;                                                                     00685000
    DOUBLE(TEMPXH,TEMPXL,DTH,DTL,×,XH[1],XL[1],+,←,XH[0],XL[0]);            00686000
    DOUBLE(TEMPYH,TEMPYL,DTH,DTL,×,YH[1],YL[1],+,←,YH[0],YL[0]);            00687000
                                                                             00688000
  END #;                                                                     00689000
  DEFINE DOUBLESTEPSIZE =                                                    00690000
  BEGIN                                                                      00691000
    IF PRDUBL ="YES"THEN FLAG :=1;                                           00692000
    FOR J :=1 STEP 1 UNTIL QD DO                                             00693000
    BEGIN                                                                    00694000
      J2 :=J +J;                                                             00695000
      XH[J]:=XH[J2];                                                         00696000
      XL[J]:=XL[J2];                                                         00697000
      YH[J]:=YH[J2];                                                         00698000
      YL[J]:=YL[J2];                                                         00699000
      VXH[J]:=VXH[J2];                                                       00700000
      VXL[J]:=VXL[J2];                                                       00701000
      VYH[J]:=VYH[J2];                                                       00702000
      VYL[J]:=VYL[J2];                                                       00703000
      FXH[J]:=FXH[J2];                                                       00704000
      FXL[J]:=FXL[J2];                                                       00705000
      FYH[J]:=FYH[J2];                                                       00706000
      FYL[J]:=FYL[J2];                                                       00707000
                                                                             00708000
    END;                                                                     00709000
    FOR J :=QD +1 STEP 1 UNTIL QPRIME DO VXH[J]:=4.1@68;                     00710000
    Q :=(QMAX+QMIN)/2;                                                       00711000
    DOUBLE(TH,TL,DTH,DTL,-,←,TH,TL);                                         00712000
    DOUBLE(DTH,DTL,DTH,DTL,+,←,DTH,DTL);                                     00713000
                                                                             00714000
  END #;                                                                     00715000
  DEFINE STORE =                                                             00716000
  BEGIN                                                                      00717000
    XPPH :=XH[0];                                                            00718000
```

```
    XPPL :=XL[0];                                                          00719000
    YPPH :=YH[0];                                                          00720000
    YPPL :=YL[0];                                                          00721000
    VXPPH :=VXH[0];                                                        00722000
    VXPPL :=VXL[0];                                                        00723000
    VYPPH :=VYH[0];                                                        00724000
    VYPPL :=VYL[0];                                                        00725000
    FXPPH :=FXH[0];                                                        00726000
    FXPPL :=FXL[0];                                                        00727000
    FYPPH :=FYH[0];                                                        00728000
    FYPPL :=FYL[0];                                                        00729000
                                                                           00730000
  END #;                                                                   00731000
  DEFINE INTERPOLATE =                                                     00732000
  BEGIN                                                                    00733000
    FOR L:=QH STEP 1 UNTIL QT DO                                           00734000
    BEGIN                                                                  00735000
      XXH[L]:=XXL[L]:=YYH[L]:=YYL[L]:=VVXH[L]:=VVYH[L]:=FFXH[L]:           00736000
      =FFYH[L]:=VVXL[L]:=VVYL[L]:=FFXL[L]:=FFYL[L]:=0;                     00737000
      FOR S :=L-QH STEP 1 UNTIL QS+L-QH DO                                 00738000
      BEGIN                                                                00739000
        G:=S-L +QH;                                                        00740000
        K :=QH;                                                            00741000
        DOUBLE(XH[S+1],XL[S+1],AH[K,G],AL[K,G],×,XXH[L],XXL[L],+,←00742000
        ,XXH[L],XXL[L]);                                                   00743000
        DOUBLE(YH[S+1],YL[S+1],AH[K,G],AL[K,G],×,YYH[L],YYL[L],+,←00744000
        ,YYH[L],YYL[L]);                                                   00745000
        DOUBLE(VXH[S+1],VXL[S+1],AH[K,G],AL[K,G],×,VVXH[L],VVXL[L]00746000
        ,+,←,VVXH[L],VVXL[L]);                                             00747000
        DOUBLE(VYH[S+1],VYL[S+1],AH[K,G],AL[K,G],×,VVYH[L],VVYL[L]00748000
        ,+,←,VVYH[L],VVYL[L]);                                             00749000
        DOUBLE(FXH[S+1],FXL[S+1],AH[K,G],AL[K,G],×,FFXH[L],FFXL[L]00750000
        ,+,←,FFXH[L],FFXL[L]);                                             00751000
        DOUBLE(FYH[S+1],FYL[S+1],AH[K,G],AL[K,G],×,FFYH[L],FFYL[L]00752000
        ,+,←,FFYH[L],FFYL[L]);                                             00753000
                                                                           00754000
      END;                                                                 00755000
                                                                           00756000
    END;                                                                   00757000
                                                                           00758000
```

```
END #;                                                                  00759000
PROCEDURE HALVESTEPSIZE ;                                               00760000
                                                                        00761000
BEGIN                                                                   00762000
  IF PRHALV ="YES"THEN FLAG :=1;                                        00763000
  INTERPOLATE;                                                          00764000
  FOR J :=QT STEP -1 UNTIL QM+1,J :=QH STEP +1 UNTIL QM DO              00765000
  BEGIN                                                                 00766000
    J2 :=2×(J -QH)+1;                                                   00767000
    XH[J2+1]:=XH[J+1];                                                  00768000
    XL[J2+1]:=XL[J+1];                                                  00769000
    YH[J2+1]:=YH[J+1];                                                  00770000
    YL[J2+1]:=YL[J+1];                                                  00771000
    VXH[J2+1]:=VXH[J+1];                                                00772000
    VXL[J2+1]:=VXL[J+1];                                                00773000
    VYH[J2+1]:=VYH[J+1];                                                00774000
    VYL[J2+1]:=VYL[J+1];                                                00775000
    FXH[J2+1]:=FXH[J+1];                                                00776000
    FXL[J2+1]:=FXL[J+1];                                                00777000
    FYH[J2+1]:=FYH[J+1];                                                00778000
    FYL[J2+1]:=FYL[J+1];                                                00779000
    XH[J2]:=XXH[J];                                                     00780000
    XL[J2]:=XXL[J];                                                     00781000
    YH[J2]:=YYH[J];                                                     00782000
    YL[J2]:=YYL[J];                                                     00783000
    VXH[J2]:=VVXH[J];                                                   00784000
    VXL[J2]:=VVXL[J];                                                   00785000
    VYH[J2]:=VVYH[J];                                                   00786000
    VYL[J2]:=VVYL[J];                                                   00787000
    FXH[J2]:=FFXH[J];                                                   00788000
    FXL[J2]:=FFXL[J];                                                   00789000
    FYH[J2]:=FFYH[J];                                                   00790000
    FYL[J2]:=FFYL[J];                                                   00791000
                                                                        00792000
  END;                                                                  00793000
  FOR J :=2×(QT-QH)+3 STEP 1 UNTIL QPRIME DO VXH[J]:=4.1@68;            00794000
  Q :=(QMIN +QMAX)DIV 2;                                                00795000
  DOUBLE(DTH,DTL,2,0,/,←,DTH,DTL);                                      00796000
  DOUBLE(-DTH,DTL,QH+QH-1,0,×,TH,TL,+,←,TH,TL);                         00797000
  NS :=NS -QH +1;                                                       00798000
```

```
    IF DTH =0 THEN GO TO AGAIN;                                          00799000
                                                                         00800000
  END ;                                                                  00801000
  COMMENT END OF DECLARATIONS;                                           00802000
  QH :=QS DIV 2;                                                         00803000
  QT :=QMAX DIV 2 +QS ;                                                  00804000
  QD :=(QMAX +1)×2;                                                      00805000
  DSTRTH :=DOFRKH;                                                       00806000
  DSTRTL :=DOFRKL;                                                       00807000
  QM :=(QT +QH)DIV 2 -1;                                                 00808000
  L9:COMPUTEA;                                                           00809000
  Q :=TH/DTH;                                                            00810000
  FOR J :=Q+2 STEP 1 UNTIL QPRIME DO                                     00811000
  BEGIN                                                                  00812000
    VXH[J]:=4.1@68;                                                      00813000
    VXL[J]:=0;                                                           00814000
                                                                         00815000
  END;                                                                   00816000
  IF Q>QMAX THEN Q :=QMAX;                                               00817000
  L7:ADAMSBASHV;                                                         00818000
  ADAMSMOULXY;                                                           00819000
  FXFYCALC;                                                              00820000
  L5:STORETOP;                                                           00821000
  L10:STORE;                                                             00822000
  ADAMSMOULV;                                                            00823000
  ADAMSMOULXY;                                                           00824000
  FXFYCALC;                                                              00825000
  IF ITER ="YES"THEN                                                     00826000
  BEGIN                                                                  00827000
    DOUBLE(VXPPH,VXPPL,VXH[0],VXL[0],-,←,TEMPXH,TEMPXL);                 00828000
    DOUBLE(VYPPH,VYPPL,VYH[0],VYL[0],-,←,TEMPYH,TEMPYL);                 00829000
    IF ABS(TEMPXH)>EMIN OR ABS(TEMPYH)>EMIN THEN                         00830000
    BEGIN                                                                00831000
      GO TO L10;                                                         00832000
                                                                         00833000
    END;                                                                 00834000
                                                                         00835000
  END;                                                                   00836000
  DOUBLE(VXH[0],VXL[0],VXPH,VXPL,-,←,TEMPXH,TEMPXL);                     00837000
  DOUBLE(VYH[0],VYL[0],VYPH,VYPL,-,←,TEMPYH,TEMPYL);                     00838000
```

```
      IF ABS(TEMPXH)≤EMIN AND ABS(TEMPYH)≤EMIN THEN GO TO TOOSMALL;    00839000
      IF ABS(TEMPXH)>EMAX OR ABS(TEMPYH)>EMAX THEN GO TO TOOBIG;       00840000
      COMMENT HERE IF ERROR OK;                                        00841000
      Q :=Q+1;                                                         00842000
      COUNT :=3;                                                       00843000
      IF Q>QMAX THEN Q :=QMAX;                                         00844000
      L6:DOUBLE(TH,TL,DTH,DTL,+,←,TH,TL);                              00845000
      NS :=NS+1;                                                       00846000
      PUSHDOWN;                                                        00847000
      PRINT;                                                           00848000
      IF SEARCH ="YES"THEN IF ABS(TH -PERIODH/2)<.0090÷DTH THEN        00849000
      BEGIN                                                            00850000
        DOUBLE(YH[1],YL[1],YH[2],YL[2],-,←,DYH,DYL);                   00851000
        DOUBLE(DYH,DYL,DYH,DYL,+,YH[1],YL[1],+,←,TEMPYH,TEMPYL);       00852000
        IF 0 <M×TEMPYH THEN SEARCHHALVEPOINT;                          00853000
                                                                       00854000
      END;                                                             00855000
      IF (DTH+DTH+TH)MOD PERIODH<DTH+DTH THEN IF DTH >DOFRKH THEN      00856000
      HALVESTEPSIZE ELSE                                               00857000
      BEGIN                                                            00858000
        DOUBLE(PERIODH,PERIODL,TH,TL,-,←,DTH,DTL);                     00859000
        L8:DOFRKH :=DTH;                                               00860000
        DOFRKL :=DTL;                                                  00861000
        RK;                                                            00862000
        DOUBLE(TH,TL,DTH,DTL,+,←,TH,TL);                               00863000
        NS :=NS+1;                                                     00864000
        PUSHDOWN;                                                      00865000
        FLAG :=1;                                                      00866000
        PRINT;                                                         00867000
        DTH :=-YH[1]/VYH[1];                                           00868000
                                                                       00869000
        BEGIN                                                          00870000
          FORMAT FMD(/"DX =",E10.2,"   DY =",E10.2,"   DVX =",E10.2," 00871000
DVY =",E10.2 /X14,"   DT =",E10.2,"   DVX =",E10.2 //);               00872000
          DOUBLE(XH[1],XL[1],XINH,XINL,-,←,TEMPXH,TEMPXL);             00873000
          DOUBLE(VYH[1],VYL[1],VYINH,VYINL,-,←,TEMPYH,TEMPYL);         00874000
          WRITE(FILEOUT,FMD,TEMPXH,YH[1],VXH[1],TEMPYH,DTH,VXH[1]+FXH 00875000
          [1]×DTH );                                                   00876000
                                                                       00877000
        END;                                                           00878000
```

```
          GO TO AGAIN;                                                    00879000
                                                                          00880000
        END;                                                              00881000
        GO TO L7;                                                         00882000
        TOOSMALL:IF Q>QMIN THEN                                           00883000
        BEGIN                                                             00884000
          Q :=Q-1;                                                        00885000
          GO TO L6;                                                       00886000
                                                                          00887000
        END;                                                              00888000
        IF 0<(COUNT :=COUNT-1)THEN GO TO L6;                              00889000
        IF VXH[QPRIME]<4@68 THEN DOUBLESTEPSIZE;                          00890000
        GO TO L6;                                                         00891000
        TOOBIG:IF Q<QMAX THEN                                             00892000
        BEGIN                                                             00893000
          Q :=Q+1;                                                        00894000
          GO TO L5;                                                       00895000
                                                                          00896000
        END;                                                              00897000
        HALVESTEPSIZE;                                                    00898000
        GO TO L7;                                                         00899000
        FINISH:                                                           00900000
      END;                                                                00901000
                                                                          00902000
    END;                                                                  00903000
    THEEND:                                                               00904000
  END.                                                                    00905000
 30         END-OF-FILE CARD
 LABEL    00000000FILEIN 0010000001
* COMMENT  HERE ARE THE LISTINGS OF THE ADAMS COEFFICIENTS    BETASTAR[ Q , R ]
 +0000000000000000001.00000000000000000000000000000000          *       0   0
 +0000000000000000000.50000000000000000000000000000000          *       1   0
 +0000000000000000000.50000000000000000000000000000000          *       1   1
 +0000000000000000000.41666666666666666666666666666667          *       2   0
 +0000000000000000000.66666666666666666666666666666666          *       2   1
 -0000000000000000000.08333333333333333333333333333333          *       2   2
 +0000000000000000000.37499999999999999999999999999999          *       3   0
 +0000000000000000000.79166666666666666666666666666670          *       3   1
 -0000000000000000000.20833333333333333333333333333337          *       3   2
 +0000000000000000000.04166666666666666666666666666668          *       3   3
```

+0000000000000000000.348611111111111111111111111110	*	4	0
+0000000000000000000.897222222222222222222222222226	*	4	1
-0000000000000000000.366666666666666666666666666671	*	4	2
+0000000000000000000.147222222222222222222222222224	*	4	3
-0000000000000000000.026388888888888888888888888889	*	4	4
+0000000000000000000.329861111111111111111111111110	*	5	0
+0000000000000000000.990972222222222222222222222226	*	5	1
-0000000000000000000.554166666666666666666666666671	*	5	2
+0000000000000000000.334722222222222222222222222224	*	5	3
-0000000000000000000.120138888888888888888888888889	*	5	4
+0000000000000000000.018750000000000000000000000000	*	5	5
+0000000000000000000.315591931216931216931216931215	*	6	0
+0000000000000000001.076587301587301587301587301596	*	6	1
-0000000000000000000.768204365079365079365079365096	*	6	2
+0000000000000000000.620105820105820105820105820124	*	6	3
-0000000000000000000.334176587301587301587301587314	*	6	4
+0000000000000000000.104365079365079365079365079370	*	6	5
-0000000000000000000.014269179894179894179894179895	*	6	6
+0000000000000000000.304224537037037037037037037033	*	7	0
+0000000000000000001.156159060846560846560846560870	*	7	1
-0000000000000000001.006919642857142857142857142918	*	7	2
+0000000000000000001.017964616402116402116402116494	*	7	3
-0000000000000000000.732035383597883597883597883684	*	7	4
+0000000000000000000.343080357142857142857142857192	*	7	5
-0000000000000000000.093840939153439153439153439169	*	7	6
+0000000000000000000.011367394179894179894179894182	*	7	7
+0000000000000000000.294868000440917107583774250436	*	8	0
+0000000000000000001.231011353615520282186948853646	*	8	1
-0000000000000000001.268902667548500881834215167634	*	8	2
+0000000000000000001.541930665784832451499118165926	*	8	3
-0000000000000000001.386992945326278659611992945474	*	8	4
+0000000000000000000.867046406525573192239858906624	*	8	5
-0000000000000000000.355823963844797178130511463885	*	8	6
+0000000000000000000.086219686948853615520282186958	*	8	7
-0000000000000000000.009356536596119929453262786597	*	8	8
+0000000000000000000.286975446428571428571428571422	*	9	0
+0000000000000000001.302044339726631393298059964772	*	9	1
-0000000000000000001.553034611992945326278659612138	*	9	2
+0000000000000000002.204905202821869488536155203102	*	9	3
-0000000000000000002.381454750881834215167548501238	*	9	4

+0000000000000000001.861508212081128747795414462388	*	9	5
-0000000000000000001.018798500881834215167548501061	*	9	6
+0000000000000000000.370351631393298059964726631462	*	9	7
-0000000000000000000.080389522707231040564373897723	*	9	8
+0000000000000000000.007892554012345679012345679014	*	9	9
+0000000000000000000.280189596443936721714499492270	*	10	0
+0000000000000000001.369902839572978461867350756292	*	10	1
-0000000000000000001.858397861301507134840468173978	*	10	2
+0000000000000000003.019207200978034311367644701342	*	10	3
-0000000000000000003.806483247655122655122655123158	*	10	4
+0000000000000000003.571542408209074875741542408692	*	10	5
-0000000000000000002.443826997655122655122655122981	*	10	6
+0000000000000000001.184653629549462882796216129702	*	10	7
-0000000000000000000.385752772015792849126182459563	*	10	8
+0000000000000000000.075751053858692747581636470534	*	10	9
-0000000000000000000.006785849984634706856929079152	*	10	10
+0000000000000000000.274265540031599059376837154606	*	11	0
+0000000000000000001.435067460108692747581636470596	*	11	1
-0000000000000000002.184220963980078563411896745498	*	11	2
+0000000000000000003.996676509013748597081930415902	*	11	3
-0000000000000000005.761421863726551226551226552278	*	11	4
+0000000000000000006.308456470709074875741542409460	*	11	5
-0000000000000000005.180741060155122655122655123749	*	11	6
+0000000000000000003.139592245620891454224787558822	*	11	7
-0000000000000000001.363222080051507134840468174123	*	11	8
+0000000000000000000.401574156537264176153065042054	*	11	9
-0000000000000000000.071950470520348992571214793456	*	11	10
+0000000000000000000.005924056412337662337662337664	*	11	11
+0000000000000000000.269028846773648774310149971515	*	12	0
+0000000000000000001.497907779204096168381882667688	*	12	1
-0000000000000000002.529842719004797377813250829504	*	12	2
+0000000000000000005.148749025762811311753110695922	*	12	3
-0000000000000000008.353585026411942334561382182323	*	12	4
+0000000000000000010.455917531005700648557791417532	*	12	5
-0000000000000000010.019445630501186056741612299833	*	12	6
+0000000000000000007.287053305917517227041036566894	*	12	7
-0000000000000000003.955385242736898242850623804168	*	12	8
+0000000000000000001.553646673286326890824245322074	*	12	9
-0000000000000000000.417572225545067806972568877462	*	12	10
+0000000000000000000.068764375507741083137908534756	*	12	11

-0000000000000000000.00523669325795028506668718309 1	*	12	12
+0000000000000000000.26435134836660650979434048215 8	*	13	0
+0000000000000000001.55871525849564560708740602932 9	*	13	1
-0000000000000000002.89468759475409401004639099935 0	*	13	2
+0000000000000000006.48651357017689896327462465202 4	*	13	3
-0000000000000000011.69799638744716146336516707257 8	*	13	4
+0000000000000000016.47585798086909508040460421999 1	*	13	5
-0000000000000000018.04603289698571196587069603644 5	*	13	6
+0000000000000000015.31364057240204313617012030350 6	*	13	7
-0000000000000000009.97532569260029267469743660662 7	*	13	8
+0000000000000000004.89805803432154601962803021232 9	*	13	9
-0000000000000000001.75533676995915545849408283356 4	*	13	10
+0000000000000000000.43360925125703771537104870460 2	*	13	11
-0000000000000000000.06604417254949972377221054473 2	*	13	12
+0000000000000000000.00467749840704226451580948935 7	*	13	13
+0000000000000000000.26013639612760103693745669053 0	*	14	0
+0000000000000000001.61772458984172222708377911212 1	*	14	1
-0000000000000000003.27824824850359204002281603749 8	*	14	2
+0000000000000000008.02075618517489108318032480461 6	*	14	3
-0000000000000000015.91716357869163979310584249220 6	*	14	4
+0000000000000000024.91419236335805173988595505924 7	*	14	5
-0000000000000000030.70353447071914695509272229532 9	*	14	6
+0000000000000000029.77935665666882598099529317080 2	*	14	7
-0000000000000000022.63282726633372766391946286551 1	*	14	8
+0000000000000000013.33639241681050267910938105158 5	*	14	9
-0000000000000000005.97450396120363378823475825319 2	*	14	10
+0000000000000000001.96785186625502983527674885719 4	*	14	11
-0000000000000000000.44960482629899775374863558288 0	*	14	12
+0000000000000000000.06368682975311888451218257214 9	*	14	13
-0000000000000000000.00421495223900547285688379162 8	*	14	14
+0000000000000000000.25630949657438915251415251413 6	*	15	0
+0000000000000000001.67512808313990049343334175803 1	*	15	1
-0000000000000000003.68007270159083990446975455886 8	*	15	2
+0000000000000000009.76199548188629849578372506388 6	*	15	3
-0000000000000000021.14088146882586203091604327001 6	*	15	4
+0000000000000000036.40637172165334066306839677042 9	*	15	5
-0000000000000000049.85716673454462849373012514729 9	*	15	6
+0000000000000000054.40545528158730224495766826619 2	*	15	7
-0000000000000000047.25892589125220392788183796090 1	*	15	8
+0000000000000000032.49002468063598421774678390355 5	*	15	9

```
-0000000000000000017.46668331949892271141719996437 4          *      15  10
+0000000000000000007.19156975638925207308694963500 4          *      15  11
-0000000000000000002.19084412301040516635203584215 0          *      15  12
+0000000000000000000.46551128284036674895912109351 9          *      15  13
-0000000000000000000.06161844553718373920644643753 8          *      15  14
+0000000000000000000.00382689955321188442330417639 4          *      15  15
+0000000000000000000.25281214672903923486004312160 1          *      16   0
+0000000000000000001.73108568066549917589909203859 1          *      16   1
-0000000000000000004.09975468303283002296288166306 8          *      16   2
+0000000000000000011.72051139528225238208498488348 6          *      16   3
-0000000000000000027.50605818736271216139513768371 6          *      16   4
+0000000000000000051.68279584614178097621822336330 9          *      16   5
-0000000000000000077.86394429610676906783814056757 9          *      16   6
+0000000000000000094.41513751239036020796911886659 2          *      16   7
-0000000000000000092.26981840090564413626971988635 1          *      16   8
+0000000000000000072.49970691143904218075823450395 5          *      16   9
-0000000000000000045.47346088106106328552521538465 4          *      16  10
+0000000000000000022.46799388087769238623677622788 4          *      16  11
-0000000000000000008.55602084154725529683113025585 0          *      16  12
+0000000000000000002.42402719623632063526038091311 9          *      16  13
-0000000000000000000.48130042697917385769957354173 8          *      16  14
+0000000000000000000.05978449707881056688905445695 4          *      16  15
-0000000000000000000.00349734984534991765410939253 5          *      16  16
*  COMMENT  HERE ARE THE LISTINGS OF THE ADAMS COEFFICIENTS      BETA[ Q , R ]
+0000000000000000001.00000000000000000000000000000000 0       *       0   0
+0000000000000000001.50000000000000000000000000000000 0       *       1   0
-0000000000000000000.50000000000000000000000000000000 0       *       1   1
+0000000000000000001.91666666666666666666666666666667 9       *       2   0
-0000000000000000001.33333333333333333333333333333334 9       *       2   1
+0000000000000000000.41666666666666666666666666666667 9       *       2   2
+0000000000000000002.29166666666666666666666666666666 9       *       3   0
-0000000000000000002.45833333333333333333333333333331 9       *       3   1
+0000000000000000001.54166666666666666666666666666664 9       *       3   2
-0000000000000000000.37499999999999999999999999999999 9       *       3   3
+0000000000000000002.64027777777777777777777777777776 9       *       4   0
-0000000000000000003.85277777777777777777777777777771 9       *       4   1
+0000000000000000003.03333333333333333333333333333324 9       *       4   2
-0000000000000000001.76944444444444444444444444444439 9       *       4   3
+0000000000000000000.34861111111111111111111111111110 9       *       4   4
+0000000000000000002.97013888888888888888888888888886 9       *       5   0
```

-0000000000000000005.502083333333333333333333333321	*	5	1
+0000000000000000006.931944444444444444444444444424	*	5	2
-0000000000000000005.068055555555555555555555555539	*	5	3
+0000000000000000001.997916666666666666666666666660	*	5	4
-0000000000000000000.329861111111111111111111111110	*	5	5
+0000000000000000003.285730820105820105820105820101	*	6	0
-0000000000000000007.395634920634920634920634920611	*	6	1
+0000000000000000011.665823412698412698412698412649	*	6	2
-0000000000000000011.379894179894179894179894179839	*	6	3
+0000000000000000006.731795634920634920634920634885	*	6	4
-0000000000000000002.223412698412698412698412698400	*	6	5
+0000000000000000000.315591931216931216931216931215	*	6	6
+0000000000000000003.589955357142857142857142857134	*	7	0
-0000000000000000009.525206679894179894179894179842	*	7	1
+0000000000000000018.054538690476190476190476190342	*	7	2
-0000000000000000022.027752976190476190476190475994	*	7	3
+0000000000000000017.379654431216931216931216931040	*	7	4
-0000000000000000008.612127976190476190476190476093	*	7	5
+0000000000000000002.445163690476190476190476190446	*	7	6
-0000000000000000000.304224537037037037037037037033	*	7	7
+0000000000000000003.884823357583774250440917107570	*	8	0
-0000000000000000011.884150683421516754850088183330	*	8	1
+0000000000000000026.310842702821869488536155202550	*	8	2
-0000000000000000038.540361000881834215167548500410	*	8	3
+0000000000000000038.020414462081128747795414461560	*	8	4
-0000000000000000025.124736000881834215167548500509	*	8	5
+0000000000000000010.701467702821869488536155202654	*	8	6
-0000000000000000002.663168540564373897707291040521	*	8	7
+0000000000000000000.294868000440917107583774250436	*	8	8
+0000000000000000004.171798804012345679012345678992	*	9	0
-0000000000000000014.466929701278659611992945326128	*	9	1
+0000000000000000036.641958774250440917107583773742	*	9	2
-0000000000000000062.646298500881834215167548499858	*	9	3
+0000000000000000074.179320712081128747795414460732	*	9	4
-0000000000000000061.283642250881834215167548499681	*	9	5
+0000000000000000034.807405202821869488536155202102	*	9	6
-0000000000000000012.994284611992945326278659611713	*	9	7
+0000000000000000002.877647018298059964726631393234	*	9	8
-0000000000000000000.286975446428571428571428571422	*	9	9
+0000000000000000004.451988400456282400726845171262	*	10	0

-0000000000000000017.268825665718026829137940248828	*	10	1
+0000000000000000049.250490614227593394260060925892	*	10	2
-0000000000000000096.269050074154240820907487572258	*	10	3
+0000000000000000133.019135965307840307840307837432	*	10	4
-0000000000000000131.891420554753888087221420551721	*	10	5
+0000000000000000093.647220456048581048581048578802	*	10	6
-0000000000000000046.617036185265351932018598684113	*	10	7
+0000000000000000015.486178858275212441879108545384	*	10	8
-0000000000000000003.088871410867938645716423494122	*	10	9
+0000000000000000000.280189596443936721714499492270	*	10	10
+0000000000000000004.726253940487881460103682325868	*	11	0
-0000000000000000020.285746606065616482283148949494	*	11	1
+0000000000000000064.335095315965541659986104429222	*	11	2
-0000000000000000141.522864179368085618085618082248	*	11	3
+0000000000000000223.526764175735529902196568857412	*	11	4
-0000000000000000258.602100049352653519320185979693	*	11	5
+0000000000000000220.357899950647346480679814006774	*	11	6
-0000000000000000137.124664395693041526374859704093	*	11	7
+0000000000000000060.739992963489057239057239055374	*	11	8
-0000000000000000018.173476112605886911442466997452	*	11	9
+0000000000000000003.297110536791526374859708192936	*	11	10
-0000000000000000000.274265540031599059376837154606	*	11	11
+0000000000000000004.995282787261530234413832297383	*	12	0
-0000000000000000023.514092767349401774004948607674	*	12	1
+0000000000000000082.090999203026360764456002549212	*	12	2
-0000000000000000200.709210469570815966318611815548	*	12	3
+0000000000000000356.696043328691673185720804757337	*	12	4
-0000000000000000471.672946694082482772958963419573	*	12	5
+0000000000000000468.940554369498813943258387686634	*	12	6
-0000000000000000350.195511040422870780013637143973	*	12	7
+0000000000000000193.909272116445200522581474955299	*	12	8
-0000000000000000077.359822402808617259675460730752	*	12	9
+0000000000000000021.053014423852345479329606312926	*	12	10
-0000000000000000003.502611701315384351098636812786	*	12	11
+0000000000000000000.269028846773648774310149971515	*	12	12
+0000000000000000005.259634135628136744208172779541	*	13	0
-0000000000000000026.950660296115286401331374875728	*	13	1
+0000000000000000102.710404375621668528414560157536	*	13	2
-0000000000000000276.313696102420277767499989712736	*	13	3
+0000000000000000545.707257410815327688674249500307	*	13	4

-0000000000000000811.893132041905060878275163956919	*	13	5
+0000000000000000922.567468166595584750346655069762	*	13	6
-0000000000000000803.822424837519641587101904527101	*	13	7
+0000000000000000534.129457464267778627897675492645	*	13	8
-0000000000000000266.371036484932271762628905473722	*	13	9
+0000000000000000096.657500056701807280510984210114	*	13	10
-0000000000000000024.122016873910692115057194421110	*	13	11
+0000000000000000003.705596375539533401636576239569	*	13	12
-0000000000000000000.264351348366606509794340482158	*	13	13
+0000000000000000005.519770531755737781145629470071	*	14	0
-0000000000000000030.592569841901700918455768543148	*	14	1
+0000000000000000126.382816423233362889723118995766	*	14	2
-0000000000000000371.003344292867055212734225065656	*	14	3
+0000000000000000806.103789934543965663068396720837	*	14	4
-0000000000000001332.686197089362336827063458397979	*	14	5
+0000000000000001703.757065737781498673529096731352	*	14	6
-0000000000000001696.610536347446400356453266426061	*	14	7
+0000000000000001315.319055035453692551080117154235	*	14	8
-0000000000000000787.164101532389547711417199914782	*	14	9
+0000000000000000357.054032580430445254905131430644	*	14	10
-0000000000000000118.811665064357469560291429774030	*	14	11
+0000000000000000027.378008423151227762945135077799	*	14	12
-0000000000000000003.906260894153021026918734149578	*	14	13
+0000000000000000000.260136396127601036937456690530	*	14	14
+0000000000000000005.776080028330126933659781984207	*	15	0
-0000000000000000034.437212290517538206168056255188	*	15	1
+0000000000000000153.295313563544223903709132980046	*	15	2
-0000000000000000487.624165234214119606673618997536	*	15	3
+0000000000000001155.966252758585158844886578516477	*	15	4
-0000000000000002102.383615302252961827063458348387	*	15	5
+0000000000000002986.586096092599207006862429982032	*	15	6
-0000000000000003345.962146803640596785024694891221	*	15	7
+0000000000000002964.670665491647888979651545619395	*	15	8
-0000000000000002069.993131887207256044750533165462	*	15	9
+0000000000000001126.751450793321070254905131381052	*	15	10
-0000000000000000468.674127888398662742109611569670	*	15	11
+0000000000000000143.998829364498292156884529009679	*	15	12
-0000000000000000030.818758034463882040904748133858	*	15	13
+0000000000000000004.104778844743438324649744402570	*	15	14
-0000000000000000000.256309496574389152514152514136	*	15	15

+0000000000000000006.028892175059166168519825105808	*	16	0
-0000000000000000038.482206638182165963928746200804	*	16	1
+0000000000000000183.632771171028932086914307572166	*	16	2
-0000000000000000629.198967402476091128297767094096	*	16	3
+0000000000000001616.084359805436566290165059830297	*	16	4
-0000000000000003206.667072214696339695731813501555	*	16	5
+0000000000000005011.105767098745399766087747762840	*	16	6
-0000000000000006238.133105383849443583918006006661	*	16	7
+0000000000000006218.362993894382841628406520624265	*	16	8
-0000000000000004962.164090467416102843643844280902	*	16	9
+0000000000000003151.271121799467263014130449161860	*	16	10
-0000000000000001572.957584800842040610777966722838	*	16	11
+0000000000000000604.116936411349699602163010323499	*	16	12
-0000000000000000172.393560202725853562528896230418	*	16	13
+0000000000000000034.442236452228146507854918994690	*	16	14
-0000000000000000004.301303844239016910274842459752	*	16	15
+0000000000000000000.252812146729039234860043121601	*	16	16

D. The Method of Runge-Kutta-Fehlberg

1. Introduction

This section of the report presents the results of a study of Runge-Kutta type formulas of high-order accuracy developed by Erwin Fehlberg [18], [19] and applied to systems of second-order differential equations.

The presentation of these results is broken down into two parts. The first part is a discussion of the theory and the second part is the double-precision program which was written for the Burroughs B-5000.

2. Theory

In a rotating coordinate system, the equations of motion of a space vehicle in the Earth-Moon System are:

$$\ddot{x} = f(x,y,\dot{y}) = x + 2\dot{y} - \mu' \cdot \frac{x+\mu}{((x+\mu)^2+y^2)^{3/2}} - \mu \cdot \frac{x-\mu'}{((x-\mu')^2+y^2)^{3/2}} ,$$

$$\ddot{y} = g(x,y,\dot{x}) = y - 2\dot{x} - \mu' \cdot \frac{y}{((x+\mu)^2+y^2)^{3/2}} - \mu \cdot \frac{y}{((x-\mu')^2+y^2)^{3/2}} , \qquad (2\text{-}1)$$

where $\mu' = 1 - \mu$ with μ representing the relative mass of the Moon.

Introducing into (2-1) the following four auxiliary functions:

$$r^2 = (x+\mu)^2+y^2 \quad , \quad s^2 = (x-\mu')^2+y^2 \quad ,$$

$$u = \mu'/r^3 \quad , \quad v = \mu/s^3 \quad , \qquad (2\text{-}2)$$

system (2-1) becomes

$$\ddot{x} = x + 2\dot{y} - u(x+\mu) - v(x-\mu') \quad ,$$

$$\ddot{y} = y - 2\dot{x} - uy - vy \quad , \tag{2-3}$$

$$r\dot{u} + 3u\dot{r} = 0 \quad ,$$

$$s\dot{v} + 3v\dot{s} = 0 \quad .$$

Substituting the following power series expansions, in terms of $h_j = (t - t_j)$ about the point t_j, into (2-3)

$$x = \sum_{i=0} X_i \cdot (h_j)^i \quad , \quad y = \sum_{i=o} Y_i \cdot (h_j)^i \quad ,$$

$$u = \sum_{i=0} U_i \cdot (h_j)^i \quad , \quad v = \sum_{i=0} V_i \cdot (h_j)^i \quad , \tag{2-4}$$

$$r = \sum_{i=0} R_i \cdot (h_j)^i \quad , \quad s = \sum_{i=0} S_i \cdot (h_j)^i \quad ,$$

and equating the coefficients of like terms, the following recurrence formulas for the power series coefficients in (2-4) result:

$$R_n = \left\{ \sum_{i=0}^{n} X_i X_{n-i} + 2\mu X_n + \sum_{i=0}^{n} Y_i Y_{n-i} - \sum_{i=1}^{n-1} R_i R_{n-i} \right\} / 2R_0 \quad ,$$

$$S_n = \left\{ \sum_{i=0}^{n} X_i X_{n-i} - 2\mu' X_n + \sum_{i=0}^{n} Y_i Y_{n-i} - \sum_{i=1}^{n-1} S_i S_{n-i} \right\} / 2S_0 \quad ,$$

$$U_n = \left\{ -3 \sum_{i=1}^{n} iR_i U_{n-i} - \sum_{i=1}^{n-1} iU_i R_{n-i} \right\} / nR_0 \quad ,$$

$$V_n = \left\{ -3 \sum_{i=1}^{n} iS_i V_{n-i} - \sum_{i=1}^{n-1} iV_i S_{n-i} \right\} / nS_0 \quad ,$$

$$X_{n+1} = \left\{ X_{n-1} + 2nY_n = \mu U_{n-1} + \mu' \quad V_{n-1} - \sum_{i=0}^{n-1} (U_i - V_i) X_{n-1-i} \right\} / n(n+1) \quad ,$$

$$Y_{n+1} = \left\{ Y_{n-1} - 2nX_n - \sum_{i=0}^{n-1} (U_i - V_i) \quad Y_{n-1-i} \right\} / n(n+1) \quad , \qquad (2\text{-}5)$$

where $n = 1,2,3, \ldots$. For $n = 1$, the last sum of the first four equations in (2-5) must be omitted.

At the start of the j^{th} integration step, the initial values for the step determine the initial coefficients as follows:

$$X_0 = x_j \quad , \quad Y_0 = y_j \quad ,$$

$$X_1 = \dot{x}_j \quad , \quad Y_1 = \dot{y}_j \quad , \qquad (2\text{-}6)$$

$$R_0 = \left[(x_j + \mu)^2 + y_j^2 \right]^{1/2} \quad , \quad S_0 = \left[(x_j - \mu')^2 + y_j^2 \right]^{1/2} \quad ,$$

$$U_0 = \mu' / R_0^3 \quad , \quad V_0 = \mu / S_0^3 \quad .$$

After computing, through formulas (2-5), a sufficiently large number, m, of consecutive derivatives of x and y with respect to t, in the form of the coefficients X_n and Y_n, we then apply the Runge-Kutta procedure, described by Fehlberg [20], to obtain (m+4)-th-order accuracy.

Introducing new variables x_T, y_T, $\dot{x}_T$, and $\dot{y}_T$ by the following transformations:

$$x_T = x - \sum_{i=1}^{m+2} X_i (h_j)^i \quad , \quad y_T = y - \sum_{i=1}^{m+2} Y_i (h_j)^i \quad ,$$

$$\dot{x}_T = \dot{x} - \sum_{i=1}^{m+2} iX_i (h_j)^{i-1} \quad , \quad \dot{y}_T = \dot{y} - \sum_{i=1}^{m+2} iY_i (h_j)^{i-1} \quad , \qquad (2\text{-}7)$$

the original differential equations (2-1) are transformed accordingly:

$$\ddot{x}_T = f_T(t, x_T, y_T, \dot{y}_T) = f(x, y, \dot{y}) - \sum_{i=2}^{m+2} i(i-1)\, X_i\, (h_j)^{i-2} \quad ,$$

$$\ddot{y}_T = g_T(t, x_T, y_T, \dot{x}_T) = g(x, y, x) - \sum_{i=2}^{m+2} i(i-1)\, Y_i\, (h_j)^{i-2} \quad . \qquad (2\text{-}8)$$

The initial values of the transformed variables for starting an orbit are

$$(x_T)_0 \stackrel{\text{def}}{=} x_0 \quad , \quad (y_T)_0 = y_0 \quad ,$$

$$(\dot{x}_T)_0 = 0 \quad , \quad (\dot{y}_T)_0 = 0 \quad . \qquad (2\text{-}9)$$

For equations (2-8) the following Runge-Kutta formulas develop (m+4)-th-order accuracy for $x, y, \dot{x}$, and $\dot{y}$ with only three substitutions per time step, $\Delta t_j = (t_{j+1} - t_j)$:

$$k_1 = \Delta t_j f_T(t_j + \alpha_1 \Delta t_j \, , \, (x_T)_j \, , \, (y_T)_j \, , \, 0) \quad ,$$

$$\ell_1 = \Delta t_j g_T(t_j + \alpha_1 \Delta t_j \, , \, (x_T)_j \, , \, (y_T)_j \, , \, 0) \quad ,$$

$$k_2 = \Delta t_j f_T(t_j + \alpha_2 \Delta t_j \, , \, (x_T)_j + \beta_0 k_1 \Delta t_j \, , \, (y_T)_j + \beta_0 \ell_1 \Delta t_j \, , \, \beta_1 \ell_1) \quad ,$$

$$\ell_2 = \Delta t_j g_T(t_j + \alpha_2 \Delta t_j \, , \, (x_T)_j + \beta_0 k_1 \Delta t_j \, , \, (y_T)_j + \beta_0 \ell_1 \Delta t_j \, , \, \beta_1 k_1) \quad ,$$

$$k_3 = \Delta t_j f_T(t_j + \alpha_3 \Delta t_j \, , \, (x_T)_j + \gamma_0 k_1 \Delta t_j + \delta_0 k_2 \Delta t_j \, , \, (y_T)_j$$

$$+ \gamma_0 \ell_1 \Delta t_j + \delta_0 \ell_2 \Delta t_j \, , \, \gamma_1 \ell_1 + \delta_1 \ell_2) \quad ,$$

$$\ell_3 = \Delta t_j g_T(t_j+\alpha_3\Delta t_j\ ,\ (x_T)_j + \gamma_0 k_1 \Delta t_j + \delta_0 k_2 \Delta t_j\ ,\ (y_T)_j$$
$$+ \gamma_0 \ell_1 \Delta t_j + \delta_0 \ell_2 \Delta t_j\ ,\ \gamma_1 k_1 + \delta_2 k_2)\ , \tag{2-10}$$

and

$$(x_T)_{j+1} = (x_T)_j + (C_1 k_1 + C_2 k_2 + C_3 k_3)\ \Delta t_j\ ,$$
$$(y_T)_{j+1} = (y_T)_j + (C_1 \ell_1 + C_2 \ell_2 + C_3 \ell_3)\ \Delta t_j\ ,$$
$$(\dot{x}_T)_{j+1} = C_1' k_1 + C_2' k_2 + C_3' k_3\ , \tag{2-11}$$
$$(\dot{y}_T)_{j+1} = C_1' \ell_1 + C_2' \ell_2 + C_3' \ell_3\ ,$$

where

$$\alpha_1 = 1\ ,\ \alpha_2 = \frac{m+2}{m+4}\ ,\ \alpha_3 = 1\ ,$$
$$\beta_0 = \frac{2}{(m+4)^2} \cdot \left(\frac{m+2}{m+4}\right)^{m+1}\ ,\ \beta_1 = \frac{1}{m+4} \cdot \left(\frac{m+2}{m+4}\right)^{m+1}\ ,$$
$$\gamma_0 = 0\ ,\ \gamma_1 = -\frac{1}{m+2}\ ,\ \delta_0 = 0\ ,\ \delta_1 = \frac{2}{m+2} \cdot \left(\frac{m+4}{m+2}\right)^{m+1}\ ,$$
$$C_1 = 0\ ,\ C_2 = \frac{1}{(m+2)(m+3)} \cdot \left(\frac{m+4}{m+2}\right)^{m+1}\ ,\ C_3 = 0,$$
$$C_1' = 0\ ,\ C_2' = \frac{1}{2} \cdot \frac{m+4}{(m+2)(m+3)} \cdot \left(\frac{m+4}{m+2}\right)^{m+1}\ ,\ C_3' = \frac{1}{2} \cdot \frac{1}{m+3}\ . \tag{2-12}$$

The following fourth substitution yields improved values, $\hat{x}_T$ and $\hat{y}_T$, which are more accurate by one power of Δt_j :

$$k_4 = \Delta t_j f_T(t_j+\alpha_4\Delta t_j\ ,\ (x_T)_j + \epsilon_0 k_1\Delta t_j + \zeta_0 k_2\Delta t_j + \eta_0 k_3\Delta t_j\ ,$$

$$(y_T)_j + \epsilon_0 \ell_1\Delta t_j + \zeta_0 \ell_2\Delta t_j + \eta_0 \ell_3\Delta t_j\ ,\ \epsilon_1\ell_1 + \zeta_1\ell_2 + \eta_1\ell_3)\ ,$$

$$\ell_4 = \Delta t_j g_T(t_j+\alpha_4\Delta t_j\ ,\ (x_T)_j + \epsilon_0 k_1\Delta t_j + \zeta_0 k_2\Delta t_j + \eta_0 k_3\Delta t_j\ , \tag{2-13}$$

$$(y_T)_j + \epsilon_0 \ell_1\Delta t_j + \zeta_0 \ell_2\Delta t_j + \eta_0 \ell_3\Delta t_j\ ,\ \epsilon_1 k_1 + \zeta_1 k_2 + \eta_1 k_3)\ ,$$

and

$$(\hat{x}_T)_{j+1} = (x_T)_j + (\hat{C}_1 k_1 + \hat{C}_2 k_2 + \hat{C}_3 k_3 + \hat{C}_4 k_4)\ \Delta t_j\ ,$$

$$(\hat{y}_T)_{j+1} = (y_T)_j + (\hat{C}_1 \ell_1 + \hat{C}_2 \ell_2 + \hat{C}_3 \ell_3 + \hat{C}_4 \ell_4)\ \Delta t_j\ , \tag{2-14}$$

where

$$\alpha_4 = \frac{m+2}{m+5}\ ,$$

$$\epsilon_0 = 0\ ,\ \epsilon_1 = \frac{1}{2}\cdot\left(\frac{1}{m+5}\right)\cdot\left(\frac{m+2}{m+5}\right)^{m+1}\ ,$$

$$\zeta_0 = 0\ ,\ \zeta_1 = \frac{1}{4}\cdot\frac{5m+16}{(m+5)^2}\cdot\left(\frac{m+4}{m+5}\right)^{m+1}\ ,$$

$$\eta_0 = \frac{3}{2}\cdot\frac{1}{(m+5)^2}\cdot\left(\frac{m+2}{m+5}\right)^{m+1}\ ,$$

$$\eta_1 = -\frac{3}{4}\cdot\frac{m+2}{(m+5)^2}\cdot\left(\frac{m+2}{m+5}\right)^{m+1}\ ,$$

$$\hat{C}_1 = 0\ ,\ \hat{C}_2 = 0\ ,\ \hat{C}_3 = \frac{1}{3}\cdot\frac{1}{(m+3)(m+4)}\ ,$$

$$\hat{C}_4 = \frac{2}{3}\cdot\frac{m+5}{(m+2)(m+3)(m+4)}\cdot\left(\frac{m+5}{m+2}\right)^{m+1}\ . \tag{2-15}$$

The approximate value of the truncation error in both variables is given by

$$(T_x)_{j+1} \approx \left|(x_T)_{j+1}-(\hat{x}_T)_{j+1}\right| \quad ,$$
$$(T_y)_{j+1} \approx \left|(y_T)_{j+1}-(\hat{y}_T)_{j+1}\right| \quad . \tag{2-16}$$

Since the evaluation of f_T and g_T, necessary in equations (2-10) and (2-13) can be accomplished only through the relationships given in (2-8), the following inverse transformations must be executed:

$$x_{j+1} = (x_T)_{j+1} + \sum_{i=1}^{m+2} X_i \, (\Delta t_j)^i \quad ,$$
$$y_{j+1} = (y_T)_{j+1} + \sum_{i=1}^{m+2} Y_i \, (\Delta t_j)^i \quad ,$$
$$\dot{x}_{j+1} = (\dot{x}_T)_{j+1} + \sum_{i=1}^{m+2} i \, X_i \, (\Delta t_j)^{i-1} \quad , \tag{2-17}$$
$$\dot{y}_{j+1} = (\dot{y}_T)_{j+1} + \sum_{i=1}^{m+2} i \, Y_i \, (\Delta t_j)^{i-1} \quad ;$$

thus allowing values of $(f)_{j+1}$ and $(g)_{j+1}$ to be obtained through equations (2-1). Equations (2-8) now yield values of $(f_T)_{j+1}$ and $(g_T)_{j+1}$.

3. The Runge-Kutta-Fehlberg Computer Program

A Burroughs B-5000 ALGOL program was written in double precision to implement the integration procedure described in the preceding paragraphs. A concerted effort was made to duplicate the numerical method as closely as possible. A flow chart of the program is presented in Section 4 and a listing of the program in Section 5.

There are four options under which the program may be run. The first option produces values of x, y, $\dot{x}$, $\dot{y}$, t, the step size, the Jacobi constant, the distance to the earth, and the distance to the Moon at the beginning and at the end of a full period. A period is determined to be complete when $|y| < \epsilon_1$ after a given time equal to 99% of the approximate time for one period. The value of ϵ_1 and the approximate time for one period are input parameters (EPS1 and PER respectively).

The second option produces the same output as the first. However, under this option it is produced at each time step around the orbit.

The third option allows the user to iterate on a given set of initial conditions until an orbit is produced which is periodic within a prescribed error tolerance or until the initial conditions no longer improve. The technique employed consists of interpolating on successive half-orbit values of $\dot{x}_{\frac{1}{2}}$ and $\dot{y}$ initial, to produce an improved initial value of $\dot{y}$. The process terminates when at half-period $|\dot{x}| < \epsilon_2$ or when $|\dot{x}|$ has failed to improve on three successive half-orbits. The value of ϵ_2 is an input parameter (EPS2) and the half-period point is determined in a manner similar to that described for determining the full period point.

The fourth option is the same as the third except that it provides for a printout of the values described in option one at each time step during each half orbit.

It should be noted that at the end of an orbit under all options the total number of steps taken for that orbit is printed out.

Automatic step size control is an inherent part of the program. Values for a particular time step are accepted if, and only if, neither of the following conditions exist:

$$\frac{T_x}{\epsilon|x|} > 1 \, , \quad \frac{T_y}{\epsilon|y|} > 1$$

where T_x and T_y are measures of the truncation error as described in the paragraphs on the numerical method and ϵ is an input parameter (EPS). If either condition exists, the step size is halved and the values are recomputed.

Doubling of the step size for the next step will occur if, and only if, both the following conditions are satisfied for the current time step:

$$\frac{T_x}{\epsilon|x|} < \left(\frac{1}{2}\right)^{m+5} \, , \quad \frac{T_y}{\epsilon|y|} < \left(\frac{1}{2}\right)^{m+5} \, ,$$

where m has the same meaning as described earlier.

There are no unusual hardware requirements. The program may be executed on a minimum B-5000 system having only one card reader and one line printer.

3.1. Data Input

Input data for the program is supplied to the program in sets. There is no limit to the number of data sets which may be queued in the card reader for processing at one time.

A data set consists of 23 parameters. These parameters are of single precision, double precision, and logical value. The read mode which was selected is Free-Field Input.

All Free-Field Input is in the form of free-field data sentences. Each field in a sentence is associated with the list element to which it corresponds according to position.

A number which is represented as an INTEGER will be converted as an INTEGER unless it is larger than the largest allowable integer, in which case

it will be converted as REAL. Numbers which contain a decimal fraction will be converted as REAL.

For the purpose of Free-Field Input, an INTEGER 1 (one) must be used in lieu of the logical value TRUE, and an INTEGER 0 (zero) must be used in lieu of the logical value FALSE.

The delimiters for the single precision and the double precision free-field read are different. For the single precision read, a comma is used for a delimiter. However, for the double precision read, a space (blank column) is used as a delimiter. Also, it should be noted that a double precision number and a single precision number must not be on the same card.

The data sets must always be arranged in the following order. The first 15 of the parameters will be read in under single precision free field and should be formatted accordingly.

r_{max}	the maximum allowable distance between the orbiting body and the Earth.
r_{min}	the minimum allowable distance between the orbiting body and the Earth.
s_{max}	the maximum allowable distance between the orbiting body and the Moon.
s_{min}	the minimum allowable distance between the orbiting body and the Moon.
t_{max}	the maximum allowable time for one orbit.
m	the number of terms of the power series expansion to be considered.

ϵ — the truncation error tolerance for halving the interval size in the test,

$$\frac{T_x}{\epsilon|x|} > 1 \quad .$$

ϵ_1 — the error tolerance involved in the test, $|y| < \epsilon_1$, during the iterative procedure for producing periodic orbits.

ϵ_2 — the error tolerance involved in the test $|\dot{x}| < \epsilon_2$, during the iterative procedure for producing periodic orbits.

PER — the approximate period of the orbit.

d_1 — the direction of crossing at half period. d_1 equal a +1 if crossing is from plus to minus, and -1 otherwise.

PLT — the input file closing variable; should be an INTEGER 0 (zero) for all data sets except the last one in which case it must be the integer 1 (one). Should be the INTEGER 1 if only one data set is to be read in.

PRITR — an output option. Should be the INTEGER 1 if it is desired to print each set of improved initial conditions when iterating to produce a periodic orbit; otherwise, should be the INTEGER 0.

ITER — an input option. Should be the INTEGER 1 if it is desired to iterate on the initial conditions; otherwise, INTEGER 0.

PRINTSTEP — an input option. Should be the INTEGER 1 if it is desired to print each time step for a complete orbit; otherwise, INTEGER 0.

This completes the list of single-precision input parameters. The remaining eight input parameters must have the double precision input format described previously.

x_0 — the initial value for x.

y_0 — the initial value for y.

$\dot{x}_0$ — the initial value for $\dot{x}$.

$\dot{y}_0$ — the initial value for $\dot{y}$.

$\dot{x}_A$ — the previous value for $\dot{x}_{\frac{1}{2}}$ to be used in interpolating at half orbit.

$\dot{y}_A$ — the previous value for $\dot{y}$ initial to be used in interpolating at half orbit.

t_0 — the starting time for an orbit.

h_0 — the initial step-size interval.

This completes the list of input parameters.

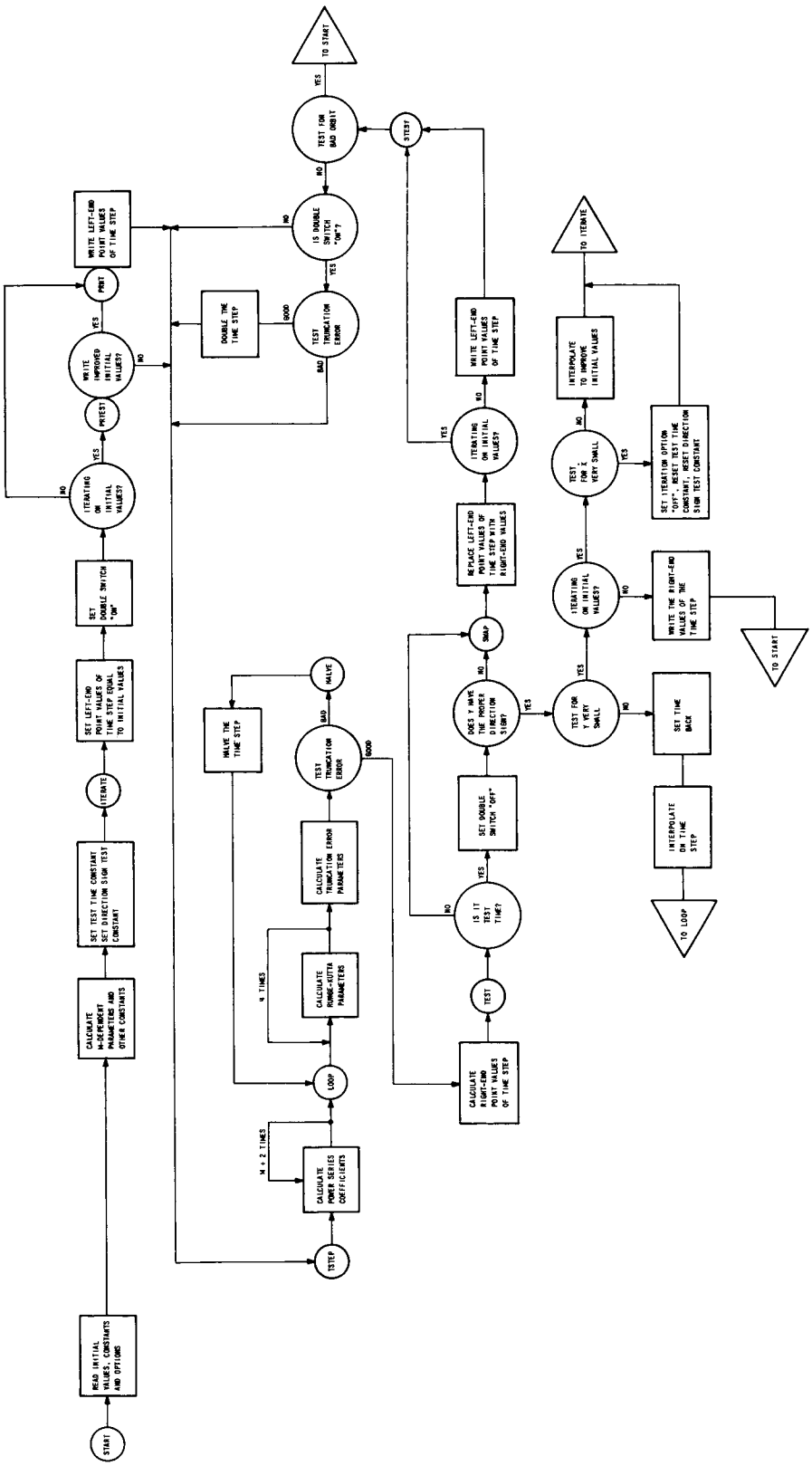

Figure 9. The Flow Diagram for the Runge-Kutta-Fehlberg Method.

4. Flow Diagram for the Runge-Kutta-Fehlberg Method

Figure 9 gives a flow diagram for the Runge-Kutta-Fehlberg computer program.

5. The Program Listing for the Runge-Kutta-Fehlberg Method

The following 24 pages comprise the program listing for the Runge-Kutta-Fehlberg Method.

```
 LABEL   000000000XXXXXX001     01
BEGIN                                                                    00001000
  COMMENT FOLLOWING ARE THE DECLARATIONS FOR THE PROGRAM;                00002000
  REAL KOUNT;                                                            00003000
  INTEGER COUNT;                                                         00004000
  REAL TSTARTH,TSTARTL,HSTARTH,HSTARTL;                                  00005000
  BOOLEAN PRNTSTEP ;                                                     00006000
  BOOLEAN DUB1;                                                          00007000
  REAL XSTARTH,XSTARTL,YSTARTH,YSTARTL,DXSTARTH,DXSTARTL;                00008000
  REAL YOSQH,YOSQL;                                                      00009000
  INTEGER LAMDA,LM1,I,MP2,K,IM1,IM2,IIM1;                                00010000
  REAL ARRAY KH,KL[0:4], LH,LL[0:4], XNUH,XNUL[0:99];                    00011000
  REAL FVALXTH,FVALXTL,FVALYTH,FVALYTL,XTH,XTL,XOH,XOL,HH,HL,YTH, YTL,   00012000
  YOH,YOL,DFVALXTH,DFVALXTL,DFVALYTH,DFVALYTL,DXTH,DXTL, DYTH,DYTL,DXOH  00013000
  ,DXOL,DYOH,DYOL,DTH,DTL,SUM3XH,SUM3XL, SUM3YH,SUM3YL,DTIH,DTIL,XH,XL,  00014000
  SUM4XH,SUM4XL,SUM4YH,SUM4YL, IDTIM1H,IDTIM1L,DXH,DXL,DYH,DYL,YSQH,YSQL 00015000
  ,YH,YL;                                                                00016000
  REAL DEN1H,DEN1L,MUH,MUL,DON1H,DON1L,DON2H,DON2L,DEN2H,DEN2L, MUPMH,    00017000
  MUPML,FH,FL,GH,GL,SUM5FTH,SUM5FTL,SUM5GTH,SUM5GTL, IIM1DTIM2H,         00018000
  IIM1DTIM2L,DTSQH,DTSQL, FTH,FTL,GTH,GTL;                               00019000
  LABEL LAST;                                                            00020000
  REAL ARRAY ALFH,ALFL[0:4],FCONH,FCONL[0:3,0:3],DFCONH,DFCONL[0:3,0:3]  00021000
  ,CH,CL,CPMH,CPML[0:4],CHATH,CHATL[0:4];                                00022000
  REAL EPSM,M1H,M1L,M2H,M2L,M3H,M3L,M4H,M4L,M5H,M5L,M6H,M6L,M7H,M7L,M8H   00023000
  ,M8L,M9H,M9L,M10H,M10L,M11H,M11L,M12H,M12L;                            00024000
  INTEGER MP3,MP4,MP5, J;                                                00025000
  REAL RNH,RNL, SNH,SNL,TERMH,TERML,TERM1H,TERM1L,TERM2H,TERM2L,TERM3H,   00026000
  TERM3L,TERM4H,TERM4L,UNUNH,UNUNL,VNUNH,VNUNL,XNUNP1H,XNUNP1L, YNUNP1H  00027000
  ,YNUNP1L,C1H,C1L,C2H,C2L;                                              00028000
  REAL ARRAY YNUH,YNUL[0:99],RNUH,RNUL[0:99],SNUH, SNUL[0:99],UNUH,UNUL  00029000
  [0:99],VNUH,VNUL[0:99],SIGH,SIGL[0:99], OT[0:4];                       00030000
  INTEGER M,N, MP1,NM1,D1,D2, NMI,NM1MI,NP1;                             00031000
  FILE OUT FLO 1(2,15);                                                  00032000
  FORMAT FMTO(5A6);                                                      00033000
  LIST LSTO(FOR K←0 STEP 1 UNTIL 4 DO OT[K]);                            00034000
  FILE IN FLI(1,10);                                                     00035000
  FORMAT IN FMTI(F6.2,I2);                                               00036000
  LIST LSTI(MUH,M);                                                      00037000
  FILE IN PCARD(1,10);                                                   00038000
  FORMAT IN PFMT (I2/5E15.8/E15.8/4L5);                                  00039000
```

```
  LABEL START,ITERATE;                                                          00040000
  REAL TSTART,XSTART,YSTART,DXSTART,DYSTART,HSTART,EPS,RMIN,RMAX,SMIN,          00041000
  SMAX,TMAX, RH,RL,SH,SL,JH,JL,DJH,DJL,SUM6TXH,SUM6TXL;                        00042000
  REAL ARRAY CDIFH,CDIFL [0:4];                                                 00043000
  FORMAT OUT LFMT(X3,"T=",E17.10,"X=",E17.10,"Y=",E17.10,"XD=",E17.10,"Y00044000
D=",E17.10,"DT=",E17.10,"J=",E17.10"DJ=",E17.10, "RE=",E17.10,"RM=",          00045000
  E17.10///);                                                                   00046000
  FILE OUT PRINT 1(3,15) ;                                                      00047000
  REAL SUM6TYH,SUM6TYL,TXH,TXL,TYH,TYL,EPSXH,EPSXL,EPSYH,EPSYL,ERRXH,           00048000
  ERRXL,ERRYH,ERRYL,SUM6XT1H,SUM6XT1L,SUM6YT1H,SUM6YT1L,XT1L,XT1H;              00049000
  REAL YT1H,YT1L,SUM6DXT1H,SUM6DXT1L,SUM6DYT1H,SUM6DYT1L,DXT1H,DXT1L,           00050000
  DYT1H,DYT1L, SUM3X1H,SUM3X1L,SUM3Y1H,SUM3Y1L,HTIH,HTIL,X1H,X1L, Y1H,          00051000
  Y1L,SUM4DX1H,SUM4DX1L,SUM4DY1H,SUM4DY1L,IHTIH,IHTIL,DX1H, DX1L,DY1H,          00052000
  DY1L,TH,TL,Y1SQH,Y1SQL,J1H,J1L;                                               00053000
  FORMAT OUT LFMT1(X6,"T=",5A6,X6, "X=",5A6,X6,"Y=",5A6/X5,"XD=",5A6,X5,00054000
  "YD=",5A6,X5,"DT=",5A6/X6,"J=",5A6,X5,"DJ=",5A6,X5, "RE=",5A6,/X5,"RM=00055000
",5A6///);                                                                      00056000
  ALPHA ARRAY OT1,OT2,OT3,OT4,OT5,OT6,OT7,OT8,OT9,OT10[0:4];                    00057000
  LIST PLINE1(FOR I ← 0 STEP 1 UNTIL 4 DO OT1[I], FOR I ← 0 STEP 1 UNTIL00058000
  4 DO OT2[I], FOR I ← 0 STEP 1 UNTIL 4 DO OT3[I], FOR I ← 0 STEP 1             00059000
  UNTIL 4 DO OT4[I], FOR I ← 0 STEP 1 UNTIL 4 DO OT5[I], FOR I ← 0 STEP 00060000
  1 UNTIL 4 DO OT6[I], FOR I ← 0 STEP 1 UNTIL 4 DO OT7[I], FOR I ← 0            00061000
  STEP 1 UNTIL 4 DO OT8[I], FOR I ← 0 STEP 1 UNTIL 4 DO OT9[I], FOR I ← 00062000
  0 STEP 1 UNTIL 4 DO OT10[I]);                                                 00063000
  LIST PLINE2(TH,XOH,YOH,DXOH,DYOH,HH,JH,DJH,RH,SH);                           00064000
  LABEL PRTEST,PRNT;                                                            00065000
  BOOLEAN ITER,PLT,DUB,PRITR;                                                   00066000
  INTEGER SAME,DIR1,DIR2,NP,CC;                                                 00067000
  REAL ARRAY PLOTX,PLOTY[0:400];                                                00068000
  REAL PER,DXAH,DXAL,DYAH,DYAL,TTEST,EPS1,EPS2,DYHOLDH,DYHOLDL,DYSTARTH,00069000
  DYSTARTL;                                                                     00070000
  INTEGER T;                                                                    00071000
  LIST PLIST(RMAX,RMIN,SMAX,SMIN,TMAX,M,EPS,EPS1,EPS2,PER,DIR1, PLT,           00072000
  PRITR,ITER,PRNTSTEP);                                                         00073000
  FORMAT OUT FMTT(" TIME ",I6);                                                 00074000
  REAL HLDDXAH,HLDDXAL,HLDDYAH,HLDDYAL ;                                        00075000
  REAL TOH,TOL;                                                                 00076000
  BOOLEAN NEW;                                                                  00077000
  INTEGER L;                                                                    00078000
  REAL ARRAY CON[0:7];                                                          00079000
```

```
REAL X,Y,X1,X2,H1Y,L1Y;                                                  00080000
COMMENT FOLLOWING IS A DOUBLE PRECISION SQUAREROOT ROUTINE;              00081000
DEFINE DDSQRT =                                                          00082000
BEGIN                                                                    00083000
  IF X1≠0 THEN                                                           00084000
  BEGIN                                                                  00085000
    CON[0]←0.00000269739888;                                             00086000
    CON[1]←0.000001603883;                                               00087000
    CON[2]←0.0000076294;                                                 00088000
    CON[3]←0.000004536465;                                               00089000
    CON[4]←2.137099@23;                                                  00090000
    CON[5]←1.270727@23;                                                  00091000
    CON[6]←7.555786@22;                                                  00092000
    CON[7]←4.482697@22;                                                  00093000
    X←X1×1.0;                                                            00094000
    L←0;                                                                 00095000
    L←X.[8:2];                                                           00096000
    L.[45:1]←X.[2:1];                                                    00097000
    Y←X;                                                                 00098000
    Y.[3:6]←Y.[2:6];                                                     00099000
    Y←Y×CON[L];                                                          00100000
    Y←(X/Y+Y)×0.5;                                                       00101000
    Y←(X/Y+Y)×0.5;                                                       00102000
    Y←(X/Y+Y)×0.5;                                                       00103000
    Y←(X/Y+Y)×0.5;                                                       00104000
    DOUBLE(X1,X2,Y,0,/,Y,0,-,,0.5,×,Y,0,+,←,H1Y,L1Y);                    00105000
                                                                         00106000
  END ELSE H1Y←L1Y←0;                                                    00107000
                                                                         00108000
END#;                                                                    00109000
FILE IN CARD(1,10);                                                      00110000
FORMAT OUT KOFMT(X6,"NUMBER OF STEPS = ",I6);                            00111000
INTEGER EVER,NEVER ;                                                     00112000
INTEGER ARRAY NOTUSE1[0:80],NOTUSE2[0:9] ;                               00113000
REAL ARRAY NOTUSE3[0:68],NOTUSE4[0:68] ;                                 00114000
COMMENT FOLLOWING IS A DOUBLE PRECISION INPUT PROCEDURE;                 00115000
PROCEDURE DREAD(AH,AL) ;                                                 00116000
REAL AH,AL ;                                                             00117000
                                                                         00118000
BEGIN                                                                    00119000
```

```
INTEGER K,T1 ;                                                          00120000
REAL TH,TL ;                                                            00121000
BOOLEAN DF,MF ;                                                         00122000
LABEL ER,L1,L2,L3,L4,L5,L6,L7 ;                                         00123000
FORMAT OUT FR(/"DATA CARD ERROR",X105) ;                                00124000
STREAM PROCEDURE EC(S,D) ;                                              00125000
                                                                        00126000
BEGIN                                                                   00127000
  SI := S ;                                                             00128000
  DI := D ;                                                             00129000
  DS := 8 LIT "0" ;                                                     00130000
  2(40(DS := 7 LIT "0" ;                                                00131000
  DS := 1 CHR)) ;                                                       00132000
                                                                        00133000
END ;                                                                   00134000
PROCEDURE GN ;                                                          00135000
                                                                        00136000
BEGIN                                                                   00137000
  LABEL L1 ;                                                            00138000
  IF NEVER ≤ 80 THEN GO TO L1 ELSE IF NEVER > 81 THEN                   00139000
  BEGIN                                                                 00140000
    READ (CARD,10,NOTUSE2[*])[LAST] ;                                   00141000
    EC(NOTUSE2,NOTUSE1) ;                                               00142000
    NEVER := 1 ;                                                        00143000
    L1:EVER := NOTUSE1[NEVER] ;                                         00144000
    IF EVER = "*" THEN                                                  00145000
    BEGIN                                                               00146000
      NEVER := 81 ;                                                     00147000
      EVER := " "                                                       00148000
    END ELSE                                                            00149000
  END ELSE EVER := " " ;                                                00150000
  NEVER := NEVER + 1 ;                                                  00151000
                                                                        00152000
END ;                                                                   00153000
L1:GN ;                                                                 00154000
IF EVER = " " THEN GO TO L1 ;                                           00155000
DF := FALSE ;                                                           00156000
K := 0 ;                                                                00157000
TH := 0 ;                                                               00158000
TL := 0 ;                                                               00159000
```

```
IF EVER = "-" THEN MF := TRUE ELSE                                    00160000
BEGIN                                                                 00161000
  MF := FALSE ;                                                       00162000
  IF EVER ≠ "+" THEN GO TO L3 ELSE                                    00163000
END ;                                                                 00164000
L2:GN ;                                                               00165000
L3:IF EVER < 10 THEN                                                  00166000
BEGIN                                                                 00167000
  DF := TRUE ;                                                        00168000
  DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL) ;                             00169000
  GO TO L2                                                            00170000
END ;                                                                 00171000
IF EVER = "." THEN                                                    00172000
BEGIN                                                                 00173000
  L4:GN ;                                                             00174000
  IF EVER < 10 THEN                                                   00175000
  BEGIN                                                               00176000
    K := K - 1 ;                                                      00177000
    DF := TRUE ;                                                      00178000
    DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL) ;                          00179000
    GO TO L4                                                          00180000
  END ELSE                                                            00181000
END ;                                                                 00182000
IF MF THEN DOUBLE(,0,TH,TL,-,:=,TH,TL) ;                              00183000
IF NOT DF THEN                                                        00184000
BEGIN                                                                 00185000
  ER:WRITE(PRINT,FR) ;                                                00186000
  GO TO LAST                                                          00187000
END ;                                                                 00188000
IF EVER = " " THEN GO TO L7 ;                                         00189000
IF EVER ≠ "@" THEN GO TO ER ;                                         00190000
GN ;                                                                  00191000
T1 := 0 ;                                                             00192000
DF := FALSE ;                                                         00193000
IF EVER = "-" THEN MF := TRUE ELSE                                    00194000
BEGIN                                                                 00195000
  MF := FALSE ;                                                       00196000
  IF EVER ≠ "+" THEN GO TO L6 ELSE                                    00197000
END ;                                                                 00198000
L5:GN ;                                                               00199000
```

```
    L6:IF EVER < 10 THEN                                                    00200000
    BEGIN                                                                   00201000
      DF := TRUE ;                                                          00202000
      T1 := T1 × 10 + EVER ;                                                00203000
      GO TO L5                                                              00204000
    END ;                                                                   00205000
    IF NOT DF THEN GO TO ER ;                                               00206000
    IF EVER ≠ " " THEN GO TO ER ;                                           00207000
    IF MF THEN K := K - T1 ELSE K := K + T1 ;                               00208000
    L7:IF K < 0 THEN DOUBLE(TH,TL,NOTUSE3[-K],NOTUSE4[-K],/,:=,AH,AL)      00209000
    ELSE DOUBLE(TH,TL,NOTUSE3[K],NOTUSE4[K],×,:=,AH,AL) ;                   00210000
                                                                            00211000
  END ;                                                                     00212000
  COMMENT FOLLOWING IS A DOUBLE PRECISION OUTPUT PROCEDURE;                 00213000
  PROCEDURE DPOP(A1, A2, F, ALF, A) ;                                       00214000
  VALUE A1, A2 ;                                                            00215000
  REAL A1, A2 ;                                                             00216000
  FILE F ;                                                                  00217000
  ALPHA ARRAY A, ALF[0] ;                                                   00218000
                                                                            00219000
  BEGIN                                                                     00220000
    STREAM PROCEDURE PRINT(X, Y, N, F, A, ALF) ;                            00221000
    VALUE X, Y, N ;                                                         00222000
                                                                            00223000
    BEGIN                                                                   00224000
      LOCAL V1, V11, V2, V22, V3, V4, ST, RP ;                              00225000
      DI ← LOC V22;                                                         00226000
      SI ← LOC X ;                                                          00227000
      SI ← SI+1 ;                                                           00228000
      SKIP 3 SB ;                                                           00229000
      13(DS ← 3 RESET ;                                                     00230000
      3(IF SB THEN DS ← SET ELSE DS ← RESET ;                               00231000
      SKIP 1 SB)) ;                                                         00232000
      SI ← LOC Y ;                                                          00233000
      SI ← SI+1 ;                                                           00234000
      SKIP 3 SB ;                                                           00235000
      13(DS ← 3 RESET ;                                                     00236000
      3(IF SB THEN DS ← SET ELSE DS ← RESET ;                               00237000
      SKIP 1 SB)) ;                                                         00238000
      DI ← LOC RP ;                                                         00239000
```

```
DS ← 8 LIT"00000000" ;                                                    00240000
DI ← LOC ST ;                                                             00241000
SI ← F ;                                                                  00242000
DS ← WDS ;                                                                00243000
DI ← ST ;                                                                 00244000
SI ← LOC X ;                                                              00245000
SKIP 1 SB ;                                                               00246000
IF SB THEN DS ← 8 LIT"     -0." ELSE DS ← 8 LIT"      0." ;               00247000
3(DS ← 8 LIT"00000000") ;                                                 00248000
11(DS ← 8 LIT"        ") ;                                                00249000
SI ← A ;                                                                  00250000
V4 ← SI ;                                                                 00251000
SI ← LOC V22;                                                             00252000
V3 ← SI ;                                                                 00253000
26(DI ← LOC RP ;                                                          00254000
DI ← DI + 7 ;                                                             00255000
DS ← CHR ;                                                                00256000
SI ← V4 ;                                                                 00257000
DI ← ST ;                                                                 00258000
DI ← DI+8 ;                                                               00259000
RP(DS ← 24 ADD ;                                                          00260000
DI ← DI - 24 ;                                                            00261000
SI ← SI - 24);                                                            00262000
SI ← SI + 24 ;                                                            00263000
V4 ← SI ;                                                                 00264000
SI ← V3 ;                                                                 00265000
SI ← SI+1 ;                                                               00266000
V3 ← SI) ;                                                                00267000
DI ← DI + 24 ;                                                            00268000
SI ← LOC N ;                                                              00269000
SKIP 1 SB ;                                                               00270000
IF SB THEN DS ← 2 LIT "@-" ELSE DS ← 2 LIT "@+" ;                         00271000
SI ← LOC N ;                                                              00272000
DS ← 2 DEC ;                                                              00273000
DI ← DI-1 ;                                                               00274000
2(DS ← RESET) ;                                                           00275000
DI ← ALF ;                                                                00276000
SI ← ST ;                                                                 00277000
SI ← SI + 5 ;                                                             00278000
DI ← DI + 2 ;                                                             00279000
```

```
    DS ← CHR ;                                                    00280000
    SI ← SI + 2 ;                                                 00281000
    DS ← CHR ;                                                    00282000
    DS ← 1 LIT"." ;                                               00283000
    DS ← 3 CHR ;                                                  00284000
    4(DI ← DI + 2 ;                                               00285000
    DS ← 6 CHR) ;                                                 00286000
                                                                  00287000
  END ;                                                           00288000
  STREAM PROCEDURE SHIFT(A, B, C) ;                               00289000
  VALUE A, B ;                                                    00290000
                                                                  00291000
  BEGIN                                                           00292000
    SI ← LOC A ;                                                  00293000
    DI ← C ;                                                      00294000
    DS ← CHR ;                                                    00295000
    SKIP 3 SB ;                                                   00296000
    DI ← DI + 1 ;                                                 00297000
    36(IF SB THEN DS ← SET ELSE DS ← RESET ;                      00298000
    SKIP 1 SB) ;                                                  00299000
    SKIP 9 DB ;                                                   00300000
    3(IF SB THEN DS ← SET ELSE DS ← RESET ;                       00301000
    SKIP 1 SB) ;                                                  00302000
    SI ← LOC B ;                                                  00303000
    SKIP 9 SB ;                                                   00304000
    36(IF SB THEN DS ← SET ELSE DS ← RESET ;                      00305000
    SKIP 1 SB) ;                                                  00306000
                                                                  00307000
  END ;                                                           00308000
  INTEGER Y, I ;                                                  00309000
  REAL T1, T2, HT ,LT;                                            00310000
  ARRAY T[0:1] ;                                                  00311000
  LABEL L1, L2, L3 ;                                              00312000
  IF A1 = 0 THEN                                                  00313000
  BEGIN                                                           00314000
    T1 ← T2 ← 0 ;                                                 00315000
    GO TO L2                                                      00316000
  END ;                                                           00317000
  HT ← 1 ;                                                        00318000
  LT ← 0 ;                                                        00319000
```

```
    Y ← 0.90309 × (IF BOOLEAN(A1.[2:1]) THEN 13 - A1.[3:6] ELSE 13 + A1.00320000
    [3:6]) ;                                                              00321000
    FOR I ← 1 STEP 1 UNTIL ABS(Y) DO DOUBLE(HT,LT,,10,×,←,HT,LT);         00322000
    IF Y>0 THEN DOUBLE(A1, A2, HT,LT, /, ←, T1, T2) ELSE DOUBLE(A1, A2, 00323000
    HT,LT, ×, ←, T1, T2) ;                                                00324000
    Y ← Y-1 ;                                                             00325000
    L1: IF T1.[3:6] < 13 THEN                                             00326000
    BEGIN                                                                 00327000
      DOUBLE(T1, T2, , 10.0, /, ←, T1, T2) ;                              00328000
      Y ← Y+1                                                             00329000
    END ELSE IF T1.[3:6] > 13 THEN                                        00330000
    BEGIN                                                                 00331000
      DOUBLE(T1, T2, , 10.0, ×, ←, T1, T2) ;                              00332000
      Y ← Y-1                                                             00333000
    END ELSE GO TO L2 ;                                                   00334000
    IF T1.[3:6] = 14 THEN                                                 00335000
    BEGIN                                                                 00336000
      SHIFT(T1, T2, T) ;                                                  00337000
      PRINT(T[0], T[1], Y, F, A, ALF) ;                                   00338000
      GO TO L3                                                            00339000
    END ELSE GO TO L1 ;                                                   00340000
    L2: PRINT(T1, T2, Y, F, A, ALF ) ;                                    00341000
    L3:                                                                   00342000
  END ;                                                                   00343000
  ALPHA ARRAY A[0:77] ;                                                   00344000
  FILL A[*] WITH "12500000", "00000000", "00000000", "01562500", "00000000345000
00", "00000000", "00195312", "50000000", "00000000", "00024414", "06250000346000
00", "00000000", "00003051", "75781250", "00000000", "00000381", "46972600347000
56", "25000000", "00000047", "68371582", "03125000", "00000005", "96046400348000
47", "75390625", "00000000", "74505805", "96923828", "00000000", "09313200349000
25", "74615479", "00000000", "01164153", "21826935", "00000000", "00145500350000
19", "15228367", "00000000", "00018189", "89403546", "00000000", "00002200351000
73", "73675443", "00000000", "00000284", "21709430", "00000000", "00000000352000
35", "52713679", "00000000", "00000004", "44089210", "00000000", "00000000353000
00", "55511151", "00000000", "00000000", "06938894", "00000000", "00000000354000
00", "00867362", "00000000", "00000000", "00108420", "00000000", "00000000355000
00", "00013553", "00000000", "00000000", "00001694", "00000000", "00000000356000
00", "00000212", "00000000", "00000000", "00000026", "00000000", "00000000357000
00", "00000003";                                                          00358000
                                                                          00359000
```

```
BEGIN                                                                    00360000
  INTEGER I ;                                                            00361000
  DOUBLE(,1,:=,NOTUSE3[0],NOTUSE4[0]) ;                                  00362000
  FOR I := 1 STEP 1 UNTIL 33 DO DOUBLE(,10,NOTUSE3[I - 1],NOTUSE4[I -    00363000
  1],×,:=,NOTUSE3[I], NOTUSE4[I]) ;                                      00364000
  FOR I := 1 STEP 1 UNTIL 35 DO DOUBLE(NOTUSE3[33],NOTUSE4[33],NOTUSE300365000
  [I],NOTUSE4[I],×,:=, NOTUSE3[I + 33],NOTUSE4[I + 33]) ;                00366000
  NEVER := 82 ;                                                          00367000
                                                                         00368000
END ;                                                                    00369000
COMMENT FOLLOWING IS THE PROGRAM BODY;                                   00370000
START: READ(CARD,/,PLIST)[LAST];                                         00371000
DREAD(XSTARTH,XSTARTL);                                                  00372000
DREAD(YSTARTH,YSTARTL);                                                  00373000
DREAD(DXSTARTH,DXSTARTL);                                                00374000
DREAD (DYSTARTH,DYSTARTL);                                               00375000
DREAD (DXAH,DXAL);                                                       00376000
DREAD (DYAH,DYAL);                                                       00377000
DREAD(TSTARTH,TSTARTL);                                                  00378000
DREAD(HSTARTH,HSTARTL);                                                  00379000
IF PLT THEN CLOSE (CARD,RELEASE);                                        00380000
DIR2 ← +1 ;                                                              00381000
COUNT ← 0;                                                               00382000
IF ITER THEN                                                             00383000
BEGIN                                                                    00384000
  TTEST ← 0.49 × PER;                                                    00385000
  SAME ← DIR1                                                            00386000
END ELSE                                                                 00387000
BEGIN                                                                    00388000
  TTEST ← 0.99 × PER;                                                    00389000
  SAME ← DIR2 ;                                                          00390000
                                                                         00391000
END ;                                                                    00392000
NP ← 1 ;                                                                 00393000
CC ← 1 ;                                                                 00394000
WRITE(PRINT[PAGE]);                                                      00395000
DOUBLE ( ,1, ,82.45,/,←,MUH,MUL) ;                                       00396000
COMMENT FOLLOWING IS THE CALCULATION OF THE M-DEPENDENT CONSTANTS;       00397000
MP1←M+1;                                                                 00398000
MP2←M+2;                                                                 00399000
```

```
MP3←M+3;                                                                    00400000
MP4←M+4;                                                                    00401000
MP5←M+5;                                                                    00402000
DOUBLE(1.0,MUH,MUL,-,MUPMH,MUPML);                                          00403000
DOUBLE( 2.0,MUH,MUL,×,C1H,C1L);                                             00404000
DOUBLE(2.0,MUPMH,MUPML,×,C2H,C2L);                                          00405000
EPSM←(0.5)*MP5;                                                             00406000
DOUBLE(MP2,0,MP4,0,/,M1H,M1L);                                              00407000
M2H←1.0;                                                                    00408000
M2L←0;                                                                      00409000
FOR I←1 STEP 1 UNTIL MP1 DO DOUBLE(M2H,M2L,M1H,M1L,×,M2H,M2L);              00410000
DOUBLE(MP4,0,MP2,0,/,M3H,M3L);                                              00411000
M4H←1.0;                                                                    00412000
M4L←0;                                                                      00413000
FOR I←1 STEP 1 UNTIL MP1 DO DOUBLE(M4H,M4L,M3H,M3L,×,M4H,M4L);              00414000
DOUBLE(MP2,0,MP5,0,/,M5H,M5L);                                              00415000
M6H←1.0;                                                                    00416000
M6L←0;                                                                      00417000
FOR I←1 STEP 1 UNTIL MP1 DO DOUBLE(M6H,M6L,M5H,M5L,×,M6H,M6L);              00418000
DOUBLE(MP5,0,MP5,0,×,M7H,M7L);                                              00419000
DOUBLE(MP2,0,MP3,0,×,M8H,M8L);                                              00420000
ALFH[1]←1;                                                                  00421000
ALFL[1]←0;                                                                  00422000
ALFH[2]←M1H;                                                                00423000
ALFL[2]←M1L;                                                                00424000
ALFH[3]←1;                                                                  00425000
ALFL[3]←0;                                                                  00426000
DOUBLE(M5H,M5L,←,ALFH[4],ALFL[4]);                                          00427000
DOUBLE(2.0,M2H,M2L,×,MP4,0,MP4,0,×,/,FCONH[1,1],FCONL[1,1]);                00428000
FCONH[2,1]←FCONL[2,1]←0;                                                    00429000
FCONH[2,2]←FCONL[2,2]←0;                                                    00430000
FCONH[3,1]←FCONL[3,1]←0;                                                    00431000
FCONH[3,2]←FCONL[3,2]←0;                                                    00432000
DOUBLE(1.5,M6H,M6L,×,M7H,M7L,/,FCONH[3,3],FCONL[3,3]);                      00433000
DOUBLE(M2H,M2L,MP4,0,/,DFCONH[1,1],DFCONL[1,1]);                            00434000
DOUBLE(-1.0,MP2,0,/,DFCONH[2,1],DFCONL[2,1]);                               00435000
DOUBLE(2.0,M4H,M4L,×,MP2,0,/,DFCONH[2,2],DFCONL[2,2]);                      00436000
DOUBLE(0.5,M6H,M6L,×,MP5,0,/,DFCONH[3,1],DFCONL[3,1]);                      00437000
DOUBLE(MP4,0,MP5,0,/,M9H,M9L);                                              00438000
M10H←1;                                                                     00439000
```

```
M10L←0;                                                                    00440000
FOR I←1 STEP 1 UNTIL MP1 DO DOUBLE(M10H,M10L,M9H,M9L,×,←,M10H,M10L);       00441000
DOUBLE(,0.25,,5.0,M,0,×,,16.0,+,×,M10H,M10L,×,M7H,M7L,/,←,DFCONH[3,2],     00442000
DFCONL[3,2]);                                                              00443000
DOUBLE(,-0.75,MP2,0,×,M6H,M6L,×,M7H,M7L,/,←,DFCONH[3,3],DFCONL[3,3]);      00444000
CH[1]←CL[1]←0;                                                             00445000
DOUBLE(M4H,M4L,M8H,M8L,/,←,CH[2],CL[2]);                                   00446000
CH[3]←CL[3]←0;                                                             00447000
CH[4] ← CL[4] ← 0 ;                                                        00448000
CPMH[1]←CPML[1]←0;                                                         00449000
DOUBLE(,0.5,MP4,0,×,M4H,M4L,×,M8H,M8L,/,←,CPMH[2],CPML[2]);                00450000
DOUBLE(,0.5,MP3,0,/,←,CPMH[3],CPML[3]);                                    00451000
CHATH[1]←CHATL[1]←0;                                                       00452000
CHATH[2]←CHATL[2]←0;                                                       00453000
DOUBLE(,0.33333333333333333333333,MP3,0,MP4,0,×,/,←,CHATH[3],CHATL[3]     00454000
);                                                                         00455000
DOUBLE(MP5,0,MP2,0,/,←,M11H,M11L);                                         00456000
M12H←1;                                                                    00457000
M12L←0;                                                                    00458000
FOR I←1 STEP 1 UNTIL MP1 DO DOUBLE(M12H,M12L,M11H,M11L,×,←,M12H,M12L);     00459000
DOUBLE(,0.66666666666666666667,MP5,0,×,M12H,M12L,×,M8H,M8L,MP4,0, ×,       00460000
/,←,CHATH[4],CHATL[4]);                                                    00461000
FOR I ← 1 STEP 1 UNTIL 4 DO DOUBLE(CH[I],CL[I],CHATH[I],CHATL[I],-,←,      00462000
CDIFH[I],CDIFL[I]);                                                        00463000
ITERATE: DUB1 ← TRUE;                                                      00464000
NEW←FALSE;                                                                 00465000
XOH ← XSTARTH;                                                             00466000
XOL ← XSTARTL;                                                             00467000
DXOH ← DXSTARTH;                                                           00468000
DXOL ← DXSTARTL;                                                           00469000
YOH ← YSTARTH;                                                             00470000
YOL ← YSTARTL;                                                             00471000
TH ← TSTARTH;                                                              00472000
TL ← TSTARTL;                                                              00473000
HH ← HSTARTH;                                                              00474000
HL ← HSTARTL;                                                              00475000
DOUBLE (DYSTARTH,DYSTARTL,←,DYOH,DYOL);                                    00476000
DOUBLE(XOH,XOL,MUH,MUL,+,←,RH,RL);                                         00477000
RH ← ABS(RH);                                                              00478000
DOUBLE(XOH,XOL,MUPMH,MUPML,-,←,SH,SL);                                     00479000
```

```
SH ← ABS(SH);                                                          00480000
JH ← DJH ← 1 ;                                                         00481000
JL ← DJL ← 0 ;                                                         00482000
IF ITER THEN GO TO PRTEST ;                                            00483000
GO TO PRNT ;                                                           00484000
PRTEST: IF PRITR THEN PRNT ;                                           00485000
BEGIN                                                                  00486000
  IF CC = 3 THEN                                                       00487000
  BEGIN                                                                00488000
    CC ← 1 ;                                                           00489000
    WRITE (PRINT[PAGE]) ;                                              00490000
                                                                       00491000
  END ;                                                                00492000
  DPOP (TH,TL,PRINT,OT1,A);                                            00493000
  DPOP (XOH,XOL,PRINT,OT2,A);                                          00494000
  DPOP (YOH,YOL,PRINT,OT3,A);                                          00495000
  DPOP (DXOH,DXOL,PRINT,OT4,A);                                        00496000
  DPOP (DYOH,DYOL,PRINT,OT5,A);                                        00497000
  DPOP (HH,HL,PRINT,OT6,A);                                            00498000
  DPOP (JH,JL,PRINT,OT7,A);                                            00499000
  DPOP (DJH,DJL,PRINT,OT8,A);                                          00500000
  DPOP (RH,RL,PRINT,OT9,A);                                            00501000
  DPOP (SH,SL,PRINT,OT10,A);                                           00502000
  WRITE (PRINT,LFMT1,PLINE1) ;                                         00503000
                                                                       00504000
END ;                                                                  00505000
KOUNT ← 0 ;                                                            00506000
T ← TIME (2);                                                          00507000
COMMENT FOLLOWING IS THE POWER SERIES EXPANSION CALCULATIONS;          00508000
                                                                       00509000
BEGIN                                                                  00510000
  LABEL L1,RETURN;                                                     00511000
  LABEL NEW1;                                                          00512000
  LABEL TSTEP,LOOP,HALVE,SWAP,STEST,CONVERT,TEST,INTERP,QUIT;          00513000
  TSTEP: DOUBLE(XOH,XOL,←,XNUH[0],XNUL[0]);                            00514000
  KOUNT ← KOUNT + 1;                                                   00515000
  DUB ← TRUE ;                                                         00516000
  DOUBLE(DXOH,DXOL,←,XNUH[1],XNUL[1]);                                 00517000
  DOUBLE(YOH,YOL,←,YNUH[0],YNUL[0]);                                   00518000
  DOUBLE(DYOH,DYOL,←,YNUH[1],YNUL[1]);                                 00519000
```

```
DOUBLE(YOH,YOL,YOH,YOL,×,←,YOSQH,YOSQL);                                    00520000
DOUBLE (XOH,XOL,MUH,MUL,+,XOH,XOL,MUH,MUL,+,×,YOSQH,YOSQL,+,←,RNH,          00521000
RNL);                                                                       00522000
X1 ← RNH;                                                                   00523000
X2 ← RNL;                                                                   00524000
DDSQRT;                                                                     00525000
RNUH[0] ← H1Y;                                                              00526000
RNUL[0] ← L1Y;                                                              00527000
DOUBLE(XOH,XOL,MUPMH,MUPML,-,XOH,XOL,MUPMH,MUPML,-,×,YOSQH,YOSQL, +,00528000
←,SNH,SNL);                                                                 00529000
X1 ← SNH;                                                                   00530000
X2 ← SNL;                                                                   00531000
DDSQRT;                                                                     00532000
SNUH[0] ← H1Y;                                                              00533000
SNUL[0] ← L1Y;                                                              00534000
DOUBLE(MUPMH,MUPML,RNUH[0],RNUL[0],RNUH[0],RNUL[0],RNUH[0], RNUL[0],00535000
×,×,/,←,UNUH[0],UNUL[0]);                                                   00536000
DOUBLE (MUH,MUL,SNUH[0],SNUL[0],SNUH[0],SNUL[0],SNUH[0],SNUL[0], ×,×00537000
,/,←,VNUH[0],VNUL[0]);                                                      00538000
FOR N←1 STEP 1 UNTIL MP1 DO                                                 00539000
BEGIN                                                                       00540000
  NP1←N+1;                                                                  00541000
  NM1←N-1;                                                                  00542000
  D1←2×N;                                                                   00543000
  D2←N×NP1;                                                                 00544000
  TERMH←TERML←0;                                                            00545000
  FOR I←0 STEP 1 UNTIL N DO                                                 00546000
  BEGIN                                                                     00547000
    NMI←N-I;                                                                00548000
    DOUBLE(TERMH,TERML,XNUH[I],XNUL[I],XNUH[NMI],XNUL[NMI], ×,YNUH[I00549000
    ],YNUL[I],YNUH[NMI],YNUL[NMI],×,+,+,←, TERMH,TERML);                    00550000
                                                                            00551000
  END;                                                                      00552000
  TERM1H←TERM1L←0;                                                          00553000
  TERM2H←TERM2L←0;                                                          00554000
  TERM3H←TERM3L←0;                                                          00555000
  TERM4H←TERM4L←0;                                                          00556000
  IF N≠1 THEN                                                               00557000
  BEGIN                                                                     00558000
    FOR I←1 STEP 1 UNTIL NM1 DO                                             00559000
```

```
    BEGIN                                                                    00560000
      NMI←N-I;                                                               00561000
      DOUBLE(TERM1H,TERM1L,RNUH[I],RNUL[I],RNUH[NMI], RNUL[NMI],×,+,00562000
      ←,TERM1H,TERM1L);                                                      00563000
      DOUBLE(TERM2H,TERM2L,SNUH[I],SNUL[I],SNUH[NMI], SNUL[NMI],×,+,00564000
      ←,TERM2H,TERM2L);                                                      00565000
      DOUBLE(TERM3H,TERM3L,I,0,UNUH[I],UNUL[I],RNUH[NMI], RNUL[NMI],00566000
      ×,×,+,←,TERM3H,TERM3L);                                                00567000
      DOUBLE(TERM4H,TERM4L,I,0,VNUH[I],VNUL[I],SNUH[NMI], SNUL[NMI],00568000
      ×,×,+,←,TERM4H,TERM4L);                                                00569000
                                                                             00570000
    END;                                                                     00571000
                                                                             00572000
  END;                                                                       00573000
  DOUBLE(TERMH,TERML,C1H,C1L,XNUH[N],XNUL[N],×,+,TERM1H, TERM1L,-,,  00574000
  2.0,RNUH[0],RNUL[0],×,/, ←,RNUH[N], RNUL[N]);                              00575000
  DOUBLE(TERMH,TERML,C2H,C2L,XNUH[N],XNUL[N],×,-,TERM2H, TERM2L,-,,  00576000
  2.0,SNUH[0],SNUL[0],×,/,←,SNUH[N],SNUL[N]);                                00577000
  UNUNH←UNUNL←0;                                                             00578000
  VNUNH←VNUNL←0;                                                             00579000
  FOR I←1 STEP 1 UNTIL N DO                                                  00580000
  BEGIN                                                                      00581000
    NMI←N-I;                                                                 00582000
    DOUBLE(UNUNH,UNUNL,I,0,RNUH[I],RNUL[I],UNUH[NMI], UNUL[NMI],×,×,00583000
    +,←,UNUNH,UNUNL);                                                        00584000
    DOUBLE(VNUNH,VNUNL,I,0,SNUH[I],SNUL[I],VNUH[NMI], VNUL[NMI],×,×,00585000
    +,←,VNUNH,VNUNL);                                                        00586000
                                                                             00587000
  END;                                                                       00588000
  DOUBLE(,-3.0,UNUNH,UNUNL,×,TERM3H,TERM3L,-,N,0,RNUH[0], RNUL[0],×,00589000
  /,←,UNUH[N],UNUL[N]);                                                      00590000
  DOUBLE(,-3,VNUNH,VNUNL,×,TERM4H,TERM4L,-,N,0,SNUH[0], SNUL[0],×,/,00591000
  ←,VNUH[N],VNUL[N]);                                                        00592000
  FOR I←0 STEP 1 UNTIL NM1 DO DOUBLE(UNUH[I],UNUL[I],VNUH[I],VNUL[I]00593000
  ,+,←,SIGH[I],SIGL[I]);                                                     00594000
  XNUNP1H←XNUNP1L←0;                                                         00595000
  YNUNP1H←YNUNP1L←0;                                                         00596000
  FOR I←0 STEP 1 UNTIL NM1 DO                                                00597000
  BEGIN                                                                      00598000
    NM1MI←NM1-I;                                                             00599000
```

```
    DOUBLE(XNUNP1H,XNUNP1L,SIGH[I],SIGL[I],XNUH[NM1MI], XNUL[NM1MI],00600000
    X,+,←,XNUNP1H,XNUNP1L);                                          00601000
    DOUBLE(YNUNP1H,YNUNP1L,SIGH[I],SIGL[I],YNUH[NM1MI], YNUL[NM1MI],00602000
    X,+,←,YNUNP1H,YNUNP1L);                                          00603000
                                                                     00604000
  END;                                                               00605000
  DOUBLE (XNUH[NM1],XNUL[NM1],D1,0,YNUH[N],YNUL[N],X,+,MUH,MUL, UNUH00606000
  [NM1],UNUL[NM1],X,=,MUPMH,MUPML,VNUH[NM1], VNUL[NM1],X,+,XNUNP1H, 00607000
  XNUNP1L,=,D2,0,/,←,XNUH[NP1], XNUL[NP1]);                          00608000
  DOUBLE(YNUH[NM1],YNUL[NM1],D1,0,XNUH[N],XNUL[N],X,=, YNUNP1H,      00609000
  YNUNP1L,=,D2,0,/,←,YNUH[NP1],YNUL[NP1]);                           00610000
                                                                     00611000
END;                                                                 00612000
COMMENT FOLLOWING IS THE CALCULATION OF THE K-S AND L-S;             00613000
LOOP: FOR LAMDA ← 1 STEP 1 UNTIL 4 DO                                00614000
BEGIN                                                                00615000
  LM1 ← LAMDA - 1 ;                                                  00616000
  FVALXTH ← 0 ;                                                      00617000
  FVALXTL ← 0 ;                                                      00618000
  FVALYTL ← 0 ;                                                      00619000
  FVALYTH ← 0 ;                                                      00620000
  FOR I ← 1 STEP 1 UNTIL LM1 DO                                      00621000
  BEGIN                                                              00622000
    DOUBLE (FVALXTH,FVALXTL,FCONH[LM1,I],FCONL[LM1,I],KH[I], KL[I],X00623000
    ,+,←,FVALXTH,FVALXTL);                                           00624000
    DOUBLE (FVALYTH,FVALYTL,FCONH[LM1,I],FCONL[LM1,I],LH[I], LL[I],X00625000
    ,+,←,FVALYTH,FVALYTL);                                           00626000
                                                                     00627000
  END;                                                               00628000
  DOUBLE (XOH,XOL,HH,HL,FVALXTH,FVALXTL,X,+,←,XTH,XTL);              00629000
  DFVALXTH ← DFVALXTL ← 0 ;                                          00630000
  DOUBLE (YOH,YOL,HH,HL,FVALYTH,FVALYTL,X,+,←,YTH,YTL);              00631000
  DFVALYTH ← DFVALYTL ← 0 ;                                          00632000
  FOR I ← 1 STEP 1 UNTIL LM1 DO                                      00633000
  BEGIN                                                              00634000
    DOUBLE (DFVALXTH,DFVALXTL,DFCONH[LM1,I],DFCONL[LM1,I], KH[I],KL 00635000
    [I],X,+,←,DFVALXTH,DFVALXTL);                                    00636000
    DOUBLE (DFVALYTH,DFVALYTL,DFCONH[LM1,I],DFCONL[LM1,I], LH[I],LL 00637000
    [I],X,+,←,DFVALYTH,DFVALYTL);                                    00638000
                                                                     00639000
```

```
END;                                                                    00640000
DOUBLE ( DFVALXTH,DFVALXTL, ←,DXTH,DXTL);                              00641000
DOUBLE ( DFVALYTH,DFVALYTL, ←,DYTH,DYTL);                              00642000
DOUBLE (ALFH[LAMDA],ALFL[LAMDA],HH,HL,×,←,DTH,DTL);                    00643000
SUM3XH ← SUM3XL ← 0 ;                                                  00644000
SUM3YH ← SUM3YL ← 0 ;                                                  00645000
FOR I ← MP2 STEP -1 UNTIL 1 DO                                         00646000
BEGIN                                                                  00647000
  DOUBLE(SUM3XH,SUM3XL,XNUH[I],XNUL[I],+,DTH,DTL,×,←,SUM3XH,SUM3XL00648000
  );                                                                   00649000
  DOUBLE(SUM3YH,SUM3YL,YNUH[I],YNUL[I],+,DTH,DTL,×,←,SUM3YH,SUM3YL00650000
  );                                                                   00651000
                                                                       00652000
END;                                                                   00653000
DOUBLE (XTH,XTL,SUM3XH,SUM3XL,÷,←,XH,XL);                              00654000
DOUBLE (YTH,YTL,SUM3YH,SUM3YL,÷,←,YH,YL);                              00655000
SUM4XH ← SUM4XL ← 0 ;                                                  00656000
SUM4YH ← SUM4YL ← 0 ;                                                  00657000
FOR I ← MP1 STEP -1 UNTIL 1 DO                                         00658000
BEGIN                                                                  00659000
  IM1 ← I+1;                                                           00660000
  DOUBLE(SUM4XH,SUM4XL,XNUH[IM1],XNUL[IM1],IM1,0,×,+,DTH,DTL,×,←,  00661000
  SUM4XH, SUM4XL);                                                     00662000
  DOUBLE(SUM4YH,SUM4YL,YNUH[IM1],YNUL[IM1],IM1,0,×,+,DTH,DTL,×,←,  00663000
  SUM4YH, SUM4YL);                                                     00664000
                                                                       00665000
END;                                                                   00666000
DOUBLE(SUM4XH,SUM4XL,XNUH[1],XNUL[1],+,←,SUM4XH,SUM4XL);               00667000
DOUBLE(SUM4YH,SUM4YL,YNUH[1],YNUL[1],+,←,SUM4YH,SUM4YL);               00668000
DOUBLE (DXTH,DXTL,SUM4XH,SUM4XL,÷,←,DXH,DXL);                         00669000
DOUBLE (YH,YL,YH,YL,×,←,YSQH,YSQL);                                    00670000
DOUBLE (DYTH,DYTL,SUM4YH,SUM4YL,+,←,DYH,DYL);                          00671000
DOUBLE (XH,XL,MUH,MUL,÷,XH,XL,MUH,MUL,÷,×,YSQH,YSQL,+,←, DON1H,    00672000
DON1L);                                                                00673000
DOUBLE (DON1H,DON1L,DON1H,DON1L,DON1H,DON1L,×,×,←,DON1H, DON1L);       00674000
X1 ← DON1H;                                                            00675000
X2 ← DON1L;                                                            00676000
DDSQRT;                                                                00677000
DEN1H ← H1Y;                                                           00678000
DEN1L ← L1Y;                                                           00679000
```

```
    DOUBLE (XH,XL,MUPMH,MUPML,-,XH,XL,MUPMH,MUPML,-,X,YSQH,YSQL, +,←, 00680000
    DON2H,DON2L);                                                    00681000
    DOUBLE (DON2H,DON2L,DON2H,DON2L,DON2H,DON2L,X,X,←,DON2H, DON2L); 00682000
    X1 ← DON2H;                                                      00683000
    X2 ← DON2L;                                                      00684000
    DDSQRT;                                                          00685000
    DEN2H ← H1Y;                                                     00686000
    DEN2L ← L1Y;                                                     00687000
    DOUBLE (XH,XL,,2, DYH,DYL,X,÷,MUPMH,MUPML,XH,XL,MUH,MUL,÷,X, DEN1H00688000
    ,DEN1L,/,-,MUH,MUL,XH,XL,MUPMH,MUPML,-,X,DEN2H, DEN2L,/,-,←,FH,FL)00689000
    ;                                                                00690000
    DOUBLE (YH,YL,,2,DXH,DXL,X,-,YH,YL,MUPMH,MUPML,DEN1H,DEN1L, /, MUH00691000
    ,MUL,DEN2H,DEN2L,/,+,X,-,←,GH,GL);                               00692000
    SUM5FTH ← SUM5FTL ← 0 ;                                          00693000
    SUM5GTH ← SUM5GTL ← 0 ;                                          00694000
    FOR I ← M STEP -1 UNTIL 1 DO                                     00695000
    BEGIN                                                            00696000
      IM2 ← I+2;                                                     00697000
      IM1 ← I+1;                                                     00698000
      DOUBLE(SUM5FTH,SUM5FTL,XNUH[IM2],XNUL[IM2],IM2,0,X,IM1,0,X,+,DTH00699000
      ,DTL,X, ←,SUM5FTH,SUM5FTL);                                    00700000
      DOUBLE(SUM5GTH,SUM5GTL,YNUH[IM2],YNUL[IM2],IM2,0,X,IM1,0,X,+,DTH00701000
      ,DTL,X, ←,SUM5GTH,SUM5GTL);                                    00702000
                                                                     00703000
    END;                                                             00704000
    DOUBLE(SUM5FTH,SUM5FTL,,2.0,XNUH[2],XNUL[2],X,+,←,SUM5FTH,SUM5FTL)00705000
    ;                                                                00706000
    DOUBLE(SUM5GTH,SUM5GTL,,2.0,YNUH[2],YNUL[2],X,+,←,SUM5GTH,SUM5GTL)00707000
    ;                                                                00708000
    DOUBLE (FH,FL,SUM5FTH,SUM5FTL,-,←,FTH,FTL);                      00709000
    DOUBLE (GH,GL,SUM5GTH,SUM5GTL,-,←,GTH,GTL);                      00710000
    DOUBLE (FTH,FTL,HH,HL,X,←,KH[LAMDA],KL[LAMDA]);                  00711000
    DOUBLE (GTH,GTL,HH,HL,X,←,LH[LAMDA],LL[LAMDA]);                  00712000
                                                                     00713000
  END;                                                               00714000
  COMMENT FOLLOWING IS CHECK ON TRUNCATION ERROR;                    00715000
  SUM6TXH ← SUM6TXL ← SUM6TYH ← SUM6TYL ← 0 ;                        00716000
  FOR I ← 1 STEP 1 UNTIL 4 DO                                        00717000
  BEGIN                                                              00718000
    DOUBLE (SUM6TXH,SUM6TXL,CDIFH[I],CDIFL[I],KH[I],KL[I],X,+, ←,    00719000
```

```
  SUM6TXH,SUM6TXL);                                                    00720000
  DOUBLE (SUM6TYH,SUM6TYL,CDIFH[I],CDIFL[I],LH[I],LL[I],×,+, ←,        00721000
  SUM6TYH,SUM6TYL);                                                    00722000
                                                                       00723000
END;                                                                   00724000
IF NEW THEN GO TO NEW1;                                                00725000
DOUBLE (HH,HL,SUM6TXH,SUM6TXL,×,←,TXH,TXL);                           00726000
TXH ← ABS(TXH);                                                        00727000
DOUBLE (HH,HL,SUM6TYH,SUM6TYL,×,←,TYH,TYL);                           00728000
TYH ← ABS(TYH);                                                        00729000
DOUBLE (XOH,XOL,EPS,0,×,←,EPSXH,EPSXL);                               00730000
EPSXH ← ABS(EPSXH);                                                    00731000
DOUBLE (YOH,YOL,EPS,0,×,←,EPSYH,EPSYL);                               00732000
EPSYH ← ABS(EPSYH);                                                    00733000
IF EPSXH = 0 THEN DOUBLE(EPS,0,←,ERRXH,ERRXL) ELSE DOUBLE(TXH,TXL,    00734000
EPSXH,EPSXL,/,←,ERRXH,ERRXL);                                          00735000
IF EPSYH = 0 THEN DOUBLE(EPS,0,←,ERRYH,ERRYL) ELSE DOUBLE(TYH,TYL,    00736000
EPSYH,EPSYL,/,←,ERRYH,ERRYL);                                          00737000
IF (ERRXH> 1 ) OR ( ERRYH> 1 ) THEN HALVE;                             00738000
BEGIN                                                                  00739000
  DOUBLE (,0.5,HH,HL,×,←,HH,HL);                                       00740000
  DUB ← FALSE;                                                         00741000
  GO TO LOOP                                                           00742000
END;                                                                   00743000
IF DUB AND DUB1 THEN                                                   00744000
BEGIN                                                                  00745000
  IF (ERRXH<EPSM) AND (ERRYH<EPSM) THEN                                00746000
  BEGIN                                                                00747000
    DOUBLE (HH,HL,,2,×,←,HH,HL);                                       00748000
    GO TO LOOP;                                                        00749000
                                                                       00750000
  END;                                                                 00751000
                                                                       00752000
END;                                                                   00753000
NEW1; COMMENT FOLLOWING IS CALCULATION OF VALUES AT RIGHT END POINT;   00754000
SUM6XT1H ← SUM6XT1L ← SUM6YT1H ← SUM6YT1L ← 0 ;                        00755000
FOR I ← 1 STEP 1 UNTIL 3 DO                                            00756000
BEGIN                                                                  00757000
  DOUBLE (SUM6XT1H,SUM6XT1L,CH[I],CL[I],KH[I],KL[I],×,+,←, SUM6XT1H    00758000
  ,SUM6XT1L);                                                          00759000
```

```
  DOUBLE (SUM6YT1H,SUM6YT1L,CH[I],CL[I],LH[I],LL[I],×,+,←, SUM6YT1H 00760000
  ,SUM6YT1L);                                                      00761000
                                                                   00762000
END;                                                               00763000
DOUBLE (XOH,XOL,HH,HL,SUM6XT1H,SUM6XT1L,×,+,←,XT1H,XT1L);          00764000
DOUBLE (YOH,YOL,HH,HL,SUM6YT1H,SUM6YT1L,×,+,←,YT1H,YT1L);          00765000
SUM6DXT1H ← SUM6DXT1L ← SUM6DYT1H ← SUM6DYT1L ← 0 ;                00766000
FOR I ←1 STEP 1 UNTIL 3 DO                                         00767000
BEGIN                                                              00768000
  DOUBLE (SUM6DXT1H,SUM6DXT1L,CPMH[I],CPML[I],KH[I],KL[I],×, +,←, 00769000
  SUM6DXT1H,SUM6DXT1L);                                            00770000
  DOUBLE (SUM6DYT1H,SUM6DYT1L,CPMH[I],CPML[I],LH[I],LL[I],×, +,←, 00771000
  SUM6DYT1H,SUM6DYT1L);                                            00772000
                                                                   00773000
END;                                                               00774000
DXT1H ← SUM6DXT1H ;                                                00775000
DXT1L ← SUM6DXT1L ;                                                00776000
DYT1H ← SUM6DYT1H ;                                                00777000
DYT1L ← SUM6DYT1L ;                                                00778000
SUM3X1H ← SUM3X1L ← SUM3Y1H ← SUM3Y1L ← 0 ;                        00779000
FOR I ← MP2 STEP -1 UNTIL 1 DO                                     00780000
BEGIN                                                              00781000
  DOUBLE(SUM3X1H,SUM3X1L,XNUH[I],XNUL[I],+,HH,HL,×,←,SUM3X1H,SUM3X1L00782000
  );                                                               00783000
  DOUBLE(SUM3Y1H,SUM3Y1L,YNUH[I],YNUL[I],+,HH,HL,×,←,SUM3Y1H,SUM3Y1L00784000
  );                                                               00785000
                                                                   00786000
END;                                                               00787000
DOUBLE (XT1H,XT1L,SUM3X1H,SUM3X1L,+,←,X1H,X1L) ;                   00788000
DOUBLE (YT1H,YT1L,SUM3Y1H,SUM3Y1L,+,←,Y1H,Y1L) ;                   00789000
SUM4DX1H ← SUM4DX1L ← SUM4DY1H ← SUM4DY1L ← 0 ;                    00790000
FOR I ← MP1 STEP -1 UNTIL 1 DO                                     00791000
BEGIN                                                              00792000
  IM1 ← I+1;                                                       00793000
  DOUBLE(SUM4DX1H,SUM4DX1L,XNUH[IM1],XNUL[IM1],IM1,0,×,+,HH,HL,×,←, 00794000
  SUM4DX1H,SUM4DX1L);                                              00795000
  DOUBLE(SUM4DY1H,SUM4DY1L,YNUH[IM1],YNUL[IM1],IM1,0,×,+,HH,HL,×,←, 00796000
  SUM4DY1H,SUM4DY1L);                                              00797000
                                                                   00798000
END;                                                               00799000
```

```
DOUBLE(SUM4DX1H,SUM4DX1L,XNUH[1],XNUL[1],+,←,SUM4DX1H,SUM4DX1L);     00800000
DOUBLE(SUM4DY1H,SUM4DY1L,YNUH[1],YNUL[1],+,←,SUM4DY1H,SUM4DY1L);     00801000
DOUBLE (DXT1H,DXT1L,SUM4DX1H,SUM4DX1L,+,←,DX1H,DX1L) ;               00802000
DOUBLE (DYT1H,DYT1L,SUM4DY1H,SUM4DY1L,+,←,DY1H,DY1L) ;               00803000
DOUBLE (TH,TL,HH,HL,+,←,TH,TL) ;                                    00804000
DOUBLE (Y1H,Y1L,Y1H,Y1L,×,←,Y1SQH,Y1SQL) ;                          00805000
DOUBLE (X1H,X1L,MUH,MUL,+,X1H,X1L,MUH,MUL,+,×,Y1SQH,Y1SQL,+, ←,DON1H00806000
,DON1L) ;                                                           00807000
X1 ← DON1H;                                                         00808000
X2 ← DON1L;                                                         00809000
DDSQRT;                                                             00810000
RH ← H1Y;                                                           00811000
RL ← L1Y;                                                           00812000
DOUBLE (X1H,X1L,MUPMH,MUPML,-,X1H,X1L,MUPMH,MUPML,-,×,Y1SQH, Y1SQL,+00813000
,←,DON1H,DON1L) ;                                                   00814000
X1 ← DON1H;                                                         00815000
X2 ← DON1L;                                                         00816000
DDSQRT;                                                             00817000
SH ← H1Y;                                                           00818000
SL ← L1Y;                                                           00819000
DOUBLE (,0.5,DX1H,DX1L,DX1H,DX1L,×,DY1H,DY1L,DY1H,DY1L,×,+, X1H,X1L 00820000
,X1H,X1L,×,-,Y1H,Y1L,Y1H,Y1L,×,-,×,MUPMH,MUPML, RH,RL,/,-,MUH,MUL,SH00821000
,SL,/,-,←,JH,JL);                                                   00822000
DOUBLE (JH,JL,J1H,J1L,-,←,DJH,DJL) ;                                00823000
DJH ← ABS(DJH) ;                                                    00824000
J1H ← JH ;                                                          00825000
J1L ← JL ;                                                          00826000
COMMENT FOLLOWING ARE TESTS ON ORBIT;                               00827000
                                                                    00828000
BEGIN                                                               00829000
  LABEL TEST;                                                       00830000
  LABEL QUIT,STEST,CONVERT,INTERP,SWAP;                             00831000
  TEST: IF TH < TTEST THEN GO TO SWAP ;                             00832000
  DUB1 ← FALSE;                                                     00833000
  IF SIGN(Y1H) = SAME THEN GO TO SWAP ;                             00834000
  IF ABS(Y1H) > EPS1 THEN                                           00835000
  BEGIN                                                             00836000
    DOUBLE (TH,TL,HH,HL,-,←,TH,TL) ;                                00837000
    DOUBLE (YOH,YOL,YOH,YOL,Y1H,Y1L,-,/,HH,HL,×,←,HH,HL);           00838000
    NEW←TRUE;                                                       00839000
```

```
    GO TO LOOP;                                                                    00840000
                                                                                   00841000
  END ;                                                                            00842000
  IF NOT ITER THEN GO TO QUIT ;                                                    00843000
  IF ABS(DX1H) < EPS2 THEN                                                         00844000
  BEGIN                                                                            00845000
    ITER ← FALSE ;                                                                 00846000
    TTEST ← (0.99) × PER;                                                          00847000
    SAME ← SIGN(DIR2) ;                                                            00848000
    WRITE (PRINT[PAGE]) ;                                                          00849000
    CC ← 1 ;                                                                       00850000
    GO TO ITERATE ;                                                                00851000
                                                                                   00852000
  END ;                                                                            00853000
  IF ABS(DX1H) > ABS (DXAH) THEN                                                   00854000
  BEGIN                                                                            00855000
    COUNT ← COUNT + 1;                                                             00856000
    IF ABS(DXAH) < ABS(HLDDXAH) THEN                                               00857000
    BEGIN                                                                          00858000
      DOUBLE (DXAH,DXAL,←,HLDDXAH,HLDDXAL);                                        00859000
      DOUBLE (DYAH,DYAL,←,HLDDYAH,HLDDYAL);                                        00860000
                                                                                   00861000
    END;                                                                           00862000
                                                                                   00863000
  END;                                                                             00864000
  IF COUNT > 2 THEN                                                                00865000
  BEGIN                                                                            00866000
    DOUBLE(HLDDYAH,HLDDYAL,←,DYSTARTH,DYSTARTL);                                   00867000
    ITER ← FALSE ;                                                                 00868000
    TTEST ← (0.99) × PER;                                                          00869000
    SAME ← SIGN(DIR2) ;                                                            00870000
    WRITE (PRINT[PAGE]) ;                                                          00871000
    CC ← 1 ;                                                                       00872000
    GO TO ITERATE ;                                                                00873000
                                                                                   00874000
  END ;                                                                            00875000
  INTERP: DOUBLE (DYAH,DYAL,DX1H,DX1L,×,DXAH,DXAL,DYSTARTH,DYSTARTL,               00876000
  ×,-, DX1H,DX1L,DXAH,DXAL,-,/,←,DYHOLDH,DYHOLDL) ;                                00877000
  DXAH ← DX1H ;                                                                    00878000
  DXAL ← DX1L ;                                                                    00879000
```

```
DYAH ← DYSTARTH;                                                        00880000
DYAL ← DYSTARTL ;                                                       00881000
DYSTARTH ← DYHOLDH ;                                                    00882000
DYSTARTL ← DYHOLDL ;                                                    00883000
GO TO ITERATE ;                                                         00884000
SWAP: X0H ← X1H;                                                        00885000
X0L ← X1L;                                                              00886000
Y0H ← Y1H;                                                              00887000
Y0L ← Y1L;                                                              00888000
DX0H ← DX1H ;                                                           00889000
DX0L ← DX1L ;                                                           00890000
DY0H ← DY1H ;                                                           00891000
DY0L ← DY1L ;                                                           00892000
IF ITER THEN GO TO STEST ;                                              00893000
IF PRNTSTEP THEN                                                        00894000
BEGIN                                                                   00895000
  IF CC = 3 THEN                                                        00896000
  BEGIN                                                                 00897000
    CC ←1 ;                                                             00898000
    WRITE (PRINT[PAGE]) ;                                               00899000
                                                                        00900000
  END ;                                                                 00901000
  CONVERT: DPOP (TH,TL,PRINT,0T1,A);                                    00902000
  DPOP (X0H,X0L,PRINT,0T2,A);                                           00903000
  DPOP (Y0H,Y0L,PRINT,0T3,A);                                           00904000
  DPOP (DX0H,DX0L,PRINT,0T4,A);                                         00905000
  DPOP (DY0H,DY0L,PRINT,0T5,A);                                         00906000
  DPOP (HH,HL,PRINT,0T6,A);                                             00907000
  DPOP (JH,JL,PRINT,0T7,A);                                             00908000
  DPOP (DJH,DJL,PRINT,0T8,A);                                           00909000
  DPOP (RH,RL,PRINT,0T9,A);                                             00910000
  DPOP (SH,SL,PRINT,0T10,A);                                            00911000
  WRITE (PRINT,LFMT1,PLINE1) ;                                          00912000
                                                                        00913000
END ;                                                                   00914000
STEST: IF (RH<RMIN) OR (RH>RMAX) OR (SH<SMIN) OR (SH>SMAX) OR (TH       00915000
>TMAX) THEN                                                             00916000
BEGIN                                                                   00917000
  DPOP (TH,TL,PRINT,0T1,A);                                             00918000
  DPOP (X0H,X0L,PRINT,0T2,A);                                           00919000
```

```
        DPOP (YOH,YOL,PRINT,OT3,A);                                    00920000
        DPOP (DXOH,DXOL,PRINT,OT4,A);                                  00921000
        DPOP (DYOH,DYOL,PRINT,OT5,A);                                  00922000
        DPOP (HH,HL,PRINT,OT6,A);                                      00923000
        DPOP (JH,JL,PRINT,OT7,A);                                      00924000
        DPOP (DJH,DJL,PRINT,OT8,A);                                    00925000
        DPOP (RH,RL,PRINT,OT9,A);                                      00926000
        DPOP (SH,SL,PRINT,OT10,A);                                     00927000
        WRITE (PRINT,LFMT1,PLINE1) ;                                   00928000
        WRITE(PRINT,KOFMT,KOUNT);                                      00929000
        IF NOT PLT THEN GO TO START;                                   00930000
                                                                       00931000
      END ;                                                            00932000
      GO TO TSTEP ;                                                    00933000
      QUIT: IF CC = 3 THEN WRITE (PRINT[PAGE]) ;                       00934000
      WRITE(PRINT,FMTT,(TIME(2)-T)/60);                                00935000
      DPOP (TH,TL,PRINT,OT1,A) ;                                       00936000
      DPOP (X1H,X1L,PRINT,OT2,A) ;                                     00937000
      DPOP (Y1H,Y1L,PRINT,OT3,A) ;                                     00938000
      DPOP (DX1H,DX1L,PRINT,OT4,A) ;                                   00939000
      DPOP (DY1H,DY1L,PRINT,OT5,A) ;                                   00940000
      DPOP (HH,HL,PRINT,OT6,A) ;                                       00941000
      DPOP (JH,JL,PRINT,OT7,A) ;                                       00942000
      DPOP (DJH,DJL,PRINT,OT8,A) ;                                     00943000
      DPOP (RH,RL,PRINT,OT9,A) ;                                       00944000
      DPOP (SH,SL,PRINT,OT10,A) ;                                      00945000
      WRITE (PRINT,LFMT1,PLINE1) ;                                     00946000
      WRITE(PRINT,KOFMT,KOUNT);                                        00947000
      IF NOT PLT THEN GO TO START;                                     00948000
                                                                       00949000
    END ;                                                              00950000
                                                                       00951000
  END;                                                                 00952000
  LAST: CLOSE(CARD,RELEASE);                                           00953000
                                                                       00954000
END.                                                                   00955000
 30         END-OF-FILE CARD
 LABEL    00000000CARD    001      01
```

E. The Method of Runge-Kutta-Shanks

1. Introduction

The procedure described here is a generalization of the Runge-Kutta method for solving a system of differential equations. It is applied to the system of second-order differential equations

$$\ddot{x} = x + 2\dot{y} - \mu' \frac{(x+\mu)}{[(x+\mu)^2+y^2]^{3/2}} - \mu \frac{(x-\mu')}{[(x-\mu')^2+y^2]^{3/2}},$$

$$y = y - 2\dot{x} - \mu' \frac{y}{[(x+\mu)^2+y^2]^{3/2}} - \mu \frac{y}{[(x-\mu')^2+y^2]^{3/2}},$$

with the initial conditions

$$x(0) = x_0, \qquad \dot{x}(0) = \dot{x}_0,$$

$$y(0) = y_0, \qquad \dot{y}(0) = \dot{y}_0.$$

The constant μ is the ratio of the mass of the moon to the combined mass of the earth and the moon, while $\mu' = 1 - \mu$.

Before applying the Shanks method, this system must be reduced to the first-order system:

$$\dot{x} = u,$$

$$\dot{y} = v,$$

$$\dot{u} = x + 2v - \mu' \frac{x+\mu}{[(x+\mu)^2+y^2]^{3/2}} - \mu \frac{x-\mu'}{[(x-\mu')^2+y^2]^{3/2}},$$

$$\dot{v} = y - 2u - \mu' \frac{y}{[(x+\mu)^2+y^2]^{3/2}} - \mu \frac{y}{[(x-\mu')^2+y^2]^{3/2}},$$

with the initial conditions

$$x(0) = x_0 , \qquad u(0) = \dot{x}_0 ,$$

$$y(0) = y_0 , \qquad v(0) = \dot{y}_0 .$$

2. Description of the Method

The Shanks method is a single-step procedure for finding a numerical solution of a first-order ordinary differential equation or system of differential equations in which the derivatives of the dependent variables may be expressed explicitly as functions of the dependent variables and the independent variables.

Consider the system of differential equations

$$\dot{\vec{u}} = \vec{f}(t,\vec{u})$$

where

$$\vec{u}(t) = \begin{pmatrix} u_1(t) \\ \vdots \\ u_p(t) \end{pmatrix}$$

and

$$\vec{f}(t,\vec{u}) = \begin{pmatrix} f_1(t,u_1,\ldots u_p) \\ \vdots \\ f_p(t,u_1,\ldots u_p) \end{pmatrix} .$$

Suppose the value of $\vec{u}(t)$ is known. The value $\vec{u}(t+h)$ is approximated by $\vec{u}(t+h) = u(t) + h \sum_{i=0}^{n} \gamma_i \vec{f}_i(t,h,\vec{u})$, where

$$\vec{f}_0 (t,h,\vec{u}) = \vec{f}(t,\vec{u}) ,$$

$$\vec{f}_i (t,h,\vec{u}) = \vec{f}(t + \alpha_i h,\ u + \alpha_i h \sum_{j=0}^{i-1} \beta_{ij} \vec{f}_j),\ i = 1,2,\ldots n ,$$

and the coefficients α_i ($i = 1,2,\ldots,n$), β_{ij} ($i = 1,2,\ldots,n$; $j = 0,1,\ldots,i-1$), and γ_i ($i = 0,1,\ldots,n$) are known.

A special case of the Shanks formula is the fourth order Runge-Kutta formula:

$$\alpha_1 = \frac{1}{2},\ \alpha_2 = \frac{1}{2},\ \alpha_3 = 1 ,$$

$$\beta_{10} = 1,\ \beta_{20} = 0,\ \beta_{21} = 1,\ \beta_{30} = 0,\ \beta_{31} = 0,\ \beta_{32} = 1 ,$$

$$\gamma_0 = \frac{1}{6} ,\ \gamma_1 = \frac{1}{3},\ \gamma_2 = \frac{1}{3},\ \gamma_3 = \frac{1}{6} .$$

For the system under consideration, t does not appear explicitly; hence, only the products $\alpha\beta_{i\ ij}$, ($i = 1,2,\ldots n$; $j = 0,1,\ldots i - 1$) are used. For useful values of the various combinations of α, β and γ, see Shanks [21].

3. The Computer Program for the Shanks Method

This procedure was programmed for the B-5000 computer in the B-5000 ALGOL language. Double precision arithmetic was used (23 decimal digits).

This program can be used with arbitrary Shanks predictor and corrector formulas for the system of differential equations under consideration.

3.1. Error Estimates and Step Size Control

In this study a predictor and a pseudo-corrector with continuous step size control were used. Suppose $\vec{u}_p$ and $\vec{u}_c$ are the estimates of $\vec{u}$ (t+h)

approximated by the predictor and the corrector, respectively. An error

$$E^2 = |\vec{u}_p - \vec{u}_c|^2$$

is calculated. If $E^2 > E^2_{max}$, where E_{max} is a given error tolerance, the step is rejected; otherwise, it is accepted. In either case, the step size is multiplied by a factor

$$\left(\frac{2E_0^2}{E_0^2 + E^2} \right)^{1/p} ,$$

where p is the order of the Shanks formula being used and E_0 is a specified error tolerance ($E_0 < E_{max}$). This method of step size control is emperical.

3.2. Searching for Improved Initial Conditions

This procedure has the option of improving initial conditions, provided that $y = \dot{x} = 0$ for the initial conditions. Within a specified region, the procedure checks at each step to find whether the orbit has crossed the x-axis. When the x-axis has been crossed, the procedure compares the absolute value of y with a given tolerance ϵ_1. If $|y_c| \geq \epsilon_1$, the previous step is rejected, the step size h is set to $-y/\dot{y}$ and another step is taken. Otherwise, the absolute value of $\dot{x}_{\frac{1}{2}}$ is compared with a specified tolerance ϵ_2. If $|\dot{x}_{\frac{1}{2}}| < \epsilon_2$, the initial value of $\dot{y}$ for that half orbit is taken as the corrected value of the initial $\dot{y}$ and 2 (t+h) is taken as the corrected value of the period. Otherwise, a new initial value

$$\dot{y}_0^{(k+1)} = \frac{\dot{y}^{(k-1)}\dot{x}_{\frac{1}{2}}^{(k)} - \dot{y}_0^{(k)}\dot{x}_{\frac{1}{2}}^{(k-1)}}{\dot{x}_{\frac{1}{2}}^{(k)} - \dot{x}_{\frac{1}{2}}^{(k-1)}}$$

is found and the iteration continues. (In the above formula $\dot{y}_0^{(k)}$ denotes the initial value of $\dot{y}$ for the kth half-orbit and $\dot{x}_{\frac{1}{2}}^{(k)}$ denotes the value of $\dot{x}_{\frac{1}{2}}$ given by the corrector formula at the half-period for the kth half-orbit).

3.3. Closing an Orbit

There are two options for stopping at the end of a full orbit. One is to stop when t is equal to the period. The other is to continue until the x-axis has been crossed within a specified region. When the orbit crosses the x-axis, the absolute value of y_c is compared with ϵ_1. If $|y_c| < \epsilon_1$, the orbit is considered complete; otherwise, the previous step is rejected, the step size is set to $- y/\dot{y}$, and another step is taken.

Using either option, the procedure computes $\dot{x}$ after a final step by linear interpolation, using the formulas

$$h = - y_c/\dot{y}_0 \quad ,$$

$$\dot{x} = \ddot{x}_c h + x_c \quad .$$

3.4. Data Input

Operating Instructions

1. Load the ALGOL program.
2. Load the following information:
   (a) order of the predictor (as a free field integer),
   (b) number of terms in the predictor (as a free field integer),
   (c) order of the corrector (as a free field integer),
   (d) number of terms in the corrector (as a free field integer),
   (e) p (as a free field integer).

3. For each term in the predictor, express the coefficients $\alpha\beta$ as fractions with a common denominator. Load all the numerators for that term, then the denominator (as free field integers).

4. For the predictor, express the coefficients γ as fractions with a common denominator. Load all the numerators, then the denominator, in free field integer form .

5. Repeat (3) - (4) for the corrector.

6. Load the following information as free field double precision real variables:

   (a) Period ,

   (b) Initial x ,

   (c) Initial y ,

   (d) Initial $\dot{x}$,

   (e) Initial $\dot{y}$,

   (f) μ .

7. Load the following information as free field single precision real variables:

   (a) Initial step size,

   (b) Tolerance level E_0 ,

   (c) Maximum tolerance level E_{max} .

8. Load a printing option: 1 if the results of every step are to be printed, 0 otherwise (as a free field integer).

9. Load the following options (as free field integers) :

   (a) 1 if improved initial conditions are to be found, 0 otherwise;

   (b) 1 if the procedure is to terminate when improved initial conditions are to be found; 0 if a complete orbit is to be run .

10. Load the following information as free field single precision real variables:

(a) the tolerance ϵ_1 on the y,

(b) the tolerance ϵ_2 on the $\dot{x}$,

(c) a time test constant for the half-period (must be less the actual period/2),

(d) a time test constant for the period,

(e) the sign of y after crossing the x-axis at the half-period (+1 if positive, -1 if negative),

(f) the sign of y after crossing the x-axis at the period (+1 if positive, -1 if negative).

11. If improved initial conditions are to be found, load the following information (as free field double precision real variables):

    (a) an estimate of $\dot{x}$ at the half-period,

    (b) an estimate of the initial $\dot{y}$ that will give the above $\dot{x}$ at half-orbit.

12. Repeat (6)-(11) for all problems to be run with the same predictor and corrector.

13. Load a zero as free field double precision real variable.

14. Repeat (2)-(13) for all predictor-corrector combinations to be used.

15. Load five zeroes as free field integers.

16. Load an end-Of-deck card.

Note that all free field integers and single precision real variables must be followed by a comma. All double precision real variables must not be followed by a comma, but must be followed by one or more spaces (blanks). As many numbers may be put on a card as is convenient but no card may contain mixtures of single and double precision numbers. In single precision a slash indicates that the rest of the card is to be ignored. In double precision an asterisk indicates the rest of the card is to be ignored.

4. Flow Diagram for the Runge-Kutta-Shanks

Figure 10 gives the flow diagram of the computer program for the Runge-Kutta-Shanks method.

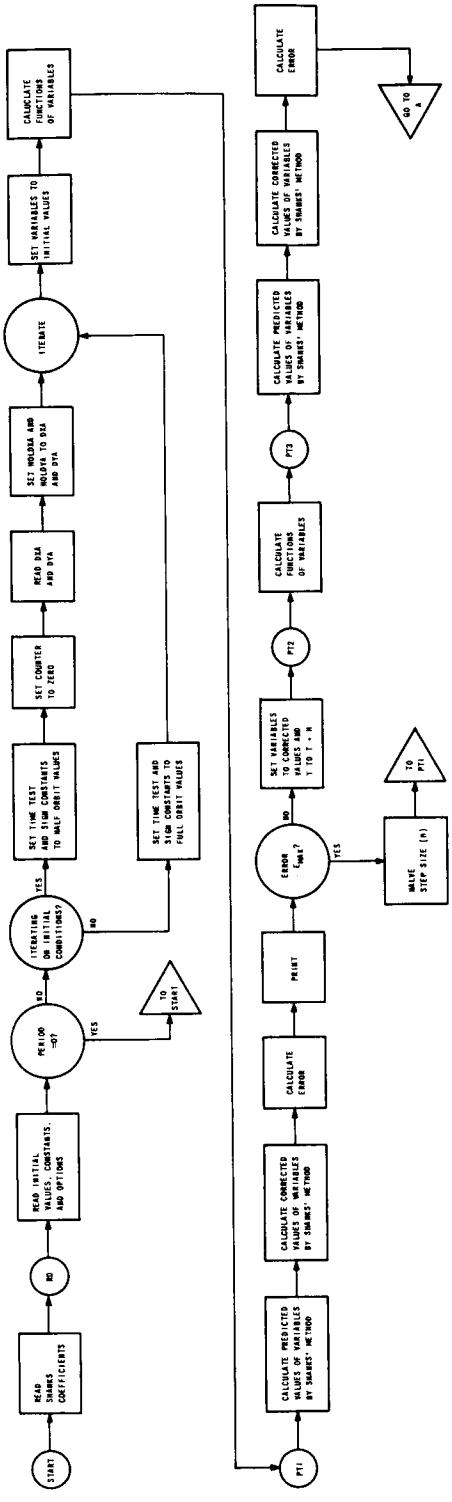

Figure 10. The Flow Diagram for the Runge-Kutta-Shanks Method.

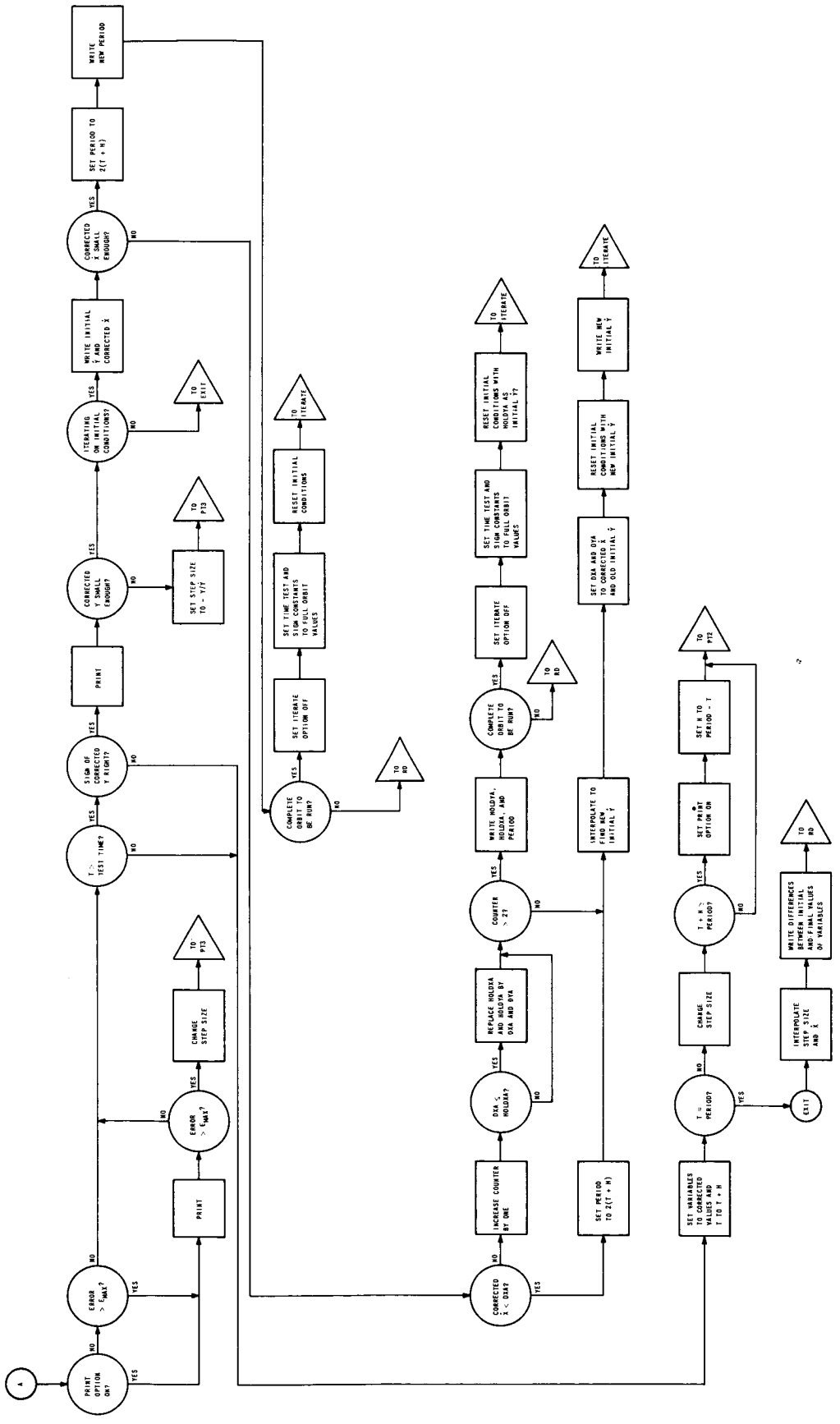

Figure 10 (Continued). The Flow Diagram for the Runge-Kutta-Shanks Method.

5. The Program Listing for the Runge-Kutta-Shanks Method

The following 21 pages list the computer program for the Runge-Kutta-Shanks method. Also listed at the end of the program are the coefficients for the Shanks 7-9 and 8-12 formulas [21].

```
 LABEL   000000000XXXXXX0010000001
BEGIN                                                                    00001000
  COMMENT THIS PROGRAM FINDS A NUMERICAL SOLUTION OF A SYSTEM OF         00002000
          DIFFERENTIAL EQUATIONS BY USE OF THE RUNGE-KUTTA-SHANKS        00003000
            PREDICTOR AND CORRECTOR FORMULAS  ;                          00004000
  LABEL START, STOP ;                                                    00005000
  BOOLEAN FLAG ;                                                         00006000
  INTEGER I, J, K, N, Q, NMAX, ST ;                                      00007000
  INTEGER SGN, SGN1, SGN2, CTR ;                                         00008000
  REAL AH, AL, BH, BL, CH, CL, EH, EL, E0, E2, EMAX, HH, HL, P, TH, TL, 00009000
  TMAXH, TMAXL, TEMPH, TEMPL, TEMP1H, TEMP1L, TEMP2H, TEMP2L, MUH, MUL, 00010000
  MPH, MPL, X1, X2, H1Y, L1Y, TM1, TM2, X, Y ;                           00011000
  REAL EPS1, EPS2, TEST, TEST1, TEST2, DX1H, DX1L, DXAH, DXAL, DYAH,     00012000
  DYAL, HOLDXAH, HOLDXAL, HOLDYAH, HOLDYAL, H0 ;                         00013000
  BOOLEAN ITER, HFORB ;                                                  00014000
  INTEGER ARRAY ORD[0:2], NT[0:2] ;                                      00015000
  ALPHA ARRAY ALEPH1[0:4], ALEPH2[0:4], ALEPH3[0:4], ALEPH4[0:4], ALEPH500016000
  [0:4] ;                                                                00017000
  FILE IN INPUT (2,10) ;                                                 00018000
  FILE OUT OUTPUT 1 (2,15) ;                                             00019000
  FORMAT ID( "RUNGE-KUTTA-SHANKS", 2I3, I6, I3, F8.1 ), LINE( 8I15 ), IC00020000
  ( "H0 ", 5A6, " MU ", 5A6, " MUPRIME ", 5A6/ "E0 ", E20.12, " EMAX ", 00021000
  E20.12, " TIME ", F7.2 ), HEAD( "E ", E20.12, " TIME ", F7.2, " STEP "00022000
  , I6 ), FMT( "TMAX ", 5A6, " X0 ", 5A6, " Y0 ", 5A6/ X34, " U0 ", 5A6,00023000
  " V0 ", 5A6 ), FMT1( "T ", 5A6, " X1 ", 5A6, " Y1 ", 5A6/ X32, " X2 ",00024000
  5A6, " Y2 ", 5A6 ), FMT2( "H ", 5A6, " U1 ", 5A6, " V1 ", 5A6/ X32, " 00025000
U2 ", 5A6, " V2 ", 5A6 ) ;                                               00026000
  FORMAT IIC( "V0 ", 5A6, " U ", 5A6 ) ;                                 00027000
  FORMAT TMX( "TMAX ", 5A6 ) ;                                           00028000
  DEFINE AD = AH, AL#, BD = BH, BL#, CD = CH, CL#, ED =.EH, EL#, HD = HH00029000
  , HL#, TD = TH, TL#, TMAXD = TMAXH, TMAXL#, TEMPD = TEMPH, TEMPL#,     00030000
  TEMP1D = TEMP1H, TEMP1L#, TEMP2D = TEMP2H, TEMP2L#, MUD = MUH, MUL#,   00031000
  MPD = MPH, MPL#, ONE = 1, 0#, TWO = 2, 0#, XD = X1, X2#, YD = H1Y, L1Y00032000
  # ;                                                                    00033000
  DEFINE DX1D = DX1H, DX1L#, DXAD = DXAH, DXAL#, DYAD = DYAH, DYAL#,     00034000
  HOLDXAD = HOLDXAH, HOLDXAL#, HOLDYAD = HOLDYAH, HOLDYAL#, ZERO = 0, 0#00035000
  ;                                                                      00036000
  DEFINE CARD = INPUT#, PRINT = OUTPUT#, LAST = STOP# ;                  00037000
  INTEGER L;                                                             00038000
  REAL ARRAY CON[0:7];                                                   00039000
```

```
DEFINE DDSQRT =                                                          00040000
BEGIN                                                                    00041000
  COMMENT DDSQRT FINDS A SQUARE ROOT IN DOUBLE PRECISION  ;              00042000
  IF X1≠0 THEN                                                           00043000
  BEGIN                                                                  00044000
    CON[0]←0.0000026973988;                                              00045000
    CON[1]←0.000001603883;                                               00046000
    CON[2]←0.0000076294;                                                 00047000
    CON[3]←0.000004536465;                                               00048000
    CON[4]←2.137099@23;                                                  00049000
    CON[5]←1.270727@23;                                                  00050000
    CON[6]←7.555786@22;                                                  00051000
    CON[7]←4.482697@22;                                                  00052000
    X←X1×1.0;                                                            00053000
    L←0;                                                                 00054000
    L←X.[8:2];                                                           00055000
    L.[45:1]←X.[2:1];                                                    00056000
    Y←X;                                                                 00057000
    Y.[3:6]←Y.[2:6];                                                     00058000
    Y←Y×CON[L];                                                          00059000
    Y←(X/Y+Y)×0.5;                                                       00060000
    Y←(X/Y+Y)×0.5;                                                       00061000
    Y←(X/Y+Y)×0.5;                                                       00062000
    Y←(X/Y+Y)×0.5;                                                       00063000
    DOUBLE(X1,X2,Y,0,/,Y,0,-,,0.5,×,Y,0,+,←,H1Y,L1Y);                    00064000
                                                                         00065000
  END ELSE H1Y←L1Y←0;                                                    00066000
                                                                         00067000
END#;                                                                    00068000
INTEGER EVER,NEVER ;                                                     00069000
INTEGER ARRAY NOTUSE1[0:80],NOTUSE2[0:9] ;                               00070000
REAL ARRAY NOTUSE3[0:68],NOTUSE4[0:68] ;                                 00071000
PROCEDURE DREAD(AH,AL) ;                                                 00072000
REAL AH,AL ;                                                             00073000
                                                                         00074000
BEGIN                                                                    00075000
  COMMENT DREAD READS A DOUBLE PRECISION NUMBER  ;                       00076000
  INTEGER K,T1 ;                                                         00077000
  REAL TH,TL ;                                                           00078000
  BOOLEAN DF,MF ;                                                        00079000
```

```
LABEL ER,L1,L2,L3,L4,L5,L6,L7 ;                                          00080000
FORMAT OUT FR(/"DATA CARD ERROR",X105) ;                                 00081000
STREAM PROCEDURE EC(S,D) ;                                               00082000
                                                                         00083000
BEGIN                                                                    00084000
  SI := S ;                                                              00085000
  DI := D ;                                                              00086000
  DS := 8 LIT "0" ;                                                      00087000
  2(40(DS := 7 LIT "0" ;                                                 00088000
  DS := 1 CHR)) ;                                                        00089000
                                                                         00090000
END ;                                                                    00091000
PROCEDURE GN ;                                                           00092000
                                                                         00093000
BEGIN                                                                    00094000
  LABEL L1 ;                                                             00095000
  IF NEVER ≤ 80 THEN GO TO L1 ELSE IF NEVER > 81 THEN                    00096000
  BEGIN                                                                  00097000
    READ (CARD,10,NOTUSE2[*])[LAST] ;                                    00098000
    EC(NOTUSE2,NOTUSE1) ;                                                00099000
    NEVER := 1 ;                                                         00100000
    L1:EVER := NOTUSE1[NEVER] ;                                          00101000
    IF EVER = "*" THEN                                                   00102000
    BEGIN                                                                00103000
      NEVER := 81 ;                                                      00104000
      EVER := " "                                                        00105000
    END ELSE                                                             00106000
  END ELSE EVER := " " ;                                                 00107000
  NEVER := NEVER + 1 ;                                                   00108000
                                                                         00109000
END ;                                                                    00110000
L1:GN ;                                                                  00111000
IF EVER = " " THEN GO TO L1 ;                                            00112000
DF := FALSE ;                                                            00113000
K := 0 ;                                                                 00114000
TH := 0 ;                                                                00115000
TL := 0 ;                                                                00116000
IF EVER = "-" THEN MF := TRUE ELSE                                       00117000
BEGIN                                                                    00118000
  MF := FALSE ;                                                          00119000
```

```
  IF EVER ≠ "+" THEN GO TO L3 ELSE                                    00120000
END ;                                                                 00121000
L2:GN ;                                                               00122000
L3:IF EVER < 10 THEN                                                  00123000
BEGIN                                                                 00124000
  DF := TRUE ;                                                        00125000
  DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL) ;                             00126000
  GO TO L2                                                            00127000
END ;                                                                 00128000
IF EVER = "." THEN                                                    00129000
BEGIN                                                                 00130000
  L4:GN ;                                                             00131000
  IF EVER < 10 THEN                                                   00132000
  BEGIN                                                               00133000
    K := K - 1 ;                                                      00134000
    DF := TRUE ;                                                      00135000
    DOUBLE(,10,TH,TL,×,EVER,0,+,:=,TH,TL) ;                          00136000
    GO TO L4                                                          00137000
  END ELSE                                                            00138000
END ;                                                                 00139000
IF MF THEN DOUBLE(,0,TH,TL,-,:=,TH,TL) ;                             00140000
IF NOT DF THEN                                                        00141000
BEGIN                                                                 00142000
  ER:WRITE(PRINT,FR) ;                                                00143000
  GO TO LAST                                                          00144000
END ;                                                                 00145000
IF EVER = " " THEN GO TO L7 ;                                         00146000
IF EVER ≠ "@" THEN GO TO ER ;                                         00147000
GN ;                                                                  00148000
T1 := 0 ;                                                             00149000
DF := FALSE ;                                                         00150000
IF EVER = "-" THEN MF := TRUE ELSE                                    00151000
BEGIN                                                                 00152000
  MF := FALSE ;                                                       00153000
  IF EVER ≠ "+" THEN GO TO L6 ELSE                                    00154000
END ;                                                                 00155000
L5:GN ;                                                               00156000
L6:IF EVER < 10 THEN                                                  00157000
BEGIN                                                                 00158000
  DF := TRUE ;                                                        00159000
```

```
    T1 := T1 × 10 + EVER ;                                                    00160000
    GO TO L5                                                                  00161000
  END ;                                                                       00162000
  IF NOT DF THEN GO TO ER ;                                                   00163000
  IF EVER ≠ " " THEN GO TO ER ;                                               00164000
  IF MF THEN K := K - T1 ELSE K := K + T1 ;                                   00165000
  L7:IF K < 0 THEN DOUBLE(TH,TL,NOTUSE3[-K],NOTUSE4[-K],/,:=,AH,AL)          00166000
  ELSE DOUBLE(TH,TL,NOTUSE3[K],NOTUSE4[K],×,:=,AH,AL) ;                       00167000
                                                                              00168000
END ;                                                                         00169000
PROCEDURE DPOP(A1,A2,F,ALF,A);                                                00170000
VALUE A1,A2 ;                                                                 00171000
REAL A1,A2 ;                                                                  00172000
FILE F ;                                                                      00173000
ALPHA ARRAY A,ALF[0];                                                         00174000
                                                                              00175000
BEGIN                                                                         00176000
  COMMENT DPOP CONVERTS A DOUBLE PRECISION NUMBER TO ALPHA FORM ;             00177000
  STREAM PROCEDURE PRINT(X,Y,N,F,A,ALF);                                      00178000
  VALUE X,Y,N ;                                                               00179000
                                                                              00180000
  BEGIN                                                                       00181000
    LOCAL V1,V11,V2,V22,V3,V4,ST,RP ;                                         00182000
    DI ←LOC V22;                                                              00183000
    SI ←LOC X ;                                                               00184000
    SI ←SI+1 ;                                                                00185000
    SKIP 3 SB ;                                                               00186000
    13(DS ←3 RESET ;                                                          00187000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                     00188000
    SKIP 1 SB));                                                              00189000
    SI ←LOC Y ;                                                               00190000
    SI ←SI+1 ;                                                                00191000
    SKIP 3 SB ;                                                               00192000
    13(DS ←3 RESET ;                                                          00193000
    3(IF SB THEN DS ←SET ELSE DS ←RESET ;                                     00194000
    SKIP 1 SB));                                                              00195000
    DI ←LOC RP ;                                                              00196000
    DS ←8 LIT"00000000";                                                      00197000
    DI ←LOC ST ;                                                              00198000
    SI ←F ;                                                                   00199000
```

```
DS ←WDS ;                                                          00200000
DI ←ST ;                                                           00201000
SI ←LOC X ;                                                        00202000
SKIP 1 SB ;                                                        00203000
IF SB THEN DS ←8 LIT"      -0."ELSE DS ←8 LIT"       0.";          00204000
3(DS ←8 LIT"00000000");                                            00205000
11(DS ←8 LIT"         ");                                          00206000
SI ←A ;                                                            00207000
V4 ←SI ;                                                           00208000
SI ←LOC V22;                                                       00209000
V3 ←SI ;                                                           00210000
26(DI ←LOC RP ;                                                    00211000
DI ←DI +7 ;                                                        00212000
DS ←CHR ;                                                          00213000
SI ←V4 ;                                                           00214000
DI ←ST ;                                                           00215000
DI ←DI+8 ;                                                         00216000
RP(DS ←24 ADD ;                                                    00217000
DI ←DI -24 ;                                                       00218000
SI ←SI -24);                                                       00219000
SI ←SI +24 ;                                                       00220000
V4 ←SI ;                                                           00221000
SI ←V3 ;                                                           00222000
SI ←SI+1 ;                                                         00223000
V3 ←SI);                                                           00224000
DI ←DI +24 ;                                                       00225000
SI ←LOC N ;                                                        00226000
SKIP 1 SB ;                                                        00227000
IF SB THEN DS ←2 LIT "@-"ELSE DS ←2 LIT "@+";                      00228000
SI ←LOC N ;                                                        00229000
DS ←2 DEC ;                                                        00230000
DI ←DI-1 ;                                                         00231000
2(DS ←RESET);                                                      00232000
DI ←ALF ;                                                          00233000
SI ←ST ;                                                           00234000
SI ←SI +5 ;                                                        00235000
DI ←DI +2 ;                                                        00236000
DS ←CHR ;                                                          00237000
SI ←SI +2 ;                                                        00238000
DS ←CHR ;                                                          00239000
```

```
  DS ←1 LIT".";                                                    00240000
  DS ←3 CHR ;                                                      00241000
  4(DI ←DI +2 ;                                                    00242000
  DS ←6 CHR);                                                      00243000
                                                                   00244000
END ;                                                              00245000
STREAM PROCEDURE SHIFT(A,B,C);                                     00246000
VALUE A,B ;                                                        00247000
                                                                   00248000
BEGIN                                                              00249000
  SI ←LOC A ;                                                      00250000
  DI ←C ;                                                          00251000
  DS ←CHR ;                                                        00252000
  SKIP 3 SB ;                                                      00253000
  DI ←DI +1 ;                                                      00254000
  36(IF SB THEN DS ←SET ELSE DS ←RESET ;                           00255000
  SKIP 1 SB);                                                      00256000
  SKIP 9 DB ;                                                      00257000
  3(IF SB THEN DS ←SET ELSE DS ←RESET ;                            00258000
  SKIP 1 SB);                                                      00259000
  SI ←LOC B ;                                                      00260000
  SKIP 9 SB ;                                                      00261000
  36(IF SB THEN DS ←SET ELSE DS ←RESET ;                           00262000
  SKIP 1 SB);                                                      00263000
                                                                   00264000
END ;                                                              00265000
INTEGER Y,I ;                                                      00266000
REAL T1,T2,HT ,LT;                                                 00267000
ARRAY T[0:1];                                                      00268000
LABEL L1,L2,L3 ;                                                   00269000
IF A1 =0 THEN                                                      00270000
BEGIN                                                              00271000
  T1 ←T2 ←0 ;                                                      00272000
  GO TO L2                                                         00273000
END ;                                                              00274000
HT ←1 ;                                                            00275000
LT ←0 ;                                                            00276000
Y ←0.90309 ×(IF BOOLEAN(A1.[2:1])THEN 13 -A1.[3:6]ELSE 13 +A1.[3:6])00277000
;                                                                  00278000
FOR I ←1 STEP 1 UNTIL ABS(Y)DO DOUBLE(HT,LT,,10,×,←,HT,LT);        00279000
```

```
    IF Y>0 THEN DOUBLE(A1,A2,HT,LT,/,←,T1,T2)ELSE DOUBLE(A1,A2,HT,LT,×,←00280000
    ,T1,T2);                                                          00281000
    Y ←Y-1 ;                                                          00282000
    L1:IF T1.[3:6]<13 THEN                                            00283000
    BEGIN                                                             00284000
      DOUBLE(T1,T2,,10.0,/,←,T1,T2);                                  00285000
      Y ←Y+1                                                          00286000
    END ELSE IF T1.[3:6]>13 THEN                                      00287000
    BEGIN                                                             00288000
      DOUBLE(T1,T2,,10.0,×,←,T1,T2);                                  00289000
      Y ←Y-1                                                          00290000
    END ELSE GO TO L2 ;                                               00291000
    IF T1.[3:6]=14 THEN                                               00292000
    BEGIN                                                             00293000
      SHIFT(T1,T2,T);                                                 00294000
      PRINT(T[0],T[1],Y,F,A,ALF);                                     00295000
      GO TO L3                                                        00296000
    END ELSE GO TO L1 ;                                               00297000
    L2:PRINT(T1,T2,Y,F,A,ALF );                                       00298000
    L3:                                                               00299000
  END ;                                                               00300000
  ALPHA ARRAY A[0:77];                                                00301000
  FILL A[*]WITH "12500000","00000000","00000000","01562500","00000000","00302000
00000000","00195312","50000000","00000000","00024414",,"06250000","000000003030000
00","00003051","75781250","00000000","00000381","46972656","25000000",,"000304000
0000047","68371582","03125000",,"00000005","96046447","75390625","000000000305000
0","74505805","96923828","00000000","09313225","74615479","00000000","0100306000
164153","21826935",,"00000000","00145519",,"15228367","00000000","000181890030700
","89403546","00000000","00002273","73675443",,"00000000",,"00000284","21700308000
09430",,"00000000","00000035","52713679","00000000",,"00000004","44089210"00309000
  ,"00000000","00000000",,"55511151",,"00000000",,"00000000",,"06938894",,"0000310000
000000","00000000","00867362","00000000",,"00000000","00108420",,"0000000000311000
","00000000","00013553",,"00000000",,"00000000","00001694",,"00000000",,"00000312000
00000","0 000212","00000000","00000000","00000026","00000000","00000000"00313000
  ,"00000003";                                                        00314000
                                                                      00315000
  BEGIN                                                               00316000
    INTEGER I ;                                                       00317000
    DOUBLE(,1,:=,NOTUSE3[0],NOTUSE4[0]) ;                             00318000
    FOR I := 1 STEP 1 UNTIL 33 DO DOUBLE(,10,NOTUSE3[I - 1],NOTUSE4[I - 00319000
```

```
  1],x,:=,NOTUSE3[I], NOTUSE4[I]) ;                                    00320000
  FOR I := 1 STEP 1 UNTIL 35 DO DOUBLE(NOTUSE3[33],NOTUSE4[33],NOTUSE300321000
  [I],NOTUSE4[I],x,:=, NOTUSE3[I + 33],NOTUSE4[I + 33]) ;              00322000
  NEVER := 82 ;                                                        00323000
                                                                       00324000
END ;                                                                  00325000
START: TM1 ← TIME(2) ;                                                 00326000
COMMENT THE ORDER AND THE NUMBER OF TERMS FOR THE PREDICTOR AND        00327000
        THE CORRECTOR  ARE READ  ;                                     00328000
READ( INPUT, /, ORD[1], NT[1], ORD[2], NT[2], P ) ;                    00329000
IF P = 0 THEN GO TO STOP ;                                             00330000
WRITE( OUTPUT, ID, ORD[1], NT[1], ORD[2], NT[2], P ) ;                 00331000
IF NT[1] > NT[2] THEN NMAX ← NT[1] ELSE NMAX ← NT[2] ;                 00332000
P ← 1.0/P ;                                                            00333000
                                                                       00334000
BEGIN                                                                  00335000
  LABEL RD, ITERATE, PT1 ;                                             00336000
  REAL ARRAY XH, XL[0:3,0:4], YH, YL[0:4], FH, FL[0:4,0:NMAX] ;        00337000
  REAL ARRAY ZH[0:4], ZL[0:4] ;                                        00338000
  INTEGER ARRAY CF[0:2,0:NMAX,0:NMAX] ;                                00339000
  LIST RESULT( FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH1[I], FOR I ← 0 STEP 100340000
  UNTIL 4 DO ALEPH2[I], FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH3[I], FOR I ←00341000
  0 STEP 1 UNTIL 4 DO ALEPH4[I], FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH5[I]00342000
  ) ;                                                                  00343000
  DEFINE FDIJ = FH[I,J], FL[I,J]#, FD3Q = FH[3,Q], FL[3,Q]#, FD4Q = FH00344000
  [4,Q], FL[4,Q]#, XD0I = XH[0,I], XL[0,I]#, XD11 = XH[1,1], XL[1,1]#,00345000
  XD12 = XH[1,2], XL[1,2]#, XD13 = XH[1,3], XL[1,3]#, XD14 = XH[1,4], 00346000
  XL[1,4]#, XD21 = XH[2,1], XL[2,1]#, XD22 = XH[2,2], XL[2,2]#, XD23 =00347000
  XH[2,3], XL[2,3]#, XD24 = XH[2,4], XL[2,4]#, YDI = YH[I], YL[I]#,    00348000
  YD1 = YH[1], YL[1]#, YD2 = YH[2], YL[2]#, YD3 = YH[3], YL[3]#, YD4 =00349000
  YH[4], YL[4]#, ZD1 = ZH[1], ZL[1]#, ZD4 = ZH[4], ZL[4]# ;            00350000
  DEFINE ERROR =                                                       00351000
  BEGIN                                                                00352000
    COMMENT ERROR COMPUTES AN ERROR FUNCTION BASED ON THE              00353000
                DIFFERENCES BETWEEN THE VALUES OF THE VARIABLES        00354000
    GIVEN                    BY THE PREDICTOR AND BY THE CORRECTOR  ;  00355000
    DOUBLE( XD11, XD21, -, ←, AD ) ;                                   00356000
    DOUBLE( XD12, XD22, -, ←, BD ) ;                                   00357000
    DOUBLE( XD13, XD23, -, ←, CD ) ;                                   00358000
    DOUBLE( XD14, XD24, -, ←, ED ) ;                                   00359000
```

```
    DOUBLE( AD, AD, x, BD, BD, x, +, CD, CD, x, +, ED, ED, x, +, ← ED00360000
    ) ;                                                                 00361000
                                                                        00362000
  END ERROR# ;                                                          00363000
  DEFINE FXCALC =                                                       00364000
  BEGIN                                                                 00365000
    COMMENT FXCALC COMPUTES THE FUNCTIONS FOR SPECIFIED VALUES OF       00366000
                    THE VARIABLES  ;                                    00367000
    DOUBLE( YD1, MUD, +, ←, AD ) ;                                      00368000
    DOUBLE( YD1, MPD, -, ←, BD ) ;                                      00369000
    DOUBLE( YD2, YD2, x, ←, CD ) ;                                      00370000
    DOUBLE( AD, AD, x, CD, +, ←, XD ) ;                                 00371000
    DDSQRT ;                                                            00372000
    DOUBLE( MPD, XD, YD, x, /, ←, TEMP1D ) ;                            00373000
    DOUBLE( BD, BD, x, CD, +, ←, XD ) ;                                 00374000
    DDSQRT ;                                                            00375000
    DOUBLE( MUD, XD, YD, x, /, ←, TEMP2D ) ;                            00376000
    FH[1,Q] ← YH[3] ;                                                   00377000
    FL[1,Q] ← YL[3] ;                                                   00378000
    FL[2,Q] ← YL[4] ;                                                   00379000
    FH[2,Q] ← YH[4] ;                                                   00380000
    DOUBLE( YD4, TWO, x, AD, TEMP1D, x, -, BD, TEMP2D, x, -, YD1, +, ←00381000
    , FD3Q ) ;                                                          00382000
    DOUBLE( ONE, TEMP1D, -, TEMP2D, -, YD2, x, YD3, TWO, x, -, ←, FD4Q00383000
    ) ;                                                                 00384000
                                                                        00385000
  END FXCALC# ;                                                         00386000
  DEFINE RKS =                                                          00387000
  BEGIN                                                                 00388000
    COMMENT RKS USES THE RUNGE-KUTTA-SHANKS FORMULAS TO COMPUTE THE     00389000
                    VALUES OF THE VARIABLES AFTER A STEP  ;             00390000
    N ← NT[K] ;                                                         00391000
    TEMPL ← 0 ;                                                         00392000
    FOR Q ← 1 STEP 1 UNTIL N DO                                         00393000
    BEGIN                                                               00394000
      FOR I ← 1 STEP 1 UNTIL 4 DO YH[I] ← YL[I] ← 0 ;                   00395000
      FOR J ← 0 STEP 1 UNTIL Q-1 DO                                     00396000
      BEGIN                                                             00397000
        TEMPH ← CF[K,Q,J] ;                                             00398000
        FOR I ← 1 STEP 1 UNTIL 4 DO DOUBLE( FDIJ, TEMPD, x, YDI, +, ←,00399000
```

```
      YDI ) ;                                                                  00400000
                                                                               00401000
    END ;                                                                      00402000
    TEMPH ← CF[K,Q,Q] ;                                                        00403000
    FOR I ← 1 STEP 1 UNTIL 4 DO DOUBLE( YDI, HD, ×, TEMPD, /, XDOI,            00404000
    +, ←, YDI ) ;                                                              00405000
    IF Q < N THEN FXCALC ;                                                     00406000
                                                                               00407000
  END ;                                                                        00408000
  FOR I ← 1 STEP 1 UNTIL 4 DO                                                  00409000
  BEGIN                                                                        00410000
    XH[K,I] ← YH[I] ;                                                          00411000
    XL[K,I] ← YL[I] ;                                                          00412000
                                                                               00413000
  END ;                                                                        00414000
                                                                               00415000
END RKS# ;                                                                     00416000
DEFINE PRNT =                                                                  00417000
BEGIN                                                                          00418000
  COMMENT PRNT PRINTS THE VALUES OF THE VARIABLES GIVEN BY THE                 00419000
            PREDICTOR AND BY THE CORRECTOR, THE VALUE OF T AT                  00420000
  THE                BEGINNING OF THE STEP, THE STEP SIZE, THE               00421000
  ERROR, THE                    PROCESSOR TIME ELAPSED, AND THE              00422000
  NUMBER OF STEPS  ;                                                           00423000
  TM2 ← (TIME(2) - TM1)/60 ;                                                   00424000
  WRITE( OUTPUT, HEAD, EH, TM2, ST ) ;                                         00425000
  DPOP( TD, OUTPUT, ALEPH1, A ) ;                                              00426000
  DPOP( XD11, OUTPUT, ALEPH2, A ) ;                                            00427000
  DPOP( XD12, OUTPUT, ALEPH3, A ) ;                                            00428000
  DPOP( XD21, OUTPUT, ALEPH4, A ) ;                                            00429000
  DPOP( XD22, OUTPUT, ALEPH5, A ) ;                                            00430000
  WRITE( OUTPUT, FMT1, RESULT ) ;                                              00431000
  DPOP( HD, OUTPUT, ALEPH1, A ) ;                                              00432000
  DPOP( XD13, OUTPUT, ALEPH2, A ) ;                                            00433000
  DPOP( XD14, OUTPUT, ALEPH3, A ) ;                                            00434000
  DPOP( XD23, OUTPUT, ALEPH4, A ) ;                                            00435000
  DPOP( XD24, OUTPUT, ALEPH5, A ) ;                                            00436000
  WRITE( OUTPUT, FMT2, RESULT ) ;                                              00437000
                                                                               00438000
END PRNT# ;                                                                    00439000
```

```
COMMENT THE COEFFICIENTS FOR THE SHANKS PREDICTOR AND CORRECTOR      00440000
          ARE READ  ;                                                00441000
FOR K ← 1,2 DO                                                       00442000
BEGIN                                                                00443000
  WRITE( OUTPUT, ID, ORD[K], NT[K] ) ;                               00444000
  FOR I ← 1 STEP 1 UNTIL NT[K] DO                                    00445000
  BEGIN                                                              00446000
    READ( INPUT, /, FOR J ← 0 STEP 1 UNTIL I DO CF[K,I,J] ) ;        00447000
    WRITE( OUTPUT, LINE, FOR J ← 0 STEP 1 UNTIL I-1 DO CF[K,I,J]) ;  00448000
    WRITE( OUTPUT[DBL], LINE, CF[K,I,I] ) ;                          00449000
                                                                     00450000
  END ;                                                              00451000
                                                                     00452000
END ;                                                                00453000
RD: COMMENT INITIAL VALUES, OPTIONS, AND CONSTANTS ARE READ AND      00454000
              INITIAL VALUES ARE PRINTED  ;                          00455000
DREAD( TMAXD ) ;                                                     00456000
IF TMAXH = 0 THEN GO TO START ;                                      00457000
FOR I ← 1 STEP 1 UNTIL 4 DO DREAD( YDI ) ;                           00458000
DPOP( TMAXD, OUTPUT, ALEPH1, A ) ;                                   00459000
DPOP( YD1, OUTPUT, ALEPH2, A ) ;                                     00460000
DPOP( YD2, OUTPUT, ALEPH3, A ) ;                                     00461000
DPOP( YD3, OUTPUT, ALEPH4, A ) ;                                     00462000
DPOP( YD4, OUTPUT, ALEPH5, A ) ;                                     00463000
WRITE( OUTPUT, FMT, RESULT ) ;                                       00464000
DREAD( MUD ) ;                                                       00465000
DOUBLE( ONE, MUD, -, ←, MPD ) ;                                      00466000
READ( INPUT, /, HH, E0, EMAX, FLAG ) ;                               00467000
H0 ← HH ;                                                            00468000
HL ← 0 ;                                                             00469000
DPOP( HD, OUTPUT, ALEPH1, A ) ;                                      00470000
DPOP( MUD, OUTPUT, ALEPH2, A ) ;                                     00471000
DPOP( MPD, OUTPUT, ALEPH3, A ) ;                                     00472000
TM2 ← (TIME(2) - TM1)/60 ;                                           00473000
WRITE( OUTPUT, IC, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH1[I], FOR I ← 0  00474000
STEP 1 UNTIL 4 DO ALEPH2[I], FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH3[I],  00475000
E0, EMAX, TM2 ) ;                                                    00476000
E0 ← E0 × E0 ;                                                       00477000
E2 ← E0 + E0 ;                                                       00478000
EMAX ← EMAX × EMAX ;                                                 00479000
```

```
READ( INPUT, /, ITER, HFORB, EPS1, EPS2, TEST1, TEST2, SGN1, SGN2 ) 00480000
;                                                                    00481000
IF ITER THEN                                                         00482000
BEGIN                                                                00483000
  COMMENT THE TIME TEST AND SIGN CONSTANTS ARE SET TO THE VALUES     00484000
            FOR THE HALF ORBIT  ;                                    00485000
  TEST ← TEST1 ;                                                     00486000
  SGN ← SGN1 ;                                                       00487000
  CTR ← 0 ;                                                          00488000
  COMMENT ESTIMATES FOR THE VALUE OF DX AT THE HALF ORBIT AND FOR    00489000
            THE INITIAL VALUE OF DY ARE READ  ;                      00490000
  DREAD( DXAD ) ;                                                    00491000
  DREAD( DYAD ) ;                                                    00492000
  HOLDXAH ← DXAH ;                                                   00493000
  HOLDXAL ← DXAL ;                                                   00494000
  HOLDYAH ← DYAH ;                                                   00495000
  HOLDYAL ← DYAL ;                                                   00496000
                                                                     00497000
END ELSE                                                             00498000
BEGIN                                                                00499000
  COMMENT THE TIME TEST AND SIGN CONSTANTS ARE SET TO THE VALUES     00500000
            FOR THE COMPLETE ORBIT  ;                                00501000
  TEST ← TEST2 ;                                                     00502000
  SGN ← SGN2 ;                                                       00503000
                                                                     00504000
END ;                                                                00505000
FOR I ← 1 STEP 1 UNTIL 4 DO                                          00506000
BEGIN                                                                00507000
  ZH[I] ← YH[I] ;                                                    00508000
  ZL[I] ← YL[I] ;                                                    00509000
                                                                     00510000
END ;                                                                00511000
ITERATE: COMMENT THE ORBIT BEGINS HERE  ;                            00512000
ST ← 1 ;                                                             00513000
TH ← TL ← 0 ;                                                        00514000
Q ← K ← 0 ;                                                          00515000
COMMENT THE VARIABLES ARE SET TO THE INITIAL VALUES  ;               00516000
FOR I ← 1 STEP 1 UNTIL 4 DO                                          00517000
BEGIN                                                                00518000
  XH[0,I] ← YH[I] ;                                                  00519000
```

```
  XL[0,I] ← YL[I] ;                                                    00520000
                                                                       00521000
END ;                                                                  00522000
COMMENT THE FUNCTIONS OF THE VARIABLES AT THE BEGINNING OF THE         00523000
          OF THE STEP ARE CALCULATED  ;                                00524000
FXCALC ;                                                               00525000
PT1: COMMENT THE VALUES OF THE VARIABLES AT THE END OF THE STEP ARE    00526000
               CALCULATED BY THE SHANKS PREDICTOR AND CORRECTOR        00527000
                  FORMULAS  ;                                          00528000
FOR K ← 1,2 DO RKS ;                                                   00529000
ERROR ;                                                                00530000
PRNT ;                                                                 00531000
COMMENT THE ERROR IS COMPARED TO THE MAXIMUM ERROR TOLERANCE  ;        00532000
IF EH > EMAX THEN                                                      00533000
BEGIN                                                                  00534000
  COMMENT THE STEP SIZE IS HALVED AND THE STEP IS REJECTED  ;          00535000
  HH ← HH/2 ;                                                          00536000
  GO TO PT1 ;                                                          00537000
                                                                       00538000
END ;                                                                  00539000
TH ← HH ;                                                              00540000
COMMENT THE VARIABLES ARE SET TO THE VALUES GIVEN BY THE               00541000
          CORRECTOR  ;                                                 00542000
FOR I ← 1 STEP 1 UNTIL 4 DO                                            00543000
BEGIN                                                                  00544000
  YH[I] ← XH[0,I] ← XH[2,I] ;                                          00545000
  YL[I] ← XL[0,I] ← XL[2,I] ;                                          00546000
                                                                       00547000
END ;                                                                  00548000
                                                                       00549000
BEGIN                                                                  00550000
  LABEL PT2, PT3 ;                                                     00551000
  LABEL EXIT ;                                                         00552000
  PT2: Q ← 0 ;                                                         00553000
  ST ← ST + 1 ;                                                        00554000
  COMMENT THE FUNCTIONS OF THE VARIABLES AT THE BEGINNING OF THE       00555000
             OF THE STEP ARE CALCULATED  ;                             00556000
                                                                       00557000
  FXCALC ;                                                             00558000
  PT3: COMMENT THE VALUES OF THE VARIABLES AT THE END OF THE STEP      00559000
  ARE                CALCULATED BY THE SHANKS PREDICTOR AND
```

```
CORRECTOR                              FORMULAS  ;                          00560000
FOR K ← 1,2 DO RKS ;                                                        00561000
ERROR ;                                                                     00562000
IF FLAG OR EH > EMAX THEN                                                   00563000
BEGIN                                                                       00564000
  PRNT ;                                                                    00565000
  COMMENT THE ERROR IS COMPARED TO THE MAXIMUM ERROR TOLERANCE  ;           00566000
  IF EH > EMAX THEN                                                         00567000
  BEGIN                                                                     00568000
    COMMENT THE STEP SIZE IS REDUCED BY THE AUTOMATIC STEP SIZE             00569000
                    CONTROL AND THE STEP IS REJECTED  ;                     00570000
    HH ← HH × (E2/(E0+EH))*P ;                                              00571000
    GO TO PT3 ;                                                             00572000
                                                                            00573000
  END ;                                                                     00574000
                                                                            00575000
END ;                                                                       00576000
IF TH > TEST THEN                                                           00577000
BEGIN                                                                       00578000
  COMMENT THE PROCEDURE CHECKS WHETHER THE X-AXIS HAS BEEN                  00579000
                  CROSSED  ;                                                00580000
  IF SIGN(XH[2,2]) = SGN THEN                                               00581000
  BEGIN                                                                     00582000
    PRNT ;                                                                  00583000
    COMMENT THE VALUE OF Y GIVEN BY THE CORRECTOR IS COMPARED TO            00584000
                    THE TOLERANCE ON Y  ;                                   00585000
    IF ABS(XH[2,2]) ≥ EPS1 THEN                                             00586000
    BEGIN                                                                   00587000
      TEMPH ← -XH[0,2]/XH[0,4] ;                                            00588000
      COMMENT THE STEP SIZE IS REDUCED AND THE STEP IS REJECTED  ;          00589000
      IF TEMPH < HH THEN HH ← TEMPH ELSE HH ← HH/2 ;                        00590000
      GO TO PT3 ;                                                           00591000
                                                                            00592000
    END ;                                                                   00593000
    IF ITER THEN                                                            00594000
    BEGIN                                                                   00595000
      DX1H ← XH[2,3] ;                                                      00596000
      DX1L ← XL[2,3] ;                                                      00597000
      DPDP( ZD4, OUTPUT, ALEPH1, A ) ;                                      00598000
      DPDP( DX1D, OUTPUT, ALEPH2, A ) ;                                     00599000
```

```
WRITE( OUTPUT, IIC, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH1[I],   00600000
FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH2[I] ) ;                    00601000
COMMENT THE VALUE OF DX GIVEN BY THE CORRECTOR IS COMPARED    00602000
TO                    THE TOLERANCE ON DX  ;                 00603000
IF ABS(DX1H) < EPS1 THEN                                     00604000
BEGIN                                                        00605000
  COMMENT THE INITIAL DY IS ACCEPTED AND A NEW PERIOD IS     00606000
                          CALCULATED  ;                      00607000
  DOUBLE( TD, HD, +, TWO, ×, ←, TMAXD ) ;                   00608000
  DPOP( TMAXD, OUTPUT, ALEPH3, A ) ;                         00609000
  WRITE( OUTPUT, TMX, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH3[I]  00610000
  ) ;                                                        00611000
  WRITE( OUTPUT[PAGE] ) ;                                    00612000
  IF HFORB THEN GO TO RD ;                                   00613000
  COMMENT THE TIME TEST AND SIGN CONSTANTS ARE SET TO THE    00614000
  VALUES                    FOR THE COMPLETE ORBIT  ;        00615000
  ITER ← FALSE ;                                             00616000
  TEST ← TEST2 ;                                             00617000
  SGN ← SGN2 ;                                               00618000
  HH ← HO ;                                                  00619000
  HL ← 0 ;                                                   00620000
  FOR I ← 1 STEP 1 UNTIL 4 DO                                00621000
  BEGIN                                                      00622000
    YH[I] ← ZH[I] ;                                          00623000
    YL[I] ← ZL[I] ;                                          00624000
                                                             00625000
  END ;                                                      00626000
  COMMENT THIS ITERATION ON THE HALF ORBIT IS NOW COMPLETE    00627000
  ;                                                          00628000
  GO TO ITERATE ;                                            00629000
                                                             00630000
END ;                                                        00631000
COMMENT THE VALUE OF DX GIVEN BY THE CORRECTOR IS COMPARED   00632000
TO                    THE FINAL VALUE OF DX FROM THE PREVIOUS00633000
 HALF ORBIT  ;                                               00634000
IF DX1H < 0 THEN                                             00635000
BEGIN                                                        00636000
  IF DXAH < 0 THEN DOUBLE( DXAD, DX1D, -, ←, TEMP1D ) ELSE   00637000
  DOUBLE( ZERO, DX1D, -, DXAD, -, ←, TEMP1D ) ;              00638000
                                                             00639000
```

```
END ELSE                                                              00640000
BEGIN                                                                 00641000
  IF DXAH < 0 THEN DOUBLE( DX1D, DXAD, +, ←, TEMP1D) ELSE             00642000
  DOUBLE( DX1D, DXAD, -, ←, TEMP1D) ;                                 00643000
                                                                      00644000
END ;                                                                 00645000
IF TEMP1H ≥ 0 THEN                                                    00646000
BEGIN                                                                 00647000
  CTR ← CTR + 1 ;                                                     00648000
  COMMENT THE FINAL VALUE OF DX FROM THE PREVIOUS HALF ORBIT          00649000
   IS                    COMPARED TO THE LOWEST PREVIOUS HALF         00650000
   ORBIT VALUE OF DX;                                                 00651000
  IF DXAH < 0 THEN                                                    00652000
  BEGIN                                                               00653000
    IF HOLDXAH < 0 THEN DOUBLE( HOLDXAD, DXAD, -, ←, TEMP2D           00654000
    ) ELSE DOUBLE( ZERO, DXAD, -, HOLDXAD, -, ←, TEMP2D ) ;           00655000
                                                                      00656000
  END ELSE                                                            00657000
  BEGIN                                                               00658000
    IF HOLDXAH < 0 THEN DOUBLE( DXAD, HOLDXAD, +, ←, TEMP2D           00659000
    ) ELSE DOUBLE( DXAD, HOLDXAD, -, ←, TEMP2D ) ;                    00660000
                                                                      00661000
  END ;                                                               00662000
  IF TEMP2H < 0 THEN                                                  00663000
  BEGIN                                                               00664000
    HOLDXAH ← DXAH ;                                                  00665000
    HOLDXAL ← DXAL ;                                                  00666000
    HOLDYAH ← DYAH ;                                                  00667000
    HOLDYAL ← DYAL ;                                                  00668000
                                                                      00669000
  END ;                                                               00670000
  IF CTR > 2 THEN                                                     00671000
  BEGIN                                                               00672000
    DPOP( HOLDYAD, OUTPUT, ALEPH1, A ) ;                              00673000
    DPOP( HOLDXAD, OUTPUT, ALEPH2, A ) ;                              00674000
    WRITE( OUTPUT, IIC, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH1[I          00675000
    ], FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH2[I] ) ;                      00676000
    DPOP( TMAXD, OUTPUT, ALEPH3, A ) ;                                00677000
    WRITE( OUTPUT, TMX, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH3[I          00678000
    ] ) ;                                                             00679000
```

```
      WRITE( OUTPUT[PAGE] ) ;                                          00680000
      IF HFORB THEN GO TO RD ;                                         00681000
      COMMENT THE TIME TEST AND SIGN CONSTANTS ARE SET TO THE          00682000
       VALUES                    FOR THE COMPLETE ORBIT  ;             00683000
      ITER ← FALSE ;                                                   00684000
      TEST ← TEST2 ;                                                   00685000
      SGN ← SGN2 ;                                                     00686000
      HH ← HO ;                                                        00687000
      HL ← 0 ;                                                         00688000
      FOR I ← 1 STEP 1 UNTIL 3 DO                                      00689000
      BEGIN                                                            00690000
        YH[I] ← ZH[I] ;                                                00691000
        YL[I] ← ZL[I] ;                                                00692000
                                                                       00693000
      END ;                                                            00694000
      COMMENT THE INITIAL DY IS SET TO THE PREVIOUS VALUE              00695000
      WHICH                          RESULTED IN THE LOWEST VALUE      00696000
       OF DX AT THE HALF ORBIT  ;                                      00697000
      YH[4] ← ZH[4] ← HOLDYAH ;                                        00698000
      YL[4] ← ZL[4] ← HOLDYAL ;                                        00699000
      COMMENT THIS ITERATION ON THE HALF ORBIT IS NOW COMPLETE         00700000
        ;                                                              00701000
      GO TO ITERATE ;                                                  00702000
                                                                       00703000
    END ;                                                              00704000
                                                                       00705000
  END ELSE COMMENT A NEW PERIOD IS CALCULATED  ;                       00706000
  DOUBLE( TD, HD, +, TWO, ×, ←, TMAXD ) ;                              00707000
  COMMENT A NEW INITIAL VALUE OF DY IS COMPUTED BY                     00708000
  INTERPOLATION;                                                       00709000
  DOUBLE( DYAD, DX1D, ×, DXAD, ZD4, ×, -, DX1D, DXAD, -, /, ←,         00710000
  TEMPD ) ;                                                            00711000
  DXAH ← DX1H ;                                                        00712000
  DXAL ← DX1L ;                                                        00713000
  DYAH ← ZH[4] ;                                                       00714000
  DYAL ← ZL[4] ;                                                       00715000
  HH ← HO ;                                                            00716000
  HL ← 0 ;                                                             00717000
  FOR I ← 1 STEP 1 UNTIL 3 DO                                          00718000
  BEGIN                                                                00719000
```

```
        YH[I] ← ZH[I] ;                                                    00720000
        YL[I] ← ZL[I] ;                                                    00721000
                                                                           00722000
      END ;                                                                00723000
      COMMENT THE INITIAL DY IS SET TO THE IMPROVED VALUE  ;               00724000
      YH[4] ← ZH[4] ← TEMPH ;                                              00725000
      YL[4] ← ZL[4] ← TEMPL ;                                              00726000
      DPOP( TEMPD, OUTPUT, ALEPH1, A ) ;                                   00727000
      WRITE( OUTPUT, IIC, FOR I ← 0 STEP 1 UNTIL 4 DO ALEPH1[I] )          00728000
      ;                                                                    00729000
      COMMENT THIS ITERATION ON THE HALF ORBIT IS NOW COMPLETE  ;          00730000
      GO TO ITERATE ;                                                      00731000
                                                                           00732000
    END ELSE GO TO EXIT ;                                                  00733000
                                                                           00734000
  END ;                                                                    00735000
                                                                           00736000
END ;                                                                      00737000
DOUBLE( TD, HD, +, ←, TD ) ;                                               00738000
DOUBLE( TD, TMAXD, -, ←, TEMPD ) ;                                         00739000
COMMENT THE VARIABLES ARE SET TO THE VALUES GIVEN BY THE                   00740000
            CORRECTOR  ;                                                   00741000
FOR I ← 1 STEP 1 UNTIL 4 DO                                                00742000
BEGIN                                                                      00743000
  YH[I] ← XH[0,I] ← XH[2,I] ;                                              00744000
  YL[I] ← XL[0,I] ← XL[2,I] ;                                              00745000
                                                                           00746000
END ;                                                                      00747000
COMMENT THE PROCEDURE CHECKS WHETHER THE PERIOD HAS BEEN                   00748000
            REACHED  ;                                                     00749000
IF TEMPH = 0 THEN GO TO EXIT ;                                             00750000
COMMENT THE STEP SIZE IS CHANGED BY THE AUTOMATIC STEP SIZE                00751000
            CONTROL  ;                                                     00752000
HH ← HH × (E2/(E0+EH))*P ;                                                 00753000
COMMENT THE PROCEDURE CHECKS WHETHER THE PERIOD WILL BE REACHED            00754000
            AFTER THE NEXT STEP  ;                                         00755000
DOUBLE( TD, HD, +, TMAXD, -, ←, TEMPD ) ;                                  00756000
IF TEMPH ≥ 0 THEN                                                          00757000
BEGIN                                                                      00758000
  DOUBLE( TMAXD, TD, -, ←, HD ) ;                                          00759000
```

```
      DOUBLE( TD, HD, +, ←, TMAXD ) ;                                    00760000
      FLAG ← TRUE ;                                                      00761000
                                                                         00762000
    END ;                                                                00763000
    GO TO PT2 ;                                                          00764000
    EXIT:                                                                00765000
    BEGIN                                                                00766000
      FORMAT DIFF( "DX ", E12.3, " DY ", E12.3, " DU ", E12.3, " DV ",   00767000
      E12.3 / X15, " DT ", E12.3, " DU ", E12.3 ) ;                      00768000
      COMMENT A NEW STEP SIZE AND THE VALUE OF DX AFTER THAT STEP ARE    00769000
                    COMPUTED BY LINEAR INTERPOLATION  ;                  00770000
      Q ← K ← 0 ;                                                        00771000
      FXCALC ;                                                           00772000
      DOUBLE( ZERO, YD2, YD4, /, -, ←, HD ) ;                            00773000
      DOUBLE( FD3Q, HD, ×, YD3, +, ←, TEMPD ) ;                          00774000
      COMMENT THE INITIAL AND FINAL VALUES OF THE VARIABLES ARE          00775000
                    COMPARED  ;                                          00776000
      DOUBLE( YD1, ZD1, -, ←, TEMP1D ) ;                                 00777000
      DOUBLE( YD4, ZD4, -, ←, TEMP2D ) ;                                 00778000
      WRITE( OUTPUT, DIFF, TEMP1H, YH[2], YH[3], TEMP2H, HH, TEMPH );    00779000
      WRITE( OUTPUT[PAGE] ) ;                                            00780000
      COMMENT THE ORBIT IS NOW COMPLETE  ;                               00781000
      GO TO RD ;                                                         00782000
                                                                         00783000
    END ;                                                                00784000
                                                                         00785000
  END ;                                                                  00786000
                                                                         00787000
  END ;                                                                  00788000
  STOP:                                                                  00789000
END.                                                                     00790000
 30         END-OF-FILE CARD
 LABEL    00000000INPUT  001      01
7,  9,    / ORDER AND NUMBER OF TERMS FOR THE PREDICTOR
8, 12,    / ORDER AND NUMBER OF TERMS FOR THE CORRECTOR
7,    / ORDER OF THE AUTOMATIC STEP SIZE CONTROL
/ THE FOLLOWING ARE THE ALPHA × BETA COEFFICIENTS FOR THE PREDICTOR
2, 9,
1, 3, 12,
1, 0, 3, 8,
```

```
23, 0, 21, -8, 216,
-4136, 0, -13584, 5264, 13104, 729,
105131, 0, 302016, -107744, -284256, 1701, 151632,
-775229, 0, -2770950, 1735136, 2547216, 81891, 328536, 1375920,
23569, 0, -122304, -20384, 695520, -99873, -466560, 241920, 251888,
/ THE FOLLOWING ARE THE GAMMA COEFFICIENTS FOR THE PREDICTOR
110201, 0, 0, 767936, 635040, -59049, -59049, 635040, 110201, 2140320,
/ THE FOLLOWING ARE THE ALPHA x BETA COEFFICIENTS FOR THE CORRECTOR
1, 9,
1, 3, 24,
1, 0, 3, 16,
29, 0, 33, -12, 500,
33, 0, 0, 4, 125, 972,
-21, 0, 0, 76, 125, -162, 36,
-30, 0, 0, -32, 125, 0, 99, 243,
1175, 0, 0, -3456, -6250, 8424, 242, -27, 324,
293, 0, 0, -852, -1375, 1836, -118, 162, 324, 324,
1303, 0, 0, -4260, -6875, 9990, 1030, 0, 0, 162, 1620,
-8595, 0, 0, 30720, 48750, -66096, 378, -729, -1944, -1296, 3240, 4428,
/ THE FOLLOWING ARE THE GAMMA COEFFICIENTS FOR THE CORRECTOR
41, 0, 0, 0, 0, 216, 272, 27, 27, 36, 180, 41, 840,
```

V. RESULTS AND CONCLUSIONS

A. Results

The results of some computer runs for various Arenstorf restricted three body orbits by the different methods are presented in Tables V, VI, and VII. Three different orbits were chosen for testing the methods. Orbit number 1 (Figure 1) comes close to the earth but not close to the moon. Orbits number 2 and 3 (Figures 2 and 3) come close to the moon but not so close to the earth.

Table V summarizes a series of runs made at relatively low orders (5 to 9) and accuracies ($\approx 10^{-12}$) for all five methods. An attempt was made here to use more or less corresponding orders for each method and choose sets of runs having approximately the same accuracy. The comparisons of the methods are then based on the number of steps and time taken to complete the orbit.

Table VI summarizes a series of runs at orders 12 and 13 for the Runge-Kutta-Fehlberg and Adams methods. The accuracies here were held to around 10^{-14}. Table VII then summarizes a series of runs at orders 14 to 16 for the same methods. Here the accuracies were held to about 10^{-16}.

The errors given in Tables V, VI, and VII are expected to be almost entirely due to the truncation error of the method and not to rounding. None of the methods used involve arithmetic operations prone to producing large rounding error (such as matrix inversions). With 23 decimal place (floating point) arithmetic rounding errors ought not be produced as far over as the 16th or 17th place. Also it can be demonstrated by the simple expedient of reducing

TABLE V

TABLE OF ORBIT RUNS OF LOW ORDER (5-9) AND ACCURACY (10^{-12})

The errors Δx, $\Delta \dot{x}$, and $\Delta \dot{y}$ represent the amount by which the initial conditions failed to be duplicated at the end of the orbit (x-axes crossing) for the x, $\dot{x}$, and $\dot{y}$ coordinates respectively.

Orbit	Number of Steps	Processor time in seconds	Δx in units of 10^{-12}	$\Delta \dot{x}$ in units of 10^{-12}	$\Delta \dot{y}$ in units of 10^{-12}	Predictor Order	Corrector Order	Method
No. 1	523	275	0.2	0.9	0.2	7	8	Runge-Kutta-Shanks
(Fig. 1)	723	250	0.01	0.03	0.1	8		Runge-Kutta-Fehlberg
	3537	543	0.4	2	0.5	9	9	Adams
	7061	1239	1	7	2	5-7		Lie Series (σ=4,α=3)
	3568	2543	8	16	8	8	8	Cowell
No. 2	551	286	0.01	4	2	7	8	Runge-Kutta-Shanks
(Fig. 2)	1606	572	0.02	0.06	1	8		Runge-Kutta-Fehlberg
	4857	688	0.007	0.2	1	9	9	Adams
	25400	4459	0.002	2	0.3	5-7		Lie Series (σ=4,α=3)
	3458	2377	0.04	33	11	8	8	Cowell
No. 3	650	378	0.008	0.4	1	7	8	Runge-Kutta-Shanks
(Fig. 3)	1427	480	0.002	0.01	0.1	8		Runge-Kutta-Fehlberg
	4654	612	0.001	0.05	0.2	9	9	Adams
	23091	3928	0.001	1	0.2	5-7		Lie Series (σ=4,α=3)
	5394	3553	0.01	4	1	8	8	Cowell

TABLE VI

TABLE OF ORBIT RUNS OF INTERMEDIATE ORDER (12-13) AND ACCURACY (10^{-14})

The errors Δx, $\Delta \dot{x}$, and $\Delta \dot{y}$ represent the amount by which the initial conditions failed to be duplicated at the end of the orbit (x-axis crossing) for the x, $\dot{x}$, and $\dot{y}$ coordinates respectively.

Orbit	Number of Steps	Processor time in seconds	Δx in units of 10^{-14}	$\Delta \dot{x}$ in units of 10^{-14}	$\Delta \dot{y}$ in units of 10^{-14}	Predictor Order	Corrector Order	Method
No. 1	370	268	0.03	0.7	1	12		Runge-Kutta-Fehlberg
(Fig. 1)	2435	451	0.8	4	0.8	13	13	Adams
No. 2	623	356	0.01	0.05	0.3	12		Runge-Kutta-Fehlberg
(Fig. 2)	2987	555	0.03	0.8	4	13	13	Adams
No. 3	479	331	0.04	0.1	0.6	12		Runge-Kutta-Fehlberg
(Fig. 3)	2887	535	0.004	0.2	0.6	13	13	Adams

TABLE VII

TABLE OF ORBIT RUNS OF HIGH ORDER (14-16) AND ACCURACY (10^{-16})

The errors Δx, $\Delta \dot{x}$, and $\Delta \dot{y}$ represent the amount by which the initial conditions failed to be duplicated at the end of the orbit (x-axis crossing) for the x, $\dot{x}$, and $\dot{y}$ coordinates respectively.

Orbit	Number of Steps	Processor time in seconds	Δx in units of 10^{-16}	$\Delta \dot{x}$ in units of 10^{-16}	$\Delta \dot{y}$ in units of 10^{-16}	Predictor Order	Corrector Order	Method
No. 1	269	258	0.3	0.07	1	16		Runge-Kutta-Fehlberg
(Fig. 1)	2842	647	1	5	1	14	15	Adams
No. 2	395	362	0.05	0.1	1	16		Runge-Kutta-Fehlberg
(Fig. 2)	3367	718	0.02	0.5	3	14	15	Adams
No. 3	284	262	0.1	0.07	2	16		Runge-Kutta-Fehlberg
(Fig. 3)	3243	721	0.06	3	9	14	15	Adams

the step size or error tolerances, that the final errors can be reduced. If the final errors were dominated by rounding, a reduction in the step size or increase in the number of steps would increase the final error.

B. Conclusions

1. The Lie Series Method

The Lie Series Method used here is basically a low order method (7th order in the position, 5th order in the velocity). It showed up well at lower accuracies (around 10^{-9}) but for the higher accuracies the number of steps required increased rapidly.

Some information was obtained on the best value of σ to use ($\sigma + 1$ is the number of points used in evaluating the integrals in the Lie Series). Since the order of integration is the same for an even value as for the next higher odd value of σ, there is no point in using an odd value. In this study $\sigma = 2$ and $\sigma = 4$ were tried. At low accuracies ($\approx 10^{-9}$) $\sigma = 2$ gave comparable accuracy to $\sigma = 4$ but ran 50% faster. At higher accuracies ($\approx 10^{-12}$) $\sigma = 4$ ran about 50% faster than $\sigma = 2$.

It was seen also that the Lie Series method worked better with the orbits that did not come so close to the singularities. For example, it took three times as long to run orbits 2 and 3 as it did to run orbit 1 (see Figures 1, 2, and 3).

One anomalous feature that exhibited itself in the Lie Series was that the net error in the velocity at the end of one full orbit was generally several orders of magnitude greater than the product of the number of step times the average error per step. This discrepancy was worse for orbits 2 and 3 than for orbit 1.

The method of step size control used here for the Lie Series worked quite well. Several other methods of step size control were tried but did not work as well.

While it is possible to extend the order of the Lie Series by evaluating more terms in the series (increasing α) and/or increasing σ, it is questionable as to whether the additional time spent in evaluating higher derivatives could be compensated for by a larger step size.

2. The Cowell Method

The Cowell method described in this report did not perform as well as had been anticipated. The net cumulative error at the end of a complete orbit was many orders of magnitude greater than would have been expected from the error criteria. That is, the final error was considerably greater than the product of the number of steps times the maximum error per step (the maximum error per step being determined by the mid-range formula). The error accumulation was a gradual process distributed over the entire orbit. There was, however, a noticeable increase in error accumulation in the vicinity of a singularity.

There are several possible explanations for this failure of the Cowell method to give high accuracy:

(1) There could be an error in the computer program.

(2) The step-size control procedure, based on the mid-range formula, may be basically unsound.

(3) The Cowell method as described in this report exhibits some instability. Consider the first of these possibilities; all the usual precautions were taken to insure that the computer program was correct. The accuracy of the program was checked by several different people. Many runs of several versions

of the program were made with extensive monitoring of intermediate results. Although programming of the B5000 in double precision proved to be treacherous, it is hard to believe that the difficulties experienced with this method are due to program errors.

The possibility of an unsound step-size control procedure was investigated. The step-size control procedure was changed to make it dependent on the difference between the predictor and the corrector. The results of computer runs made with this version of the program were no more accurate than results obtained from the version in which the step-size control is based on the mid-range formulas.

There is the possibility that the Cowell method used here is unstable. The Dahlquist criterion for stability cannot be applied directly in this case; therefore, no proof of either stability or instability is known.

3. The Adams Method

The results of the investigation were favorable. Specifically, the following was learned.

Varying the order, or q, during the running of an orbit does not produce better results than when q is held fixed. Changing the order from step to step should help in randomizing the truncation error, but no improvement was found.

There seems to be some advantage in making the order of the corrector one or two greater than that of the predictor. This advantage is slight.

Using the difference between the predictor and the corrector for error control worked quite effectively.

When the iteration option was used, (that is when the corrector was applied repeatedly until the difference between two successive iterations was less than E_{min}) the average number of applications of the corrector was not excessive and averaged about 2 in the runs listed in Tables V, VI, and VII.

The step size alteration worked well. In a multistep method it is difficult to do anything other than double or halve the step size. Doubling requires that a large history be maintained and halving requires an accurate interpolation scheme, but both conditions were met satisfactorily.

4. The Runge-Kutta-Fehlberg Method

The results of the investigation were favorable especially at the higher orders. The method of error control worked well.

Two improvements may be possible here with the error control. First, an error control based on the velocity rather than the position has been developed by Fehlberg [19]. This may increase the accuracy significantly. Second, halving and doubling is not an efficient way of changing the step size for a single step method. A less drastic change or continuously changing step size could eliminate the necessity of ever repeating a step. This procedure worked well in the Runge-Kutta-Shanks method.

5. The Runge-Kutta-Shanks Method

The results of the investigation were favorable. The method of error control and step size alteration worked well.

The step size control was especially effective. For the runs listed in Table V, no step was ever rejected for being too large once a satisfactory starting step size was found.

One disadvantage in the Runge-Kutta-Shanks method is the necessity of making two distinct calculations at each step to determine the error. An

extension of the Runge-Kutta-Shanks method to provide for a fast and easy error estimate (as in the Runge-Kutta-Fehlberg method) could significantly increase the speeds attained here.

6. Comparison of the Various Methods

At the lower order and accuracies, the one step methods of Runge-Kutta-Fehlberg and Runge-Kutta-Shanks were superior to the others examined here. Runge-Kutta-Shanks was somewhat faster than Runge-Kutta-Fehlberg on orbits number 2 and 3 but the Runge-Kutta-Fehlberg was faster on orbit number 1.

At higher orders and accuracies the Runge-Kutta-Fehlberg method continued to be superior. (Runge-Kutta-Shanks methods of higher order are not available at this time). At the highest orders the Runge-Kutta-Fehlberg method was about twice as fast as the Adams method.

7. General Remarks

Apart from the speeds and accuracies exhibited by the various programs, different methods have their own inherent relative advantages and disadvantages.

All multistep methods such as Adams must have a starting procedure to accumulate a history before the method proper can be applied. These separate starting procedures take up space in the computer and can be slow. The multi-step methods must also maintain in memory a long history. Not only must enough points be maintained to calculate the next point but enough history must be maintained to be able to double the step size if called for.

Single step methods can start themselves and need maintain no history of previous values. Also, with the single step methods, any step size change can be made at any time with almost no effort. The multistep methods find step size changes quite difficult and time consuming, especially step size reduction, and any step size change other than halving or doubling especially laborous.

The Runge-Kutta-Shanks and Adams methods have the advantage that they can be applied directly to any non-linear differential equation with no preliminary calculations or analysis. On the other hand, both the Runge-Kutta-Fehlberg and the Lie Series methods require an extensive preliminary analysis of the particular problem. In the Lie Series case, the higher order derivatives must be calculated up to maximum order or degree to be used. In the case of Runge-Kutta-Fehlberg, the recursion relations of the method must be derived. However, much of this work could be done by the computer through suitable symbol manipulation or list processor (LISP) type programs. Only the analysis of the variable transformation in the Runge-Kutta-Fehlberg method that reduces $f(y,t)$ to quadratic form is not yet amenable to computerization.

The Runge-Kutta-Shanks methods have a characteristic disadvantage in that it is not easy to raise the order of the method above that used here. The Adams, Lie Series, and Runge-Kutta-Fehlberg methods can all be carried to any desired order in a straight forward (though often laborous) manner. It is not at all clear how to go about increasing the order of the Runge-Kutta-Shanks methods. Furthermore, the number of function evaluations per step increases rapidly (much faster than the order) as the order is raised.

As a final conclusion then, it is suggested that in the cases where it is possible to find the recursion relations, the Runge-Kutta-Fehlberg method promises to be the most useful research tool for providing high precision results efficiently.

C. Recommendations

The two most promising methods are the Runge-Kutta-Fehlberg and Runge-Kutta-Shanks. The Fehlberg method can be improved by using an error control on the velocity and by a step size change other than halving or doubling.

The Shanks method can be improved by finding a quicker method of error estimation. The present study took the same step twice by two different order Shanks methods and compared the results. If each step could be taken only once and perhaps estimate the error by increasing the number of function evaluations slightly, then the speed of the Shanks method could be increased significantly.

There is no immediate prospect of improving the Adams method as such, but new multistep methods of high order have been introduced by Gragg and Stetter [22] that promise to be significantly advanced in improving multistep methods.

The order of the Lie Series method can be increased by increasing the number of terms in the series, but there is some question as to whether this will increase the speed of the method.

D. Summary

This study was an examination of various methods for integrating non-linear coupled differential equations. The methods used were the following:

A. the single step Lie Series method,

B. the multistep Cowell method,

C. the multistep Adams method,

D. the single step Runge-Kutta-Fehlberg method,

E. the single step Runge-Kutta-Shanks method.

Each of the methods is discussed in detail in this report.

The above methods were applied to the restricted three body problem. In particular, Arenstorf orbits of the restricted three body problem were used. The error in the methods was checked by noting the degree by which the initial conditions failed to be reproduced at the end of one complete period.

Programs for each method were written in double precision floating point arithmetic (23 decimal places) in Extended Algol for the B5000. A series of runs were made on three different Arenstorf orbits at various orders from 7 to 16 and accuracies from 10^{-12} to 10^{-16}. These results are presented in Tables V, VI, and VII.

The conclusions reached were that each of the methods, except that of Cowell, could be considered effective, but the methods of Runge-Kutta-Shanks and Runge-Kutta-Fehlberg were the best. At the highest accuracies and orders, where Runge-Kutta-Shanks formulas are not available, the Runge-Kutta-Fehlberg method was superior.

Respectfully submitted,

I. E. Perlin
Project Director

VI. REFERENCES AND LITERATURE CITED

References on periodic orbits of the restricted three body problem:

1. Arenstorf, R. F., American Journal of Mathematics, LXXXV, (1963), 27.

References on the Lie Series Method of Gröbner:

2. Gröbner, W., "Die Lie-Reihen und ihre Anwendungen," Deutscher Verlag der Wissenschaften, Berlin, 1960.

3. Gröbner, W. and Cap, F., "Perturbation Theory of Celestial Mechanics Using Lie-Series," XIth International Astronautical Congress, Stockholm, 1960.

4. Gröbner, W. and Cap, F., "The Three-Body Problem Earth-Moon-Spaceship," Astronautica Acta, V, (1959), Fasc. 5.

5. Lesky, P., "Lösung des dreidimensionalen Vierkörperproblems: Sonne, Erde, Mond und Raumschiff," Symposium, Provisional International Computation Centre, 1960.

6. Knapp, H., "Ergebnisse einer Untersuchung über den Wert der Lie-Rheihen-Theorie für numerische Rechnungen in der Himmelsmechanik," ZAMM, 42, (1962), T25-27.

7. Knapp, H., "Über eine Verallgemeinerung des Verfahrens der Sukzessiven Approximation zur Lösung Von Differentialgleichungssystemen," Monatshefte für Mathematik, 68, (1964), 33-45.

8. Reutter, F. and Knapp, H., "Über die numerische Auswertung von Gröbners Methode sur Lösung von Anfangswertproblemen gewohnlicher Differential-gleichungssysteme," Forschungsbericht Nr. 1367 des Landes Nordrhein-Westfalen, Westdeutscher Verlag, Köln und Oplader, 1964.

References on the method of Cowell:

9. Cowell, P. H., and Crommelin, A.C.D., "Investigation of the Motion of Halley's Comet from 1759 to 1910." Appendix to Greenwich Observations for 1909. Edinburgh, 1910, 84.

10. Herrick, S., "Step-by-Step Integration of $\dot{x} = f(x,y,z,t)$ without a "corrector," Mathematical Tables and Other Aids to Computation, No. 34, April 1951, 61-67.

11. Currie, J. C., et al, Study of Satellite Orbit Computation Methods and an Error Analysis of the Mathematical Procedures, Technical Report No. 3, prepared for Department of the Air Force Contract No. AF 29(600)-1756, Project No. 1770, Headquarters Air Force Missile Development Center, Air Research and Development Command, Holloman Air Force Base, New Mexico, August 1959.

12. Henrici, P., Discrete Variable Methods in Ordinary Differential Equations, John Wiley and Sons, New York, 1962, 292, 312, and 330.

References on the method of Adams:

13. Bashforth, F., and Adams, J. C., Theories of Capillary Action, Cambridge University Press, 1883.

14. Dahlquist, Germund, Math. Scan., 4, (1956) 33.

15. Dahlquist, Germund, Stockholm Tekniska Hogskolan No. 130, (1959), 1-87.

16. Henrici, Peter, Discrete Variable Methods in Ordinary Differential Equations, John Wiley and Sons, New York, 1962, 191-199.

17. Newberry, A. C. R., Math. Comp. 17, 84, (1963) 452-455.

References on the Runge-Kutta-Fehlberg method:

18. Fehlberg, E., "Runge-Kutta Type Formulas of High-Order Accuracy and Their Application to the Numerical Integration of the Restricted Problem of Three Bodies," Presented at the Colloque International des Techniques de Calcul Analogique et Numerique in Aeronautique in Liege, Belgium, September 1963.

19. Fehlberg, E., "New High-Order Runge-Kutta Formulas with Step Size Control for Systems of First- and Second-Order Differential Equations," Presented by S. Filippi at the meeting of the GAMM in Giessen, Germany, April 1964.

20. Fehlberg, E., "Runge-Kutta Type Formulas of High-Order Accuracy and Their Application to the Numerical Integration of the Restricted Problem of Three Bodies," (private communication).

References on the Runge-Kutta-Shanks method:

21. Shanks, E. B., "Formulas for Obtaining Solutions of Differential Equations by Evaluations of Functions," presented at the summer meeting of the American Mathematical Society in Boulder, Colorado, August 1963, and private communication.

Additional References:

22. Gragg, W. B., and Stetter, H. J., J. ACM, 11 (1964) 188-209.