NASA TECHNICAL NOTE

NASA TN D-6712

OPTIMIZED SOLUTION OF KEPLER'S EQUATION

by John M. Kohout and Lamar Layton

Goddard Space Flight Center
Greenbelt, Md. 20771

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MAY 1972

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TN D-6712		

4. Title and Subtitle

Optimized Solution of Kepler's Equation

5. Report Date: May 1972

6. Performing Organization Code

7. Author(s): John M. Kohout and Lamar Layton

8. Performing Organization Report No.: G-1066

9. Performing Organization Name and Address

Goddard Space Flight Center
Greenbelt, Maryland 20771

10. Work Unit No.

11. Contract or Grant No.

13. Type of Report and Period Covered: Technical Note

12. Sponsoring Agency Name and Address

National Aeronautics and Space Administration
Washington, D. C. 20546

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This document presents a detailed description of KEPLER, an IBM 360 computer program used for the solution of Kepler's equation for eccentric anomaly. KEPLER employs a second-order Newton-Raphson differential correction process, and it is faster than previously developed programs by an order of magnitude.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Kepler's Equation KEPLER (DODS) KEPLR1(DODS)	Unclassified—Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	38	$3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

CONTENTS

Page

OPTIMIZED SOLUTION OF KEPLER'S EQUATION

by
John M. Kohout and Lamar Layton
Goddard Space Flight Center

1.0 INTRODUCTION

1.1 General Description of KEPLER

KEPLER is an IBM 360 computer program used to solve Kepler's equation for eccentric anomaly:

$$E = M + e \sin E .$$

The double precision input to the program consists of mean anomaly M (in radians) and eccentricity of the orbit e. The double precision output from program consists of eccentric anomaly E (in radians), sin E, and cos E.

1.2 KEPLER and KEPLR1

KEPLER has been developed by the authors as a replacement for KEPLR1, a similar program contained in the definitive orbit determination system (DODS) used at Goddard Space Flight Center to determine orbits for NASA's scientific satellites. KEPLR1 was not a hastily coded and formulated program designed for early replacement as soon as DODS became operational. On the contrary, KEPLR1 was developed after a rather extensive research effort that recommended a particular algorithm, the Myles Standish algorithm.* This algorithm was then implemented by Federal Systems Division of IBM under contract to GSFC (see Appendix A). KEPLR1 has been in productive use since May 1968.

KEPLER was developed as a part of an effort to optimize the execution times of frequently called subprograms in DODS. The initial thought was simply to recode KEPLR1 in assembly language coding (ALC), since, like most other programs in DODS, KEPLR1 was coded in FORTRAN, and previous experience with other programs in DODS led the authors to anticipate a 20 to 30 percent speed improvement from a FORTRAN-to-ALC recoding. However, a little analysis of the formulation used in KEPLR1 led the authors to believe that a new analytic approach to the age-old problem was

*Cole, Isabella, and Borchers, Raymond V., "A Comparison of Some Iterative Techniques for the Solution of Kepler's Equation", NASA/GSFC Document X-552-67-421, September 1967.

in order. As a result, they engaged in a research effort of their own* and developed a completely new computer program, KEPLER.

KEPLER not only solves Kepler's equation for eccentric anomaly but also outputs accurate values for the sine and cosine of the eccentric anomaly. This is important since in almost every case for which DODS calls on KEPLR1 to solve Kepler's equation for E, it then uses E as the input argument to the DSIN(X) and DCOS(X) functions. These function calls are unnecessary when KEPLER is used since $\sin E$ and $\cos E$ are part of its output.

1.3 Outline of Remainder of This Document

Section 2 of this document is a description of the relative performance of KEPLER versus that of KEPLR1. Section 3 is a detailed design and performance description of the newly developed program, KEPLER. Section 4 presents the mathematical derivation of the principal formulas used in KEPLER, namely the second-order Newton-Raphson differential correction of eccentric anomaly. Section 5 summarizes the significance of the use of KEPLER and its called module in DODS, DPSC (double precision sine/cosine), and recommends the development of related programs.

Appendix A is a module performance and design description of KEPLR1 prepared for GSFC by IBM. Appendixes B and C list the test programs used to compare KEPLER with KEPLR1. Appendix D lists KEPLR1 (in FORTRAN) and KEPLER and its called program, DPSC (in ALC).

2.0 A COMPARISON OF KEPLER AND KEPLR1

This section deals with accuracy, speed, core storage requirement, and reentrant properties of the KEPLER program. The DODS module KEPLR1 is used as a benchmark for comparison.

2.1 Accuracy of KEPLER

Appendix B lists a test program, TEST 1, which is used to exercise both KEPLR1 and KEPLER over a full range of the input arguments, M and e. The maximum error produced by each program over full ranges of the arguments is obtained by substitution of the solution for eccentric anomaly back into Kepler's equation:

$$\text{KEPLR1 error} = |E - (M + e \sin E)| = 0.50 \times 10^{-15}$$

$$\text{KEPLER error} = |E - (M + e \sin E)| = 0.44 \times 10^{-15} .$$

2.2 Execution Times

Appendix C lists a test program, TEST 2, which times the execution of KEPLER and KEPLR1 on GSFC's IBM 360/95 computer. Full ranges of e and M are used in this test. The average execution

*Kohout, J., and Layton, L., "GSFC Optimized Solution of Kepler's Equation", NASA/GSFC Document X-541-71-229, May 1971.

times reported are based on 400 000 test cases:

Average KEPLR1 execution time = 314 μs*

Average KEPLER execution time = 74 μs .

2.3 Core Requirements

KEPLR 1 requires 820 bytes of core for itself and 620 bytes of core for its called module, IHCLSCN, the standard, release 19 FORTRAN library (FORTLIB) double precision sine/cosine routine. KEPLER requires 440 bytes of core for itself and 2360 bytes of core for its called module, DPSC.

2.4 Reentrancy of KEPLER

Both KEPLER and its called module, DPSC, are reentrant and, therefore, are candidates for the high-speed system link pack area. Neither KEPLR1 nor its called module are reentrant. Therefore, they may not be stored in the high-speed system link pack area.

2.5 Optimal Execution Times for KEPLER

The favorable ratio of execution times reported in Section 2.2 (KEPLR1/KEPLER = 314/74 = 4.24) will be enhanced by factors of 2, 3, and 4, on the IBM 360/95, 360/75, and 360/65, respectively, when KEPLER and DPSC are located in the high-speed system link pack area and KEPLR1 and its called module are located in low-speed core. Under these optimal conditions, the execution time ratios would be 8.48, 12.72, and 16.96, respectively.

Furthermore, when KEPLER and DPSC are located in the system link pack area, several concurrent jobs could be calling on KEPLER, and only one copy of KEPLER would be in core. (The core storage requirement of KEPLER would be charged to system overhead and not to a particular job.)

3.0 KEPLER–MODULE PERFORMANCE AND DESIGN DESCRIPTION

3.1 Language

KEPLER is written in ALC in order to reduce both execution time and core storage. The core storage savings are incidental; the main reason for the use of ALC is to produce a faster executing program. It is estimated that the use of ALC for KEPLER is responsible for about one-third of the improvement in execution time when that program is used in place of KEPLR1.

3.2 Module Size

KEPLER requires 440 bytes of core storage.

*This figure includes the time required to calculate sin E and cos E. Without these calculations, the average execution time for KEPLR1 is 266 μs.

3.3 Purpose of KEPLER

KEPLER solves Kepler's equation for eccentric anomaly when mean anomaly and eccentricity are known. The sine and cosine of eccentric anomaly are output by this module.

3.4 Linkage Information

3.4.1 Calling Sequence

KEPLER is invoked via the following call statement:

CALL KEPLER(MA, ECC, IERR, OUT) ,

where

MA (input) = M, mean anomaly (in radians);
ECC (input) = e, eccentricity;
IERR (output) = error code
= 0, if no error
= 1, if e is negative or greater than 0.99;
OUT (output) = 3 word matrix
OUT(1) = E, eccentric anomaly
OUT(2) = $\sin E$
OUT(3) = $\cos E$.

The qualities M, e, E, $\sin E$, and $\cos E$ are double precision floating point numbers. The error code, IERR, is a full word integer.

3.4.2 Called Modules

KEPLER calls on the reentrant DPSC module. It uses the ALC entry point, SINCOS, which inputs x in FR0 and outputs $\sin x$ in FR0 and $\cos x$ in FR2. (Note: FR0 is floating point register 0 and FR2 is floating point register 2.)

3.4.3 Calling Modules

The following modules in DODS call on KEPLER:

NEGEN0,	DCCON0,
EPTRB0,	CNVRT0,
ELCON0,	UNCAL0.

3.5 Functional Analysis

3.5.1 Module Component 1, Main Program

3.5.1.1 Method

Kepler's equation, $E = M + e \sin E$, is solved for E by use of a second-order Newton-Raphson iterative algorithm. The derivation of this algorithm is discussed in Section 4 of this document. The iterative algorithm is enhanced by four features of KEPLER:

(1) The mean anomaly M is reduced to a value between $-\pi$ and $+\pi$, and the resultant sign is saved. Then, the absolute value of M is used to solve Kepler's equation. After a solution is obtained, E is set equal to $2\pi - E$ if the reduced value of M is negative.

(2) A highly efficient initial estimate algorithm is used to generate E', a starting value for the iterative process. This algorithm is discussed in Section 3.5.2.

(3) The SINCOS entry in the DPSC module is used to calculate simultaneously $\sin E'$ and $\cos E'$.

(4) Sum formulas are used to calculate $\sin (E' + C)$ and $\cos (E' + C)$ whenever C becomes small enough that first- or second-order approximations of $\sin C$ and $\cos C$ are tolerable.

3.5.1.2 Main Program Algorithm

Step 1: Error exit if e is negative or greater than 0.99.

Step 2: Reduce M modulo 2π to range $[-\pi, +\pi]$.
Save sign of M and set $M = |M|$.

Step 3: E' = ESTIMATE (M, e). (See module component 2.)

Step 4: Calculate $\sin E'$ and $\cos E'$ via SINCOS routine.

Step 5: $F = M + e \sin E' - E'$. (linear correction)

Step 6: $D = 1 - e \cos E'$. (first-order derivative)

Step 7: $D' = D + 0.5Fe \sin E'/D$. (second-order derivative)

Step 8: $C = F/D'$. (second-order correction)

Step 9: $E'' = E' + C$; store as E' (enhanced E')

Step 10: If $|C| > 10^{-5}$, return to Step 4.

Step 11: If $|C| < 10^{-8}$, skip to Step 13.

Step 12: $\sin E'' = \sin (E' + C) = (1 - 0.5C^2) \sin E' + C \cos E'$;
$\cos E'' = \cos (E' + C) = (1 - 0.5C^2) \cos E' - C \sin E'$.
(second-order sums formulas)
Replace $\sin E'$ with $\sin E''$, replace $\cos E'$ with $\cos E''$, and return to Step 5.

Step 13: $\sin E'' = \sin (E' + C) = \sin E' + C \cos E'$;
$\cos E'' = \cos (E' + C) = \cos E' - C \sin E'$.
(first-order sums formulas)

Step 14: If sign of reduced M is positive, skip to Step 16.

Step 15: Set $\sin E' = -\sin E'$ and $E' = 2\pi - E'$.

Step 16: Output $E = E'$, $\sin E = \sin E''$, and $\cos E = \cos E''$, with E, $\sin E$, and $\cos E$ accurate to 15 decimal places.

Step 17: Return to calling program.

3.5.1.3 Explanation of Algorithm

Step 1: If e is out of range, no output other than error code is generated.

Step 2: The reduction of M to the range $[-\pi, +\pi]$ and the saving of its sign have several advantages: (1) it increases the precision of the calculation of the linear correction in Step 5 since M and E' will not exceed π; (2) it simplifies the estimation function (Step 3) since M is constrained to the range $[0, \pi]$; (3) it provides a convenient test for E being output in the range $[0, 2\pi]$ (Steps 14-15).

Step 3: The estimation function is treated in detail in Section 3.5.2. The purpose of this function is to provide a sufficiently accurate initial estimate of eccentric anomaly to ensure (1) that the differential correction process defined in Steps 4-9 converges and (2) that this convergence takes place in a minimum number of iterations.

Steps 4-9: This sequence of steps constitutes one second-order Newton-Raphson iteration. This algorithm is considerably more accurate than a first-order differential correction and its use has two basic advantages: (1) it converges in fewer iterations than the first-order correction and (2) the convergence tolerance ϵ used to terminate the second-order differential correction process may be larger than that for the first-order correction because the correction is more accurate. (Other programs use a convergence tolerance of 5×10^{-12}, but KEPLER is able to maintain accuracy with a convergence tolerance of 10^{-8}.)

Step 10: If the absolute value of the correction C, is greater than 10^{-5}, Steps 4-8 are repeated. That is, the lengthy SINCOS routine is reexecuted in Step 4 to provide accurate values for $\sin(E' + C)$ and $\cos(E' + C)$.

Step 11: If the absolute value of C is less than 10^{-8}, sufficient convergence is obtained to guarantee that E is accurate to 15 significant digits. Step 12 is skipped when this condition is met.

Step 12: When the absolute value of C is between 10^{-5} and 10^{-8}, the algorithm iterates, but the lengthy sine/cosine calculation (Step 4) is replaced by the second-order sum formulas in Step 12. The largest truncated term in these sum formulas is $C^3/3!$. This means that when $C < 10^{-5}$, the sum formulas have a relative accuracy of 0.167×10^{-15}, which is slightly more accurate than the original calculation of $\sin E'$ and $\cos E'$ in Step 4.

Step 13: When convergence takes place ($C < 10^{-8}$), $\sin E'$ and $\cos E'$ are updated by the first-order sum formulas in Step 13. The largest truncated term in the first-order sum formula is $C^2/2!$. This means that when $C < 10^{-8}$, the sum formulas have a relative accuracy of 0.5×10^{-16}, which again is more accurate than the original calculation of $\sin E'$ and $\cos E'$.

Step 14: If M is positive after being reduced to the range $[-\pi, +\pi]$, Step 15 is skipped ($\sin E > 0, E < \pi$).

Step 15: If $0 > M > -\pi$, $\sin E'$ is set negative, and E' is set equal to $2\pi - E'$.

Step 16: The quantities E, $\sin E$, and $\cos E$ are sequentially output in a 3 word matrix.

Step 17: The program is concluded.

3.5.2 Module Component 2, Initial Estimate

3.5.2.1 Method

As stated in the preceding section, the main purpose of the initial estimate algorithm is to provide a starting value for the differential correction process defined in Steps 4-9 of the main program. There is an obvious tradeoff between time and accuracy in this initial estimate algorithm. As the initial estimate is made more accurate, the number of times Steps 4-9 of the main program must be executed is reduced. There are, however, constraints on this tradeoff.

When E' is as accurate as one part in 10^5, the time-consuming sine/cosine calculation step in the main program will be executed only once. Therefore, it is desirable to generate E' to this degree of accuracy for most combinations of the input parameters e and M. However, it would be uneconomical to spend too much time trying to achieve a greater overall accuracy since, regardless of the accuracy achieved, the full sine/cosine calculation must be executed at least once in order to further refine E' and to generate the $\sin E$ and $\cos E$ output.

The algorithm used by KEPLER for the initial estimate was selected only after a large number of alternative algorithms were tested and proven to be less efficient.* The name abbreviated Newton-Raphson is given to this algorithm because it represents a first-order Newton-Raphson correction in truncated precision. It possesses two desirable properties: (1) it is executed in a minimal amount of time (37 μs on the IBM 360/75) and (2) for most combinations of M and e, it achieves the desired accuracy of one part in 10^5.

3.5.2.2 Initial Estimate Algorithm (Abbreviated Newton-Raphson)

The abbreviated Newton-Raphson algorithm consists of two steps.

Step A: $$\bar{E} = M + ez\ , \quad 0 < M < \pi\ ,$$

where z is a linear estimate of $\sin M$:

$$z = 0.75\, M,\ M \leqslant \pi/2;$$
$$z = 0.75\, (\pi - M),\ M > \pi/2.$$

Step B: $$E' = \bar{E} + \frac{M + e \sin \bar{E} - \bar{E}}{1 - e \cos \bar{E}}\ ,$$

*Kohout, J., and Layton, L., "GSFC Optimized Solution of Kepler's Equation", NASA/GSFC Document X-541-71-229, May 1971.

where $\sin \bar{E}$ and $\cos \bar{E}$ are calculated from the first two terms of the Maclaurin expansions:

$$\sin x = x - \frac{x^3}{3!}$$

$$\cos x = 1 - \frac{x^2}{2!} .$$

Step B involves several substeps:

(1) Set $x = \bar{E}$ and $S = 1$ (S = SWITCH).

(2) If $x > \pi/2$, set $x = \pi - x$ and $S = 2$.

(3) If $x > \pi/4$, set $x = \pi/2 - x$ and $S = -S$.

(4) Calculate $\sin \bar{E} = x - x^3/6$ and $\cos \bar{E} = 1 - x^2/2$.

(5) If S is negative, exchange $\sin \bar{E}$ and $\cos \bar{E}$ and set $S = -S$.

(6) If $S = 2$, set $\cos \bar{E} = -\cos \bar{E}$.

(7) $E' = \bar{E} + \dfrac{M + e \sin \bar{E} - \bar{E}}{1 - e \cos \bar{E}}$.

3.5.2.3 Relation of Component 2 to Main Program

The initial estimate function (component 2) is linearly coded as Step 3 of the main program (component 1). Component 2 does not have a separate entry point in KEPLER.

3.5.3 Flowcharts

No flowcharts are provided for the main program or the initial estimation function since KEPLER's source code and source code comments directly conform to the logic outlined in Sections 3.5.1.2 and 3.5.2.2.

3.6 Restrictions and Limitations

If the input value of e is negative or greater than 0.99, no output other than error code is generated.

3.7 Storage Tables External to KEPLER

None.

3.8 Input/Output Device Requirements

None.

3.9 Error Conditions and Recovery

The error code, IERR, is examined after execution. If IERR = 1, the input value of e is out of range, and hence no other output can be expected.

3.10 Module Design Technique

KEPLER is reentrant and, therefore, may be loaded in the system link pack area. KEPLER is optimized for fast execution; it is several times faster than existing modules.

3.11 Test Procedures and Results

The speed and accuracy of KEPLER have been verified by the test programs given in Appendixes B and C. The results of these tests are summarized in Sections 2.1 and 2.2.

4.0 SECOND-ORDER NEWTON-RAPHSON METHOD

Deutsch applies the Newton-Raphson method to the problem of solving Kepler's equation (Reference 1, pp. 24–25). He extends the procedure to include second-order effects, but the final equation in the development includes an error in sign, as will be noted later.

If

$$f(E) = E - M - e \sin E ,$$

then,

$$f'(E) = 1 - e \cos E .$$

Let

$$\Delta E = E_1 - E_0 .$$

Then,

$$\Delta E = \frac{-(E_0 - M - e \sin E_0)}{1 - e \cos E_0} + O[(\Delta E)^2] .$$

To obtain an expression valid to terms of order $(\Delta E)^2$, Deutsch proceeds as follows:

$$\begin{aligned} M &= (E_0 + \Delta E) - e \sin (E_0 + \Delta E) \\ &= E_0 + \Delta E - e(\sin E_0 \cos \Delta E + \sin \Delta E \cos E_0) \\ &= E_0 + \Delta E - e\left\{\sin E_0 \left[1 - \frac{(\Delta E)^2}{2}\right] + \Delta E \cos E_0\right\} ; \end{aligned}$$

then,

$$\frac{e \sin E_0}{2} (\Delta E)^2 + (1 - e \cos E_0)\Delta E + (E_0 - M - e \sin E_0) = 0 .$$

Let

$$x = \frac{1}{\Delta E},$$

$$A = E_0 - M - e \sin E_0,$$

$$B = 1 - e \cos E_0,$$

$$C = \frac{e \sin E_0}{2}.$$

We have then

$$Ax^2 + Bx + C = 0$$

$$x = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}.$$

Hence,

$$\Delta E = \frac{2A}{-B \pm \sqrt{B^2 - 4AC}}$$

$$\simeq \frac{2A}{-B \pm (B - 2AC/B)}.$$

Following Deutsch, we adopt the minus sign as the appropriate choice in the denominator;

$$\Delta E = \frac{E_0 - M - e \sin E_0}{-(1 - e \cos E_0) + (1/2)(E_0 - M - e \sin E_0) e \sin E_0 (1 - e \cos E_0)^{-1}},$$

$$\Delta E = \frac{M - E_0 + e \sin E_0}{1 - e \cos E_0 + (1/2)(M - E_0 + e \sin E_0) e \sin E (1 - e \cos E_0)^{-1}},$$

$$\Delta E = \frac{M - E_0 + e \sin E_0}{1 - e[\cos E_0 - (1/2)(M - E_0 + e \sin E_0) \sin E_0 (1 - e \cos E_0)^{-1}]}.$$

(The minus sign within the brackets in the denominator which precedes (1/2) is incorrectly given as a plus sign in Reference 1.)

In general, for functions $f(E)$ for which the relevant derivatives exist, we obtain from a Taylor's series expansion:

$$\Delta E = \frac{f(E_0)}{-f'(E_0) + f(E_0) f''(E_0)/2f'(E_0)},$$

where terms through $(\Delta E)^2$ have been included.

5.0 CONCLUSIONS AND RECOMMENDATIONS

5.1 Effect of KEPLER on DODS Performance

The use of KEPLER in DODS results in a relatively insignificant enhancement of that system because, prior to the use of KEPLER, DODS was spending less than three percent of its time solving Kepler's equation. Therefore, even if KEPLER were one hundred times faster than KEPLR1, the time saving would not be highly significant in DODS operation.

5.2 Effect of the DPSC Module on KEPLER

Of more significance to DODS is the concurrent development of DPSC, with ALC entry points SINCOS, DSINX, and DCOSX and FORTRAN entry points, DPSC, DSIN, and DCOS. KEPLER uses only the SINCOS entry point. The use of this efficient subroutine to simultaneously calculate sin E' and cos E' accounts for about one-third of KEPLER's enhancement of DODS. Use of ALC and the improved mathematical model account for the other two-thirds.

5.3 Effect of DPSC on DODS

The inclusion of the DSIN and DCOS entry points in the DPSC module will enhance DODS considerably more than the inclusion of KEPLER alone. Every sine and cosine calculation in DODS will be executed more efficiently since the DSIN and DCOS entry points in DPSC will override the DSIN and DCOS entry points in the FORTLIB module, IHCLSCN.

5.4 Recommended Use of Simultaneous Sine/Cosine Routine

The DODS formulation contains many situations in which the calculation of both the sine and cosine of a given angle is required. The simultaneous sine/cosine entry points (DPSC and SINCOS) in the DPSC module are now available to DODS programmers who are optimizing DODS modules (such as KEPLER) that call for the calculation of both sin (x) and cos (x). The FORTRAN subprogram call,

```
CALL DPSC (X, SC) ,
```

takes about one-half as much time to execute as the separate function calls to IHCLSCN:

```
SC(1) = DSIN(X)
SC(2) = DCOS(X) .
```

(When DSIN and DCOS entry points are in the DPSC module, the time enhancement is reduced from a factor of 2 to a factor of 1.5.)

5.5 Recommended Use of ALC Entry Points in DPSC

The ALC entry points in the DPSC module enable an ALC program to execute register-to-register sine/cosine functions that bypass the highly indirect FORTRAN convention of passing to the function program *the address of the address* of the argument in general register 1. Besides saving time (the ALC

functions are about seven percent faster), the ALC entry points make it possible for some calling programs to be written in reentrant code, without using the time-consuming GETMAIN macro. This is possible since the ALC functions do not require the storage of an argument list and use only the last eight bytes of the save area, which they can share with the calling program. KEPLER is a good example of a second-order reentrant program sharing its save area with the SINCOS routine.

5.6 Expansion of the DPSC Module

Because of the frequency of calls to mathematical functions in DODS (and other production programs run on GSFC computers), the authors are developing a series of reentrant modules to replace the most frequently called function subprogram modules in FORTLIB. The following function subprograms, called TRIGPACK, are all either completed or nearing completion:

DSQRT(X) ,	DTAN(X) ,	DATAN2(X,Y) ,
DSIN(X) ,	DCOT(X) ,	DASIN(X) ,
DCOS(X) ,	DATAN(X) ,	DACOS(X) .

The single precision counterparts of these double precision function subprograms are also nearing completion.

Besides the standard FORTRAN entry points, these modules all contain corresponding ALC entry points (FORTRAN name with an X suffix—DTANX, for example). The ALC entry points assume that the argument is already in floating point register 0. (For the case of the double argument in the DATAN2X function, the arguments are assumed to be in floating point registers 0 and 2.) The ALC entry points will permit the development of a large number of reentrant second-order subroutines since only the last eight bytes of the save area are used and the storage of an argument list is not required as a prelude to the subroutines execution.

5.7 Element Conversion Module

In conjunction with the development of KEPLER and an optimized reentrant TRIGPACK (Section 5.6), the authors are recoding a DODS module called ELCON0, which contains two inverse subprograms: one for converting position and velocity vectors to osculating Keplerian elements, and the other for performing the reverse of this transformation. This third-order module, written in ALC, calls on KEPLER and the ALC entry points SINCOS, DSQRTX, DSINX, DATANX, DATAN2X, DACOSX, and DTANX in the TRIGPACK modules. It also calls on ALC entry points, VCROSSX, VDOTX, and XDOTX in a newly developed vector package. KEPLER and the SINCOS entry in DPSC are the heart of this newly optimized module.

ACKNOWLEDGMENT

Miss Anne Bomford provided invaluable computer programming support in the development of the programs presented in this document.

Goddard Space Flight Center
National Aeronautics and Space Administration
Greenbelt, Maryland, November 4, 1971
311-80-22-02-51

REFERENCE

1. Deutsch, Ralph, "Orbital Dynamics of Space Vehicles", Englewood Cliffs: Prentice-Hall, Inc., 1963.

Appendix A*

Definitive Orbit Determination System—Model 1 (Module Performance and Design)

5. MODULE NAME: KEPLR1—SOLUTION OF KEPLER'S EQUATION FOR ECCENTRIC ANOMALY.

5.1 LANGUAGE

FORTRAN IV

5.2 MODULE SIZE

The source deck of KEPLR1 consists of 18 executable FORTRAN statements and requires 820 bytes of core storage.

5.3 PURPOSE

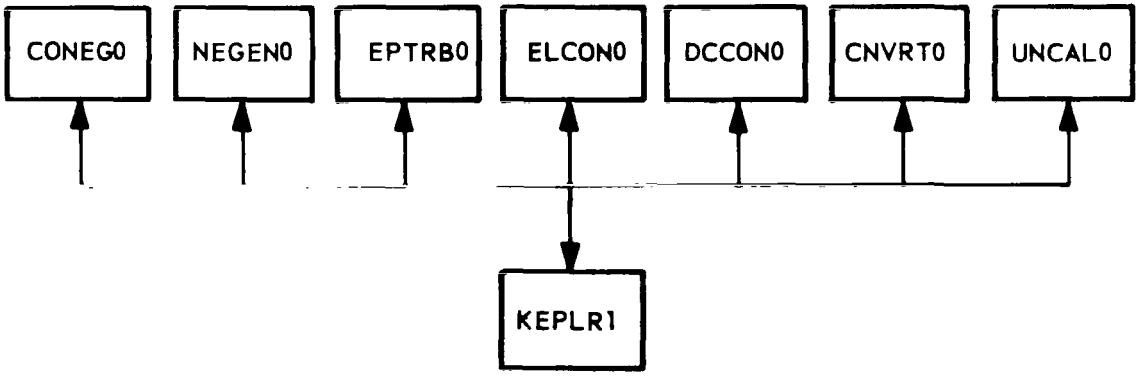
KEPLR1 solves Kepler's equation for eccentric anomaly given mean anomaly and eccentricity by the Myles Standish algorithm.

5.4 INTERFACE INFORMATION

5.4.1 LINKAGE DEFINITION

Linkage to this module requires the following CALL statement: CALL KEPLR1 (MA, ECC, ERRC, E2). See Table 1 for the definition of the calling sequence arguments.

5.4.2 INTERFACE BLOCK DIAGRAM



*Prepared by J. H. Seid, International Business Machines, Inc., under NASA contract NAS5-10022, May 1968. The format employed in this appendix is defined in GSFC X-544-70-324: *Documentation Standards for the Definitive Orbit Determination System—Performance and Design Descriptions* by R. R. Hohl and Lamar Layton (August 1970).

Table 1. Calling Sequence Arguments

Argument Name	Analytic Symbol	I/O	Description	Units	Format*	Limits Min/Max	Dimensions
MA	M	I	Mean anomaly	Radians	LF	-	1
ECC	e	I	Eccentricity	-	LF	0 - 1	1
ERRC		O	Error code =0, convergence <0, no convergence	-	LI	-	1
E2	E	O	Eccentric anomaly	Radians	LF	-	1

*Format Key

LF - Long Form Floating Point

LI - Long Form Integer

5.4.3 INTERFACE BLOCK DIAGRAM NARRATIVE

None

5.4.4 CALLED MODULES

None

5.4.5 CALLING MODULES

ELCON0 - Elements Conversion Package

DCCON0 - DC Control

CNVRT0 - CONVERT Control

UNCAL0 - Unknown Calculation

CONEG0 - CONVERT Normal Equations

NEGEN0 - DC Normal Equations

EPTRB0 - EPHEM Tape Record Builder

5.5 FUNCTIONAL ANALYSIS

5.5.1 MODULE COMPONENT 1: Solve Kepler's equation

5.5.1.1 Method:

Given the mean anomaly, M, and the eccentricity, e, the algorithm for

computing the eccentric anomaly, E, will be:

1. Set error code = 0
   Set limit of number of iterations, MAX = 10
2. Set E = 0
   If M = 0, go to Step 13
   If M $\neq$ 0, go to Step 3
3. $E_0 = M + e \sin M$
   Set number of iterations = 1
4. $F = E_0 - (e \sin E_0) - M$
5. $D = 1.0 - [\, e \cos (E_0 - 0.5F) \,]$
6. $E = E_0 - F/D$
7. If $| E_0 - E | - TOL \leq 0$, go to Step 13; otherwise continue to Step 8.
8. Add 1 to number of iterations
9. If (number of iterations - MAX) $\leq$ 0, continue; otherwise go to Step 12
10. $E_0 = E$
11. Return to Step 4
12. Set error code = 4
13. Modulo E by 2 π
14. Return to calling program

The limit of iterations through Steps 4 to 11 is 10. Thus MAX = 10. If this number is exceeded, the error code is set to 4.

TOL is the tolerance at which the last significant digit of the difference between the previous calculated eccentric anomaly and the present calculated anomaly is allowed. TOL allows an error of $\pm \quad 5 \times 10^{-15}$

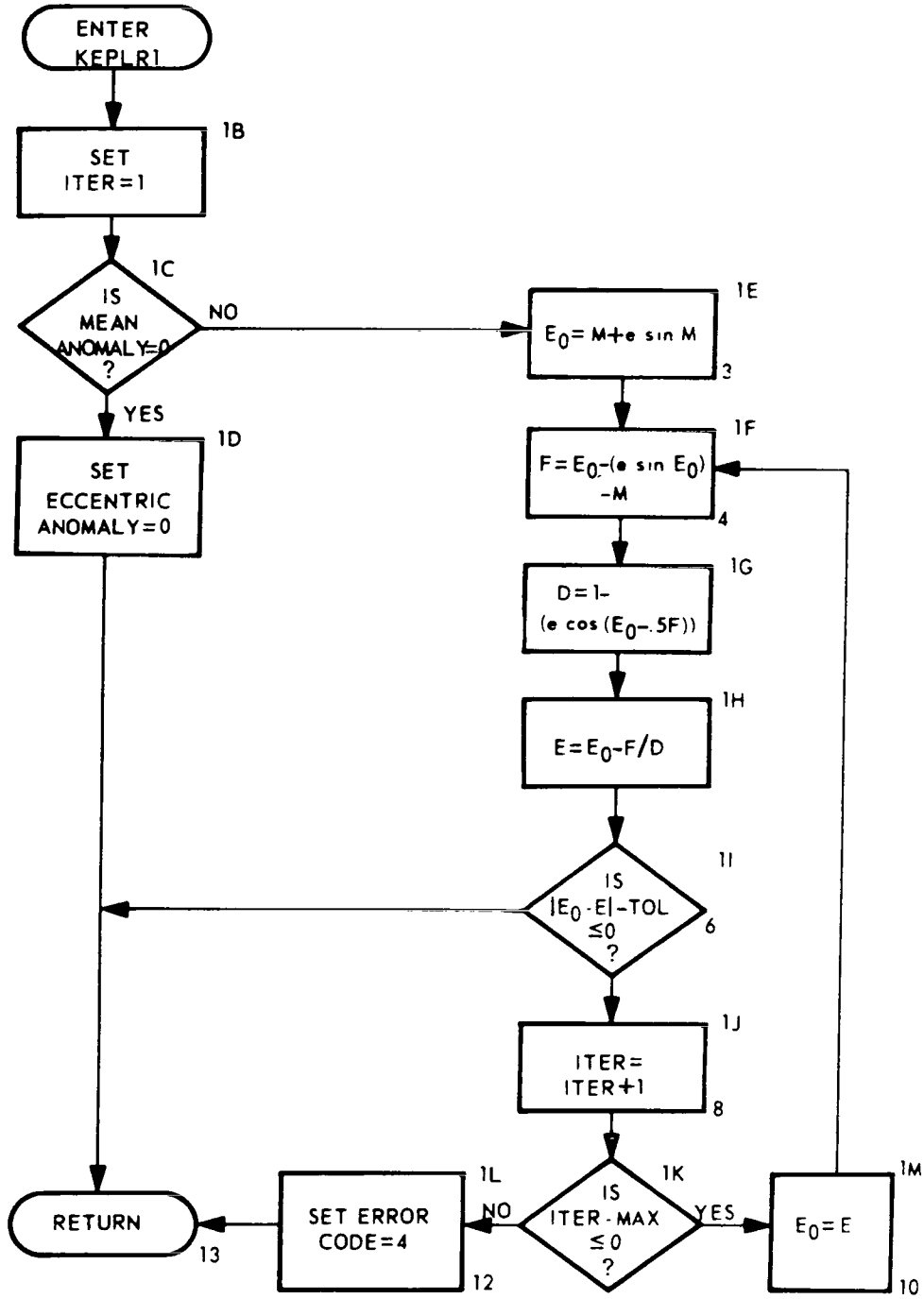
5.5.1.2 Source and Type of Inputs:

The mean anomaly, M, and the eccentricity, e, will be obtained by this module from the calling sequence of the CALL statement which transfers control to this function. The format of M and e will be long form floating-point.

5.5.1.3 Destination and Type of Outputs:

Output of this module will be the eccentric anomaly, E, in long form floating-point format. Also the error code, ERRC, in long integer format will be returned to the calling sequence of the CALL statement which transfers control to this function.

5.5.1.4 Component Level Flowcharts



5.5.1.5 Component Level Flowchart Description

1A - Values for mean anomaly, M, and eccentricity, e, are transferred to the module via the calling sequence. The eccentric anomaly, E, and the error code, ERRC, are returned via the calling sequence.

1B - The counter for the number of iterations is set to 1.

1C - Test to see if mean anomaly is equal to zero.

1D - Sets the eccentric anomaly, E, equal to zero when the mean anomaly, M, is equal to zero and returns to calling module.

1E - Computes the initial eccentric anomaly, E_0.

1F - Computes the expression $E_0 - (e \sin E_0) - M$ using the initial computed value of E.

1G - Computes the expression $1.0 - [e \cos (E_0 - 0.5F)]$ using the initial computed value of E_0 and F computed in 1F.

1H - Computes a more accurate value for eccentric anomaly, $E = E_0 - F/D$.

1I - Test to see if eccentric anomaly has been determined. If the value of the expression $|E_0 - E| - TOL$ is equal to or less than zero, E has been determined. If E has been determined control is returned to the calling module.

1J - Increase number of iterations by 1 when eccentric anomaly has not been determined.

1K - Test to see if number of iterations is less than or equal to the maximum number of iterations allowed.

1L - If the number of iterations exceeds the maximum, set error code equal to 4. Return control to calling module.

1M - If the number of iterations meets the test, make the initial eccentric anomaly, E_0, equal to the computed eccentric anomaly, E, and return to 1F.

5.6 RESTRICTIONS AND LIMITATIONS

None

5.7 STORAGE TABLES EXTERNAL TO MODULE

None

5.8 INPUT/OUTPUT DEVICE REQUIREMENTS

None

5.9 ERROR CONDITIONS AND RECOVERY

Check to see that number of iterations does not exceed the maximum. If so, set error code, ERRC = 4.

5.10 MODULE DESIGN TECHNIQUE

5.10.1 MODULARITY REQUIREMENT

None

5.10.2 EXPANDABILITY REQUIREMENT

None

5.10.3 PARAMETERIZATION

The maximum number of iterations to determine the eccentric anomaly was set equal to 10.

5.10.4 SPECIAL FEATURES

None

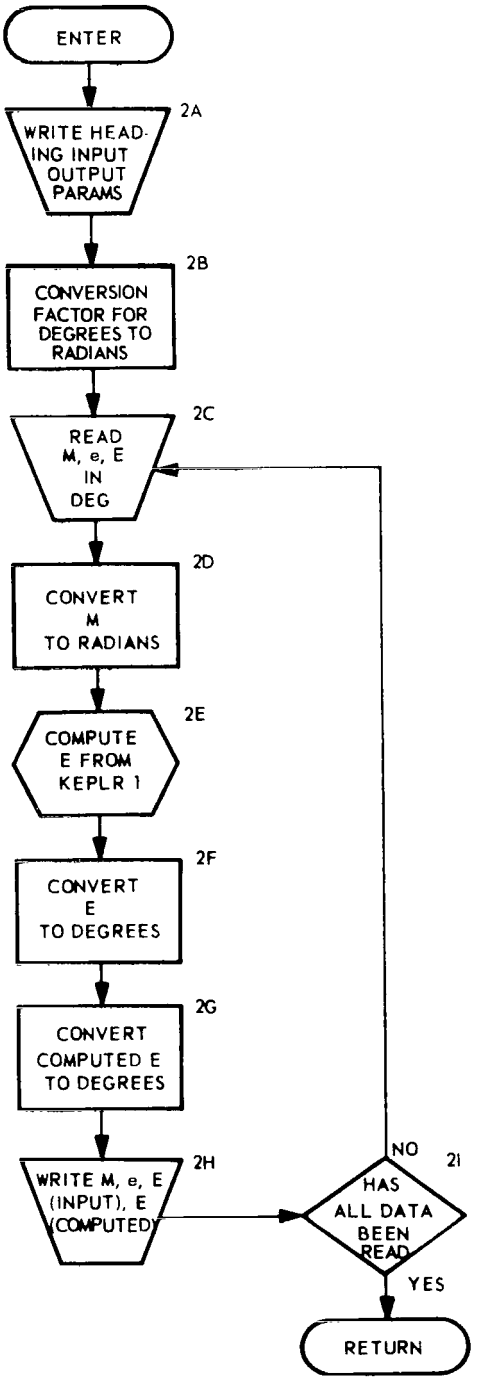
5.11 TESTING PROCEDURES AND RESULTS

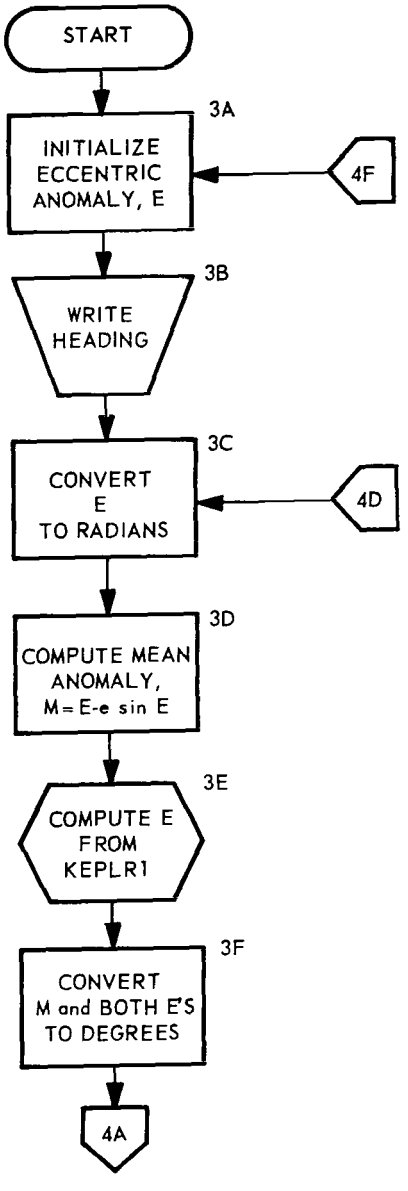
5.11.1 UNIT TEST DRIVER DESIGN AND IMPLEMENTATION

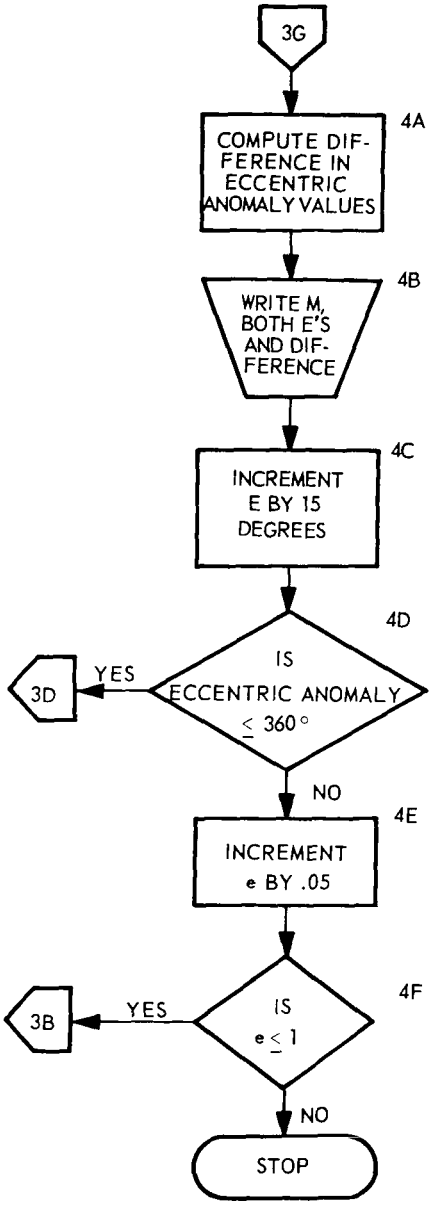
Test 1 - Input data-eccentricity and mean anomaly are read from data cards, subroutine KEPLR1 is executed and the results are printed.

Test 2 - The eccentricity is varied from 0 to 1 by increments of .05 for all values of eccentric anomaly from 0 to 360 degrees, incremented by 15 degrees in order to compute values of mean anomaly. Subroutine KEPLR1 is called for all values of mean anomaly and corresponding eccentricity, and the results are printed out.

5.11.1.1 Unit Test Driver Flowchart




START
3A
INITIALIZE ECCENTRIC ANOMALY, E
4F
3B
WRITE HEADING
3C
CONVERT E TO RADIANS
4D
3D
COMPUTE MEAN ANOMALY, M=E-e sin E
3E
COMPUTE E FROM KEPLR1
3F
CONVERT M and BOTH E'S TO DEGREES
4A


3G
4A
COMPUTE DIF-FERENCE IN ECCENTRIC ANOMALY VALUES
4B
WRITE M, BOTH E'S AND DIF-FERENCE
4C
INCREMENT E BY 15 DEGREES
4D
IS ECCENTRIC ANOMALY ≤ 360°
YES
3D
NO
4E
INCREMENT e BY .05
4F
IS e ≤ 1
YES
3B
NO
STOP

5.11.2 BENCHMARK TESTING

None

5.11.3 TEST RESULTS AND ACCURACY

This item is not covered at the module level; it is included in the system evaluation document.*

5.12 GLOSSARY

A glossary of internal symbols associated with quantities having analytic significance is given in Table 2.

5.13 REFERENCES

Memorandum from I. Cole to IBM; 1 August 1967

Table 2. Internal Symbols

Program Symbol	Analytic Symbol	Description of Term	Units	Format	Limits Min/Max
E1	E_0	Eccentricity anomaly Compare value	Radians	LF	-
ITER	ITER	Number of iterations Completed	-	LI	-
MAX	MAX	Limit of number of iterations	-	LI	-
TOL	TOL	Tolerance for convergence	-	LF	-

*Working document in use at GSFC.

Appendix B

Test 1 and Results

```
C                  TEST 1
C       TEST PROGRAM TO TIME TWO KEPLER PROGRAMS
C                                                    PROGRAMMER  ANNE BOMFORD
      REAL*8 M,E,EA(3),DE
  500 FORMAT('1',//,49X,'EXECUTION TIMES FOR KEPLER AND KEPLR1',//)
      WRITE(6,500)
      E=.1D-03
      ITIME =2
      CALL INTIMO(1)
      DO 3 I=1,20
      M=.1D-02
      DO 2 J=1,200
      DO 1 K=1,100
    1 CALL KEPLER(M,E,IRR,EA)
    2 M=M+6.28D-02
    3 E=E+.49D-01
      CALL INTIMO(ITIME)
      ITIME=ITIME/400000
  506 FORMAT(//,30X,'AVERAGE EXECUTION TIME FOR KEPLER ON 360/95',1X,I3,
     11X,'MICROSECONDS')
      WRITE(6,506) ITIME
      E=.1D-03
      ITIME=2
      CALL INTIMO(1)
      DO 6 I=1,20
      M=.1D-02
      DO 5 J=1,200
      DO 4 K=1,100
    4 CALL KEPLR1(M,E,IRR,E2)
    5 M=M+6.28D-02
    6 E=E+.49D-01
      CALL INTIMO(ITIME)
      ITIME=ITIME/400000
  507 FORMAT(//,30X,'AVERAGE EXECUTION TIME FOR KEPLR1 ON 360/95',1X,I3,
     11X,'MICROSECONDS')
  508 WRITE(6,507) ITIME
      E=.1D-03
      ITIME=2
      CALL INTIMO(1)
      DO 10 I=1,20
      M=.1D-02
      DO 11 J=1,200
      DO 12 K=1,100
      CALL KEPLR1(M,E,IRR,E2)
      EA(2)=DSIN(E2)
   12 EA(3)=DCOS(E2)
   11 M=M+6.28D-02
   10 E=E+.49D-01
      CALL INTIMO(ITIME)
      ITIME=ITIME/400000
  607 FORMAT(//,30X,'AVERAGE EXECUTION TIME FOR KEPLR1 ON 360/95',1X,I3,
     11X,'MICROSECONDS(WITH SINCOS)')
      WRITE(6,607) ITIME
      RETURN
      END
```

```
                          EXECUTION TIMES FOR KEPLER AND KEPLR1
                AVERAGE EXECUTION TIME FOR KEPLER ON 360/95  74 MICROSECONDS
  (WITH SINCOS)AVERAGE EXECUTION TIME FOR KEPLR1 ON 360/95 314 MICROSECONDS
(WITHOUT SINCOS)AVERAGE EXECUTION TIME FOR KEPLR1 ON 360/95 266 MICROSECONDS
```

Appendix C

Test 2 and Results

```
C                  TEST 2
C       TEST PROGRAM TO TEST VALIDITY OF TWO KEPLER PROGRAMS
C                                                    PROGRAMMER  ANNE BOMFORD
      REAL*8 M,E,EA(3),DE
      REAL*8 A,B,ERR,MAXERR
      REAL*8 E2
  800 FORMAT('1',//,49X,'VALIDITY TEST FOR KEPLER AND KEPLR1',//)
      WRITE(6,800)
      DO 501 K=1,2
      MAXERR=0.0D00
      E=.1D-03
      DO 802 I=1,100
      M=.1D-02
      DO 803 J=1,1000
      IF(K.EQ.2) GO TO 505
      CALL KEPLER(M,E,IRR,EA)
      ERR=DABS(EA(1)-M-E*EA(2))
      IF(ERR.LE.MAXERR) GO TO 803
      MAXERR=ERR
      A=M
      B=E
      GO TO 803
  505 CALL KEPLR1(M,E,IRR,E2)
      EA(1)=E2
      EA(2)=DSIN(E2)
      ERR=DABS(EA(1)-M-E*EA(2))
      IF(ERR.LE.MAXERR) GO TO 803
      MAXERR=ERR
      A=M
      B=E
  803 M=M+.628D-02
  802 E=E+.980D-02
      IF(K.EQ.2) GO TO 808
  806 FORMAT(//,20X,'MAXIMUM ERROR FOR KEPLER=',D10.3,'  M=',E10.3,'  E=
     1',E10.3)
      WRITE(6,806) MAXERR,A,B
      GO TO 501
  807 FORMAT(//,20X,'MAXIMUM ERROR FOR KEPLR1=',D10.3,'  M=',E10.3,'  E=
     1',E10.3)
  808 WRITE(6,807) MAXERR,A,B
  501 CONTINUE
      RETURN
      END
```

```
**********************************************************************
**********************************************************************
                      VALIDITY TEST FOR KEPLER AND KEPLR1
          MAXIMUM ERROR FOR KEPLER= 0.444D-15 M= 0.520D 01 E= 0.951D 00
          MAXIMUM ERROR FOR KEPLR1= 0.500D-15 M= 0.264D 01 E= 0.941D 00
**********************************************************************
**********************************************************************
```

Appendix D

KEPLR1 and KEPLER Source Code

```
      SUBROUTINE KEPLR1(MA,ECC,ERRC,E2)
C
C          THIS SUBROUTINE IS USED TO SOLVE KEPLER'S EQUATION
C     FOR ECCENTRIC ANOMALY GIVEN MEAN ANOMALY AND ECCENTRICITY
C     BY THE MYLES STANDISH ALGORITHM.
C     IN THE CALLING SEQUENCE MA IS THE MEAN ANOMALY, ECC IS
C     THE ECCENTRICITY, ERRC IS THE ERROR CODE FOR NUMBER OF
C     ITERATIONS AND ER IS THE ECCENTRIC ANOMALY.
      INTEGER  ERRC,MAX,ITER,EC
      DOUBLE PRECISION MA,ECC,E1,E2,F,D,ABS
      REAL*8 TOL/.05D-10/,PI2/6.283185307179586/
      MAX=10
      ERRC=0
      E2=0.0D0
      IF(MA) 3,13,3
3     E1=MA+ECC*DSIN(MA)
      ITER=1
4     F=E1-(ECC*DSIN(E1))-MA
      D=1.0D0-(ECC*DCOS(E1-0.5D0*F))
      E2=E1-F/D
C     TEST FOR CONVERGENCE
6     IF(DABS(E1-E2)-TOL) 13,13,8
8     ITER=ITER + 1
C     TEST FOR NUMBER OF ITERATIONS
      IF(ITER-MAX)  10,10,12
10    E1=E2
      GO TO 4
12    ERRC=4
   13 E2=DMOD(E2,PI2)
      IF(E2.LT.0) E2=PI2+E2
      RETURN
      END

KEPLER   CSECT
         USING *,15
         STM   14,4,12(13)     SAVE REGISTERS
         LM    1,4,0(1)        ARGUMENT ADDRESSES M,E,IERR,EA
*     STEP ONE
         LD    2,0(2)          E
         LTDR  2,2             TEST E  FOR MINUS
         BM    ERROR
         CE    2,=E'.99'       TEST E  FOR SIZE
         BL    *+12            OK IF LESS THAN .99
ERROR    MVI   3(3),1             * SET ERROR CODE =1
         B     EXIT               * AND EXIT
         MVI   3(3),0          ZERO ERROR CODE (NO ERROR)
```

```
*       STEP TWO
            LD      4,0(1)          M    HOLD FOR SIGN
            LPDR    0,4             M ABSOLUTE
            CD      0,TWOPI         OK IF LESS THAN 2PI
            BL      *+20               *
            LDR     6,0                * OTHERWISE
            DD      6,TWOPI            * REDUCE M
            AW      6,ZERO14           * TO MODULO 2PI
            MD      6,TWOPI            *
            SDR     0,6                *
            CD      0,PI            *
            BL      *+12            * IF X GREATER THAN PI
            LCDR    4,4             * AND REVERSE SIGN
            LCDR    0,0             * AND SET X = 2PI-X
            AD      0,TWOPI         *
            STE     4,52(13)        SIGN OF M SAVED
            STD     0,56(13)        M ABSOLUTE BETWEEN 0 AND PI
*       STEP THREE-A
            CE      0,=E'1.57'  *
            BL      *+10        *
            LCDR    0,0         *   QUICK AND DIRTY SIN(M)
            AD      0,PI        *
            HDR     0,0         *   =(3/4)X     X= M        M BETWEEN 0-PI/2
            HDR     4,0         *           OR X= PI-M      M BETWEEN PI/2-PI
            ADR     0,4         *
            MER     0,2             E*SIN(E)
            AD      0,56(13)    EA = M + E*SIN(M), 1ST ESTIMATE
            STD     0,0(4)      HOLD AS EA (POSITIVE)
*       STEP THREE-B
            LA      0,1             S=1 (ASSUMES EA IN QUAD1)
            CE      0,=E'1.5707963'
            BL      QUAD1       OK 1ST QUADRANT
            LA      0,2             OTHERWISE SET S=2 *
            LCDR    0,0             AND               *
            AD      0,PI            EA= PI-EA         * EA LESS THAN PI/2
QUAD1       CE      0,=E'.7853981'
            BL      OCT1        OK 1ST OCTANT
            LCR     0,0               * OTHERWISE SET S - *
            LCDR    0,0               * AND                *
            AE      0,=E'1.5707963'   * EA = PI/2 -EA      * EA LESS PI/4
OCT1        LCDR    2,0             -X
            HDR     4,0              X/2
            MER     2,4             -X2/2
            MER     4,2             -X3/4
            ME      4,=E'.6666667'  -X3/6
            AD      2,ONEX          1-X2/2  = COS(EA)
            ADR     0,4             X-X3/6  = SIN(EA)
            LTR     0,0         TEST S FOR + OR -
            BP      *+12        OK 1ST OCTANT
            LCR     0,0             * OTHERWISE SET S =+1 OR +2
            LDR     4,2             * AND
            LDR     2,0             * EXCHANGE
            LDR     0,4             * SIN(EA) AND COS(EA)
            BCT     0,*+6       * -COS(EA)    - IN QUAD 1
            LCDR    2,2         * -COS(EA)    + IN QUAD 2
```

```
          ME     0,0(2)              E*SIN(EA)
          ME     2,0(2)             -E*COS(EA)
          AD     2,ONEX            1-E*COS(EA)        =D     1ST ORDER
          AD     0,56(13)          M+E*SIN(EA)
          SD     0,0(4)            M+E*SIN(EA) -EA    =F
          DER    0,2               F/D                =C    1ST ORDER
          AD     0,0(4)            EA+C
          STD    0,0(4)            STORE AS EA ESTIMATE (POSITIVE)
*      STEP FOUR
          L      15,=V(SINCOS)
          BALR   14,15
          USING *,14
*      STEP FIVE
ITERATE   STD    0,8(4)          SIN(EA)
          MD     0,0(2)          E*SIN(EA)
          HDR    4,0                                  .5*E*SIN(EA) HOLD
          AD     0,56(13)       M+E*SIN(EA)
          SD     0,0(4)         M+E*SIN(EA) -EA   =F
*      STEP SIX
          LCDR   6,2               -COS(EA)
          MD     6,0(2)         -E*COS(EA)
          AD     6,ONEX        1-E*COS(EA)         =D  1ST ORDER
*      STEP SEVEN
          MER    4,0              F*.5*E*SIN(EA)
          DER    4,6              F*.5*E*SIN(EA)/D
          ADR    6,4          D +F*.5*E*SIN(EA)/D     = D 2ND ORDER
*      STEP EIGHT
          DDR    0,6              F/D                 = C 2ND ORDER
          LDR    4,0              SAVE C
*      STEP NINE
          AD     0,0(4)           EA+C
          STD    0,0(4)           SAVE AS ENHANCED EA
*      STEP TEN
          LPER   6,4              C ABSOLUTE
          CE     6,=E'1.E-5'        * RETURN FOR FULL SINCOS AND ITERATE
          BCR    2,15               * IF C GREATER THAN .00001
*      STEP ELEVEN
          CE     6,=E'1.E-8'        * CONVERGENCE
          BL     OUT                * WHERE C LESS THAN .00000001
*      STEP TWELVE
          LCDR   0,4              -C                2ND ORDER SUMS FORMULAE  *
          HDR    6,4              C/2               C BETWEEN 10**-5,10**-8  *
          MDR    0,6               -C*C/2                                    *
          AD     0,ONEX           1-C*C/2 = COS(C)                          *
          LDR    6,4              C = SIN(C)                                *
          MDR    6,2              SIN(C)*COS(EA)                            *
          MDR    2,0              COS(C)*COS(EA)                            *
          MD     4,8(4)           SIN(C)*SIN(EA)                            *
          MD     0,8(4)           COS(C)*SIN(EA)                            *
          ADR    0,6              SIN(EA+C)=SIN(C)*COS(EA)+COS(C)*SIN(EA)   *
          SDR    2,4              COS(EA+C)=COS(C)*COS(EA)-SIN(C)*SIN(EA)   *
          BR     14               ITERATE                                   *
*      STEP THIRTEEN
OUT       LDR    0,4        C               1ST ORDER SUMS FORMULAE  *
          MDR    0,2        C*COS(EA)       C LESS THAN 10**-8       *
```

```
          AD     0,8(4)      SIN(EA+C) = SIN(EA) +C*COS(EA)          *
          MD     4,8(4)      C*SIN(EA)                               *
          SDR    2,4         COS(EA+C) = COS(EA) -C*SIN(EA)          *
*       STEP FOURTEEN
          TM     52(13),128             CHECK SIGN OF M
          BZ     PLUS                   OK IF M POSITIVE
*       STEP FIFTEEN
          LCDR   0,0         OTHERWISE COMPLIMENT SIN(EA)
          LD     4,TWOPI      * AND
          SD     4,0(4)       * SET
          STD    4,0(4)       * EA = 2PI-EA
*       STEP SIXTEEN
PLUS      STD    0,8(4)           OUTPUT SIN(EA)
          STD    2,16(4)          OUTPUT COS(EA)
*       STEP SEVENTEEN
EXIT      LM     14,4,12(13)      RESTORE REGISTERS
          BR     14               AND RETURN
TWOPI     DC     D'6.2831853071795864'    2PI
PI        DC     X'413243F6A8885A31'       PI
ZERO14    DC     X'4E00000000000000'
ONEX      DC     X'40FFFFFFFFFFFFF'
          END
                 =E'.99'
                 =E'1.57'
                 =E'1.5707963'
                 =E.7853981'
                 =E'.6666667'
                 =V(SINCOS)
                 =E'1.E-5'
                 =E'1.E-8'
INTIMO    START 0
          BC     15,12(15)              BRANCH AROUND CONSTANTS
          DC     X'7'                   ESTABLISH A HALF-WORD BOUNDARY
          DC     CL7' INTIMO'           NAME
          STM    14,12,12(13)           SAVE THE REGISTERS
          BALR   12,0                   BASE REGISTER
          USING *,12
          LR     3,1
          L      7,0(3)
          CLI    3(7),X'2'              CHECK THE INDICATOR
          BC     8,INTERV               BRANCH IF SECOND ENTRY
          STIMER TASK,TUINTVL=INTER     SET THE TIMER
          LM     14,12,12(13)           RESTORE THE RGGISTERS
          MVI    12(13),X'FF'           INDICATE CONTROL RETURNEC
          BCR    15,14                  RETURN TO CALLING PROGRAM
INTERV    L      5,INTER                LOAD MAXIMUM TIME
          TTIMER
          SR     5,0                    DETERMINE ELAPSED TIME
          M      4,TWSIX                CONVERT UNITS TO MICRO SECONDS
          ST     5,0(7)                 STORE TIME IN RETURN LOCATION
          LM     14,12,12(13)           RESTORE REGISTERS
          MVI    12(13),X'FF'           INDICATE CONRROL RFTURNED
          BCR    15,14                  RETURN TO CALLING PROGRAM
          DS     0F
INTER     DC     F'2147483647'
TWSIX     DC     F'26'
          END
```

```
DPSC     CSECT
         USING *,15
         STM   14,2,12(13)     SAVE REGS
         LM    1,2,0(1)        X,SC ADDRESSES
         LD    0,0(1)          X TO FR0
         LA    15,SINCOS       SINCOS ADD
         BALR  14,15           SIN TO FR0, COS TO FR0
         STD   0,0(2)          OUTPUT SIN(X)
         STD   2,8(2)          OUTPUT COS(X)
         LM    14,2,12(13)     RESTORE REGS
         BR    14              AND RETURN
         ENTRY SINCOS
         USING *,15
SINCOS   LDR   6,0             SIGN OF SIN(A)
         LPDR  0,0             X ABSOLUTE
         CD    0,TWOPI    *
         BL    *+20       *   IF X GREATER THAN 2PI
         LDR   2,0        *   REDUCE
         DD    2,TWOPI    *   TO
         AW    2,ZERO14   *   MODULO
         MD    2,TWOPI    *   TWOPI
         SDR   0,2        *
         CD    0,PI         *
         BL    *+12            * IF X GREATER THAN PI
         LCDR  6,6             * REVERSE SIGN OF SINE
         LCDR  0,0             * AND SET X = 2PI-X
         AD    0,TWOPI         *
         LA    0,2        SET SWITCH = 2  (COS +, NO SIN/COS EXCHANGE)
         CD    0,PIOVER2    *
         BL    *+14         * IF X GREATER THAN PI/2
         LA    0,1          * SET X =PI-X
         LCDR  0,0          * AND SET SWITCH =1 (COS -)
         AD    0,PI         *
         CD    0,PIOVER4       * IF X GREATER THAN PI/4
         BL    *+16            * SET X = PI/4 +PI/4 -X
         LCR   0,0             * AND SET SWITCH -     (SIN/COS EXCHANGE)
         LCDR  0,0             *
         AD    0,PIOVER4       *
         AD    0,PIOVER4       *
         LER   2,0
         AU    2,=X'45000000' 4500000X  1ST HEX DIGIT OF X  =INDEX
         STE   2,68(13)
         L     1,68(13)        4500000I
         SLL   1,3             8I    =CONSTANT INDEX  I=0,1,2,...,12
         LDR   2,0             X TO FR2 AND FR0
         MDR   2,2             X*X =X2
         LDR   4,2             X2
         MD    4,S5(1)
         AD    4,S4(1)
         MDR   4,2
         AD    4,S3(1)
         MDR   4,2
         AD    4,S2(1)
         MDR   4,2
         AD    4,S1(1)
         MDR   0,4             SIN(X) VIA 9TH DEGREE ODD POLYNOMIAL
```

```
         LDR    4,2              X2
         CE     4,=E'.25'        X2 VS 1/4
         BL     *+18             8TH DEGREE COS IF X LESS THAN 1/2
         MD     2,C6-64(1)
         AD     2,C5(1)          10TH DEGREE COS IF X GREATER THAN 1/2
         MDR    2,4
         B      *+8
         MD     2,C5(1)
         AD     2,C4(1)
         MDR    2,4
         AD     2,C3(1)
         MDR    2,4
         AD     2,C2(1)
         MDR    2,4
         AD     2,C1(1)          COS(X) VIA 8 OR 10 DEGREE POLYNOMIAL
         LTR    0,0                   TEST SWITCH FOR SIGN
         BP     *+12             NO EXCHANGE IF +
         LCR    0,0                   OTHERWISE SET SWITCH +1 OR +2
         LDR    4,2                   AND
         LDR    2,0                   EXCHANGE
         LDR    0,4                   SIN AND COS
         BCT    0,*+6          * IF SWITCH = 1
         LCDR   2,2            * SET COSINE -
         LTDR   6,6              * IF SIGN OF SIN +
         BCR    10,14            * EXIT
         LCDR   0,0            OTHERWISE SET SIGN -
         BR     14             AND EXIT
         USING  *,15
         ENTRY  DCOS,DCOSX
DCOS     L      1,0(1)
         LD     0,0(1)
         LNDR   0,0
         AD     0,PIOVER4
         AD     0,PIOVER4
         LA     15,DSINX
         BR     15
         USING  *,15
DCOSX    LNDR   0,0
         AD     0,PIOVER4
         AD     0,PIOVER4
         LA     15,DSINX
         BR     15
         ENTRY  DSIN,DSINX
DSIN     L      1,0(1)
         LD     0,0(1)
         LA     15,12(15)
         USING  *,15
DSINX    LDR    6,0
         LPDR   0,0
         CD     0,TWOPI
         BL     *+20
         LDR    2,0
         DD     2,TWOPI
         AW     2,ZERO14
         MD     2,TWOPI
```

```
        SDR   0,2
        CD    0,PI
        BL    *+10
        LCDR  6,6
        SD    0,PI
        CD    0,PIOVER2
        BL    *+10
        LCDR  0,0
        AD    0,PI
        CD    0,PIOVER4
        BNL   COSS
        LER   2,0
        AU    2,=X'45000000'
        STE   2,68(13)
        L     1,68(13)
        SLL   1,3
        LDR   2,0
        MDR   2,2
        LDR   4,2
        MD    4,S5(1)
        AD    4,S4(1)
        MDR   4,2
        AD    4,S3(1)
        MDR   4,2
        AD    4,S2(1)
        MDR   4,2
        AD    4,S1(1)
        MDR   0,4
        LTDR  6,6
        BCR   10,14
        LCDR  0,0
        BR    14
COSS    LCDR  0,0
        AD    0,PIOVER4
        AD    0,PIOVER4
        LER   2,0
        AU    2,=X'45000000'
        STE   2,68(13)
        L     1,68(13)
        SLL   1,3
        MDR   0,0
        LDR   2,0
        CE    2,=E'.25'
        BL    *+18
        MD    0,C6-64(1)
        AD    0,C5(1)
        MDR   0,2
        B     *+8
        MD    0,C5(1)
        AD    0,C4(1)
        MDR   0,2
        AD    0,C3(1)
        MDR   0,2
        AD    0,C2(1)
        MDR   0,2
```

```
         AD     0,C1(1)
         LTDR   6,6
         BCR    10,14
         LCDR   0,0
         BR     14
         DS     0D
ZERO14   DC     X'4E00000000000000'
TWOPI    DC     D'6.28318530717958 64'   2PI
PI       DC     X'413243F6A8885A31'       PI
PIOVER2  DC     X'411921FB54442D18'       PI/2
PIOVER4  DC     X'40C90FDAA22168C2'
*         SINE COEFFICIENTS
S1       DC     X'40FFFFFFFFFFFFFA'           0.0
         DC     X'40FFFFFFFFFFFF37'           0.6250000000D-01
         DC     X'40FFFFFFFFFFFD7C'           0.1250000000D 00
         DC     X'40FFFFFFFFFFE8D8'           0.1875000000D 00
         DC     X'40FFFFFFFFFFC96A'           0.2500000000D 00
         DC     X'40FFFFFFFFFF49BA'           0.3125000000D 00
         DC     X'40FFFFFFFFFCFB23'           0.3750000000D 00
         DC     X'40FFFFFFFFF01C57'           0.4375000000D 00
         DC     X'40FFFFFFFFCC5C90'           0.5000000000D 00
         DC     X'40FFFFFFFF6D8C98'           0.5625000000D 00
         DC     X'40FFFFFFFE63DA15'           0.6250000000D 00
         DC     X'40FFFFFFFC1E8C81'           0.6875000000D 00
         DC     X'40FFFFFFF85A9960'           0.7500000000D 00
S2       DC     X'C02AAAAAAAAA713F'           0.0
         DC     X'C02AAAAAAAA931F5'           0.6250000000D-01
         DC     X'C02AAAAAAAA8FD27'           0.1250000000D 00
         DC     X'C02AAAAAAAA2D215'           0.1875000000D 00
         DC     X'C02AAAAAAA9E6858'           0.2500000000D 00
         DC     X'C02AAAAAAA8CDDDA'           0.3125000000D 00
         DC     X'C02AAAAAAA4CBBA7'           0.3750000000D 00
         DC     X'C02AAAAAA947FA52'           0.4375000000D 00
         DC     X'C02AAAAAA71E4DEB'           0.5000000000D 00
         DC     X'C02AAAAAA2826B13'           0.5625000000D 00
         DC     X'C02AAAAA98025048'           0.6250000000D 00
         DC     X'C02AAAAA84FD178E'           0.6875000000D 00
         DC     X'C02AAAAA69F0419B'           0.7500000000D 00
S3       DC     X'3F222221FB15B5CF'           0.0
         DC     X'3F22222212475F04'           0.6250000000D-01
         DC     X'3F2222221B88AD9A'           0.1250000000D 00
         DC     X'3F22222121225BEF1'          0.1875000000D 00
         DC     X'3F2222211106E925'           0.2500000000D 00
         DC     X'3F22222203229637'           0.3125000000D 00
         DC     X'3F222221D93FF490'           0.3750000000D 00
         DC     X'3F22222 15CBB1C5D'          0.4375000000D 00
         DC     X'3F222220934E1941'           0.5000000000D 00
         DC     X'3F22221F3A705D7F'           0.5625000000D 00
         DC     X'3F22221CBB9904EF'           0.6250000000D 00
         DC     X'3F222218FE37316B'           0.6875000000D 00
         DC     X'3F2222146F331054'           0.7500000000D 00
S4       DC     X'BDCFBE6DAB33F92D'           0.0
         DC     X'BDD008956DAAEA4B'           0.6250000000D-01
         DC     X'BDD00C4AC2C8225A'           0.1250000000D 00
         DC     X'BDD00C15380ED130'           0.1875000000D 00
```

```
         DC     X'BDD00C4960797744'          0.2500000000D 00
         DC     X'BDD00C0081D853F7'          0.3125000000D 00
         DC     X'BDD00B3D87649F44'          0.3750000000D 00
         DC     X'BDD0099428EEBB20'          0.4375000000D 00
         DC     X'BDD00789E59C8C04'          0.5000000000D 00
         DC     X'BDD004BB8BF3436D'          0.5625000000D 00
         DC     X'BDD000816BFF2313'          0.6250000000D 00
         DC     X'BDCFFB44C616A714'          0.6875000000D 00
         DC     X'BDC=F5CCED99F6BD'          0.7500000000D 00
S5       DC     X'BD27ABC0019B0975'          0.0
         DC     X'3C27402 0A16A8780'         0.6250000000D-01
         DC     X'3C2DC3E3A8A96C91'          0.1250000000D 00
         DC     X'3C2DE2C5F202B5DE'          0.1875000000D 00
         DC     X'3C2E01B434E997D5'          0.2500000000D 00
         DC     X'3C2DF9F7766C5869'          0.3125000000D 00
         DC     X'3C2DE4A5BC081B71'          0.3750000000D 00
         DC     X'3C2DC260BE20B1AB'          0.4375000000D 00
         DC     X'3C2DA291A6506887'          0.5000000000D 00
         DC     X'3C2D7F6C3453E649'          0.5625000000D 00
         DC     X'3C2D545C68C1566F'          0.6250000000D 00
         DC     X'3C2D284AE4CA5DCB'          0.6875000000D 00
         DC     X'3C2D00E6F7D092D6'          0.7500000000D 00
*          COSINE COEFFICIENTS
C1       DC     X'4110000000000000'          0.0
         DC     X'4110000000000000'          0.6250000000D-01
         DC     X'4110000000000010'          0.1250000000D 00
         DC     X'40FFFFFFFFFFF6B5'          0.1875000000D 00
         DC     X'40FFFFFFFFFF02CC'          0.2500000000D 00
         DC     X'40FFFFFFFFF9D8B0'          0.3125000000D 00
         DC     X'40FFFFFFFFDB4BF3'          0.3750000000D 00
         DC     X'40FFFFFFFF66CA3C'          0.4375000000D 00
         DC     X'40FFFFFFFFBFB694'          0.5000000000D 00
         DC     X'411000000002779F'          0.5625000000D 00
         DC     X'4110000000000C484'         0.6250000000D 00
         DC     X'40FFFFFFFFF0ACD4'          0.6875000000D 00
         DC     X'40FFFFFFFF328278'          0.7500000000D 00
C2       DC     X'C0800000000008FC'          0.0
         DC     X'C080000000001F56'          0.6250000000D-01
         DC     X'C08000000000A56E'          0.1250000000D 00
         DC     X'C07FFFFFFFFB422C'          0.1875000000D 00
         DC     X'C07FFFFFFFC18410'          0.2500000000D 00
         DC     X'C07FFFFFFEF2E9A8'          0.3125000000D 00
         DC     X'C07FFFFFFBA0034A'          0.3750000000D 00
         DC     X'C07FFFFFF24D4282'          0.4375000000D 00
         DC     X'C07FFFFFFB7C7615'          0.5000000000D 00
         DC     X'C080000002240B38'          0.5625000000D 00
         DC     X'C08000000006D49C3'         0.6250000000D 00
         DC     X'C07FFFFFFF1A5797'          0.6875000000D 00
         DC     X'C07FFFFFF88B5A64'          0.7500000000D 00
C3       DC     X'3FAAAAAAACFAD91B'          0.0
         DC     X'3FAAAAAAABB4F60A'          0.6250000000D-01
         DC     X'3FAAAAAAAC8F2FE5'          0.1250000000D 00
         DC     X'3FAAAAAA9BD824DF'          0.1875000000D 00
         DC     X'3FAAAAAA487F9F90'          0.2500000000D 00
         DC     X'3FAAAAA985B06FCA'          0.3125000000D 00
```

```
        DC    X'3FAAAAA756C2D236'      0.3750000000D 00
        DC    X'3FAAAAA2D85E052B'      0.4375000000D 00
        DC    X'3FAAAAA8A34E1B3E'      0.5000000000D 00
        DC    X'3FAAAAAB6650E57D'      0.5625000000D 00
        DC    X'3FAAAAAABBBF8614'      0.6250000000D 00
        DC    X'3FAAAAAA5544824F'      0.6875000000D 00
        DC    X'3FAAAAA8E1C6081B'      0.7500000000D 00
C4      DC    X'BE5B05E544ACFAED'      0.0
        DC    X'BE5B05B3A3D232D3'      0.6250000000D-01
        DC    X'BE5B05B13C670BD8'      0.1250000000D 00
        DC    X'BE5B059A0ECCDD46'      0.1875000000D 00
        DC    X'BE5B0563A45C8E91'      0.2500000000D 00
        DC    X'BE5B0511816D0F42'      0.3125000000D 00
        DC    X'BE5B046D420B0F0B'      0.3750000000D 00
        DC    X'BE5B0375D64C08AC'      0.4375000000D 00
        DC    X'BE5B053B6D350E39'      0.5000000000D 00
        DC    X'BE5B05CF9CE1EFA5'      0.5625000000D 00
        DC    X'BE5B05AF4705E015'      0.6250000000D 00
        DC    X'BE5B059FF6063D5B'      0.6875000000D 00
        DC    X'BE5B0576C215745A'      0.7500000000D 00
C5      DC    X'3D1B83EA0F58760F'      0.0
        DC    X'3D1A0381E55BED90'      0.6250000000D-01
        DC    X'3D19FD902FF6A01A'      0.1250000000D 00
        DC    X'3D19F15D857FB484'      0.1875000000D 00
        DC    X'3D19E3CB6B016FA1'      0.2500000000D 00
        DC    X'3D19D6B96D105CAD'      0.3125000000D 00
        DC    X'3D19C48042D34407'      0.3750000000D 00
        DC    X'3D19B076C277FE83'      0.4375000000D 00
        DC    X'3D19F45FAA3F5386'      0.5000000000D 00
        DC    X'3D1A040A69823422'      0.5625000000D 00
        DC    X'3D1A0107B2439E18'      0.6250000000D 00
        DC    X'3D19FFE547902218'      0.6875000000D 00
        DC    X'3D19FD9AC88121EB'      0.7500000000D 00
C6      DC    X'BB4010A69645256D'      0.5000000000D 00
        DC    X'BB4ABF0291131076'      0.5625000000D 00
        DC    X'BB48F9D4FC6AC452'      0.6250000000D 00
        DC    X'BB487121A5139E61'      0.6875000000D 00
        DC    X'BB479FD89AA019C2'      0.7500000000D 00
        END
              =X'45000000'
              =E'.25'
```