



MEMORANDUM

1988 December 16 Revision 1

To: Distribution

From: C. Uphoff

Subject: Orbit Propagator for Close Orbiters in an Oblate Field

This memorandum is a description of a FORTRAN subroutine package for propagation of the state of a spacecraft in close orbit about an oblate planet. The heart of the package is a subroutine called KOZSAK, based on the work of Kozai¹ and Izsak², that performs the transformation from mean to osculating orbital elements in what Brouwer³ called the main problem of artificial satellite theory. The mean elements are propagated according to Brouwer's theory up through terms of order J_2^2 in the mean rates of the node, pericenter, and mean anomaly. The FORTRAN listings attached here include a main program that shows how to use the software to generate an ephemeris. It is expected that most users will wish to write their own routines to solve a specific problem or provide specific information.

Theoretical Background

Izsak² wrote the equations for the first order variations in the osculating elements in terms of the ingenious variables of Hill⁴ viz. the set $\{\dot{r}, G, H, r, u, h\}$ where r and \dot{r} represent the radius and its time derivative, G is the orbital angular momentum, u is the argument of latitude, H is the z-component of G , and h is the longitude of the ascending node. In terms of the more familiar Keplerian elements, the variables used here are:

$$G = \sqrt{\mu a (1-e^2)}, H = G \cos i, r = \frac{a (1-e^2)}{1 + e \cos f}, \text{ and } \dot{r} = \sqrt{\frac{\mu}{p}} e \sin f.$$

Also note that $u = \omega + f$, where ω is the argument of pericenter and f is the true anomaly. In the above, a is the semi-major axis and e the eccentricity of the Keplerian ellipse. We shall also use the expressions $p = a (1 - e^2)$ and $l = \text{mean anomaly}$ and $\theta = H/G = \cos i$. By treating the Hill variables as a canonical set in the Hamilton-Jacobi determining function derived by Brouwer, Izsak obtained the following expressions for the short-periodic variations in the "coordinate" \dot{r} and the "momenta" r and u :

$$\delta \dot{r} = \frac{J_2 R_e^2 \mu^{\frac{1}{2}}}{4p^{\frac{5}{2}}} \left\{ (3\theta^2 - 1)e \sin f \left[\sqrt{1-e^2} + \frac{(1+e \cos f)^2}{1 + \sqrt{1-e^2}} \right] - 2(1-\theta^2)(1+e \cos f)^2 \sin 2u \right\},$$

$$\delta r = -\frac{J_2 R_e^2}{4p} \left\{ (3\theta^2 - 1) \left[\frac{2\sqrt{1-e^2}}{1+e \cos f} + \frac{e \cos f}{1 + \sqrt{1-e^2}} + 1 \right] - (1-\theta^2) \cos 2u \right\}, \text{ and}$$

$$\delta u = -\frac{J_2 R_e^2}{8p^2} \left\{ 6(1-5\theta^2)(f-l) + 4 \left[(1-6\theta^2) + \frac{1-3\theta^2}{1 + \sqrt{1-e^2}} \right] e \sin f \right. \\ \left. + \frac{1-3\theta^2}{1 + \sqrt{1-e^2}} e^2 \sin 2f + 2(5\theta^2-2)e \sin(2u-f) \right. \\ \left. + (7\theta^2-1) \sin 2u + 2\theta^2 e \sin(2u+f) \right\}.$$

These very useful expressions eliminate the confusing difficulties often associated with satellites of very small eccentricity. By using the expressions above in conjunction with the more familiar ones below, one can transform from mean to osculating elements (directly) or from osculating to mean (by a simple iteration) without the numerical sensitivity of the pericenter to near-zero values of the orbital eccentricity.

With the expressions above, we use the following short-periodic variations in the angular momentum and inclination and those of the ascending node from the work of

Kozai to complete the transformation from mean to osculating elements. First note that, because

$$G = \sqrt{\mu p} \quad , \quad \text{then} \quad 2 \frac{\delta p}{p} = \frac{\delta G}{G} \quad ,$$

and, because the Hamiltonian describing the perturbations does not depend upon the position of the node, the axial angular momentum, H , is conserved, which is to say that

$$\delta H = \delta (G \cos i) = 0 \quad , \quad \text{or}$$

$$\frac{\delta G}{G} = \tan i \quad \delta i = 2 \frac{\delta p}{p} \quad .$$

Thus, the constancy of the z-component of angular momentum allows us to use Kozai's² expression for the short-periodic variation in inclination to obtain the corresponding variation in the semi-latus rectum, p :

$$\delta i = \frac{3}{4} \frac{J_2 R_e^2}{p^2} \sin i \cos i \left[\cos 2u + e \cos (2u - f) + \frac{e \cos (2u + f)}{3} \right] \quad ,$$

from which we have

$$\delta p = \frac{3}{2} \frac{J_2 R_e^2}{p} \sin^2 i \left[\cos 2u + e \cos (2u - f) + \frac{e \cos (2u + f)}{3} \right] \quad .$$

The only remaining variable is the ascending node, h (or Ω), whose short-periodic variation we obtain from Kozai's 1959 paper:

$$\delta \Omega = \delta h = -\frac{3}{2} \frac{J_2 R_e^2}{p^2} \cos i \left[(f - l) + e \sin f - \frac{\sin 2u}{2} - \frac{e \sin (2u - f)}{2} - \frac{e \sin (2u + f)}{6} \right] \quad .$$

The constants of the problem above are represented by the gravitational strength of the central planet, μ ($= GM$), R_e is the equatorial radius, and J_2 is the second zonal harmonic coefficient describing the dynamic effect of the planet's oblateness, given explicitly by the potential

$$V = -\frac{\mu}{r} \left[1 - \sum_{k=2}^{\infty} J_k \left(\frac{R_e}{r} \right)^k P_k(\sin \varphi) \right] \quad ,$$

where $P_k(\sin \phi)$ is the Legendre polynomial of degree k in the sine of the latitude, ϕ , and with $J_k = 0$ for $k > 2$.

Osculating and Mean Elements

In the formulae above, it is assumed that the expressions are evaluated using the mean values of the orbital elements on the right hand sides. The variations (the δ 's) are then added to the mean values to obtain the osculating values of the variables. This is a straightforward procedure and can be carried out without any iteration. But Nature does not provide us with mean elements; the more usual problem starts with an actual (osculating) position and velocity of a spacecraft and it is desired to know where it will be at some time in the future. The mean elements are a construct that allows us to approximate the long-term motion of the spacecraft. In nearly every real problem, then, we are given a set of osculating variables and it is required to obtain the mean values at the epoch to start the long term prediction. If the short-periodic variations are small, it is sometimes sufficient to use the osculating values on the right-hand sides of the equations above and then subtract the variations thus obtained from the initial elements to yield an approximation to the mean elements. In subroutine KOZSAK, to be described later, provision is made to iterate for the mean elements using the osculating elements on the right-hand sides of the equations as a starting guess. This procedure provides a consistent set of mean elements which, if substituted into the formulae, will yield the initial osculating elements that were used to start the procedure. If the iteration is not performed, the initial elements obtained by the method will be slightly different from those that were input. This is not only esthetically unsatisfactory, it is also confusing. Because the iteration is quite easily performed and, in the author's experience, stable and quickly convergent, it is recommended that the user always make sure that the conversion control parameter MNOSC is set to -3 or -4 (to allow three or four iterations) for the conversion of osculating to mean elements.

Once the transformation from mean to osculating (or vice versa) Hill variables is complete, it remains only to calculate the Keplerian quantities associated with the transformed quantities for printout or for use in other calculations. The transformed eccentricity (either mean or osculating) is easily obtained from the relations

$$A = e \cos f = \frac{p}{r} - 1 \quad \text{and} \quad B = e \sin f = \sqrt{\frac{p}{\mu}} \dot{r}$$

Now , $e = \sqrt{A^2 + B^2}$, $f = \arctan (B/A)$, and $\omega = u - f$,

$$\text{along with } a = p/(1-e^2)$$

complete the transformation to Keplerian elements.

Propagation of the Mean Elements

The mean elements as a function of time are obtained from Brouwer's³ theory exactly as written by Aksnes⁵ in a paper that shows how to use Hill's variables in a theory that accounts for the effects of higher order harmonics and includes the long-periodic variations due to the motion of the pericenter. These are not included in the theoretical writeup here because I intend to use a different method of propagating the so-called "secular" terms of the motion and the long-periodic variations in such a way as to account for the major part of drag perturbations and without restrictions on use near the critical inclination. But this implementation will require more time and testing as the method has not been tried before. It seemed wise to distribute the work to date so as to provide a fairly useful orbit propagator that will be good to a few hundred meters over a period of several tens of revolutions provided that drag is not a major factor and that the orbit period is less than about 18 hours so that luni-solar perturbations do not become dominant. The effects of J_3 and J_4 are included in the listing and are automatically set for use in the program being distributed for use on the Macintosh and PC computers. These effects were coded directly from Aksnes's⁵ paper.

The mean elements l , g (or ω), and h (or Ω) are identical to Brouwer's l'' , g'' , and h'' and are given by

$$l = l_0 + nt - \frac{3}{4} \gamma \sqrt{1-e^2} \left[1 - 3c^2 - \frac{1}{32} \gamma \left\{ 10(1 - 6c^2 + 13c^4) - 5(5 - 18c^2 + 5c^4)e^2 + 16 \sqrt{1-e^2} (1 - 6c^2 + 9c^4) \right\} \right] \cdot nt ,$$

$$g = \omega = \omega_0 - \frac{3}{4} \gamma \left[1 - 5c^2 + \frac{1}{32} \gamma \left\{ 2(5 + 43c^2)(1 - c^2) + (25 - 126c^2 + 45c^4)e^2 - 24 \sqrt{1-e^2} (1 - 8c^2 + 15c^4) \right\} \right] \cdot nt ,$$

and

$$h = \Omega = \Omega_0 - \frac{3}{2} \gamma c \left[1 - \frac{1}{16} \gamma \left(4 - 40c^2 - (9 - 5c^2)e^2 + 12\sqrt{1-e^2} (1 - 3c^2) \right) \right] \cdot nt ,$$

where, with Aksnes⁵, we have employed the abbreviations,

$$\gamma = J_2 \left(\frac{R_e^2}{p^2} \right), c = H/G = \cos i, \text{ and } n = \sqrt{\mu/a^3}$$

and it is understood that mean values of the elements are to be used on the right hand sides of the equations and that the mean values of a , e , and i are constants.

The propagation of the mean elements is done in the main program called KOZTST only because this code is expected to be temporary. When higher order harmonics, drag, and long-periodic terms are added, this propagation will be done in its own separate subroutine. The user may note that the terms proportional to the square of J_2 are included only if the flag J2SQ is set to a non-zero value. Although the FORTRAN listing is intended to be self-explanatory, the next section contains a sample case showing how to use the program with suggestions for how to use the output files to get graphical output from existing plotting programs. The sample case is compared with a numerically integrated trajectory to give the potential user an idea of the accuracy to expect for a typical close Earth orbiter.

Sample Case for Desktop Computers

When you get the program into your Mac or PC, launch(double click) the application (Mac) or execute the program (PC) by whatever means your operating system requires. UNIX users call your local systems expert. The first thing the program does is give you a polite request to name the filename path to the file where you want your output to go. On a PC, name the directory or drive and give the Unit 6 output file a descriptive name like A:Unit6.Out. On a Mac, give the Unit 6 output file a descriptive name like KOZTSTDisk:SampleCaseFolder:LEOSample. In either case, make sure the path you specify exists and that there's enough room on the disk or device to hold the output you expect from the program. If you don't understand this and you're on a Mac, call the file AJAX6 (or Sam) and the file will appear in the folder from which you launched the application. If you're on a UNIX system, call your local systems expert.

After you get an output file set up for the Unit 6 output, you get a chance to do the same thing for the Unit99 output which is tab-delimited output for pasting or importing into plotting programs or spreadsheets. On Macs and PC's you do the same thing you just did for Unit 6 only you give the file a slightly different name (Like Comet instead of AJAX or Mary Jane instead of Sam). On a UNIX system, you call your local systems expert back to find out how to change the number and to help you reload your system when it crashes because you put a comma where a slash should be.

After you successfully establish the two files where your output is going, the program will ask you for an initial orbit which is assumed to be in the Earth True Equator and

Equinox frame of Epoch (Epoch means the time associated with the elements you're going to input). Input the elements in the order given and use kilometers for semi-major axis and degrees for the angular elements. Let's go through everything so far.

{Launch or Execute KOZTST}

Computer says:

Please input the Filename Path for Unit 6 Output:

You type:

Koz6 {for example}

Computer says:

Please input the Filename Path for Unit 99 Output:

You type:

Koz99 {Enter a carriage return when you're finished typing.}

Computer says:

Please enter initial Orbit (a,e,f,w,i,Node)

You type:

7545. , .005 , 0. , 30. , 28.5, 0. {Spaces don't count; separate numbers by commas.}

O.K., your initial orbit is established. It's an 1167 km altitude orbit (a= 7545 km), nearly circular with the spacecraft at perigee (f = 0 deg), the perigee point is 30 degrees along the orbit from the ascending node, the inclination is 28.5 degrees, and the line to the ascending node points to the vernal equinox. Now it's time to tell the main program what kind of output you want and how much of it. The next polite statement from the computer is:

Computer says:

Please enter run control values:

MNOSC, KPROP, KOUT, KU99, DELT, TF, TS, TCON, NODALP, NREV, J2SQ

You type:

-3, 1, 1, 3, 600., 6000., 0., 1., 0, 0, 1 {On a Mac, or PC you can skip the decimal points}

You just told the program that the initial elements are the osculating (instantaneous) values and that you want it to try 3 iterations to convert to mean elements. KPROP = 1 means you want to propagate the orbit in time, KOUT = 1 means you want output to the screen or console (Note: Writing to the screen on a Mac is very slow; don't do it for long runs.) KU99 = 3 means you want both osculating orbital elements and position and velocity tabulated as a function of time in the file Koz99. DELT = 600. means you want ephemeris data every 600 seconds, TF = 6000 sec is the final time, TS is the start time for the ephemeris and is set to zero here which means you'll get printout starting at the time for which you input the elements. TCON = 1. means you'll get the time printed out in seconds. (Time is divided by TCON for printout; if you want time printed in minutes, set TCON to 60; internal units for time are always seconds.) NODALP and NREV are set to zero to indicate that you don't want to do an iteration to solve for the nodal period of the orbit over NREV revolutions. J2SQ = 1 means that you want to include the effects of the square of J_2 in the simulation. Make sure you have what you want entered on this line before you hit carriage return because this is it; from now on, the program will run like a batch job on a mainframe and the next thing you get to do is input a new case or look at the output or, on UNIX systems, try to figure out what happened to the output files and call your local systems expert. If you're on a Mac or PC and you have trouble, read the listing and try again. WARNING: the FORTRAN used for the Mac version will try to spool the Unit 6 output to the system printer. If you want to keep the Unit 6 file on your hard disk, make sure the program can't find a routine called spool.sub by moving it to another folder or throwing it away. When the run finishes, you will then get a dialogue box that asks you to find spool.sub. Just click the cancel button and you can go look at the Unit 6 output using a text editor. If this is too much trouble, recompile using some other number (like 88) in place of the traditional value 6 for your non-tab-delimited output file.

If you have a lot of trouble, you can send me a nasty message at CO-10, or call me at 6480 and say nasty things. If you're on a UNIX system, though, don't call me --- call your local systems expert. After each case is finished, the program will return to let you input a new orbit and the output will go to the same files as the previous case. If you want to quit and look at the output, enter a negative number for the semi-major axis, that is, enter

-1,0,0,0,0 {carriage return} .

To stop at any time on a Mac, hold down the Command key and type a period.

Here's the Unit 6 output for the sample case.
(This is almost the same as the screen Output.)

```

INPUT ORBITAL ELEMENTS: (a,e,f,w,i,Node)
  7545.0000  .0050000  .0000  30.0000  28.5000  .0000
MNOSC,KPROP,KOUT,KU99,DELT,TF,TS,TCON,NODALP,NREV,J2SQ
-3 1 1 3 600.0 6000. .0000 1.000 0 0 1
  .00  7545.0000  .0050000  0.0000  30.0000  28.5000  0.0000
      7543.9383  .0041194  1.0700  28.9513  28.4930  -.0255  30.0213
  600.00  7542.7696  .0046230  29.2911  34.1738  28.4846  -.0363
      7543.9383  .0041194  34.4773  29.0065  28.4930  -.0594  63.4838
 1200.00  7542.0128  .0043774  61.1993  35.6661  28.4794  -.1008
      7543.9383  .0041194  67.7968  29.0618  28.4930  -.0933  96.8586
 1800.00  7543.5942  .0041326  90.8020  39.3137  28.4906  -.1566
      7543.9383  .0041194 100.9735  29.1170  28.4930  -.1272 130.0905
 2400.00  7545.5575  .0035717 122.4715  40.7148  28.5045  -.1778
      7543.9383  .0041194 133.9997  29.1722  28.4930  -.1611 163.1719
 3000.00  7545.5642  .0031926 163.8819  32.2490  28.5047  -.1796
      7543.9383  .0041194 166.9161  29.2275  28.4930  -.1950 196.1436
 3600.00  7543.6181  .0034779 -154.7728  23.8292  28.4910  -.1998
      7543.9383  .0041194 -160.2018  29.2827  28.4930  -.2289 -130.9192
 4200.00  7542.0085  .0038321 -120.4636  22.5256  28.4796  -.2543
      7543.9383  .0041194 -127.2683  29.3379  28.4930  -.2628 -97.9304
 4800.00  7542.6671  .0040546 -85.4380  20.6355  28.4841  -.3185
      7543.9383  .0041194 -94.2133  29.3932  28.4930  -.2967 -64.8201
 5400.00  7544.8743  .0045432 -50.9012  19.3653  28.4994  -.3561
      7543.9383  .0041194 -61.0051  29.4484  28.4930  -.3306 -31.5567
 6000.00  7545.9996  .0050212 -22.4703  24.3100  28.5070  -.3622
      7543.9383  .0041194 -27.6618  29.5036  28.4930  -.3645  1.8418

RUN TIME =.9667 SECONDS Macintosh{16 MHz 68020/68881 AbSoft Fortran/020 v.2.3}
{With Print to Screen}

```

The format is

Time Osculating Elements (a,e,f,w,i,Node)
 Mean Elements (a,e,f,w,i,Node), w+f

Elements are in km and degrees, Time is in Seconds/TCON.

The Unit 99 output is tab-delimited output in the following format.

Time , Osculating Elements (a,e,f,w,i,Node), Cartesian (X,Y,Z,XDOT,YDOT,ZDOT)

Cartesian Coordinates are in km and km/sec., Time is in Seconds/TCON.

(See Listing for options on Unit 99 Output)

Comparison With Numerically Integrated Solution

In this section, we compare the approximate solution above with a numerically integrated solution of the same orbit using J_2 as the only perturbation. In the previous sections, the development of the theory was for J_2 effects only as we hope to include the effects of drag in a later version of the software and will continue the development in a different way than other investigators. But for this distribution, we have included the effects of J_3 and J_4 exactly as given in the 1970 paper by Aksnes⁵. Although this formulation suffers from a singularity near the critical inclination ($\arctan 2$), it is very useful for most other orbits and includes the long-periodic variations due to J_3 and the so-called "secular" variations to first order in J_4 and to 2nd order in J_2 .

The case run previously as a sample case was simulated for one day using full numerical integration of the equations of motion for a model including the 2nd, 3rd, and 4th zonal harmonics of the Earth's gravitational field. The results were compared with an ephemeris generated using the distribution version of KOZTST whose listing is included in this memo.

**Error in Semi-Major Axis vs Time
(1st Order Oblateness Theory)**

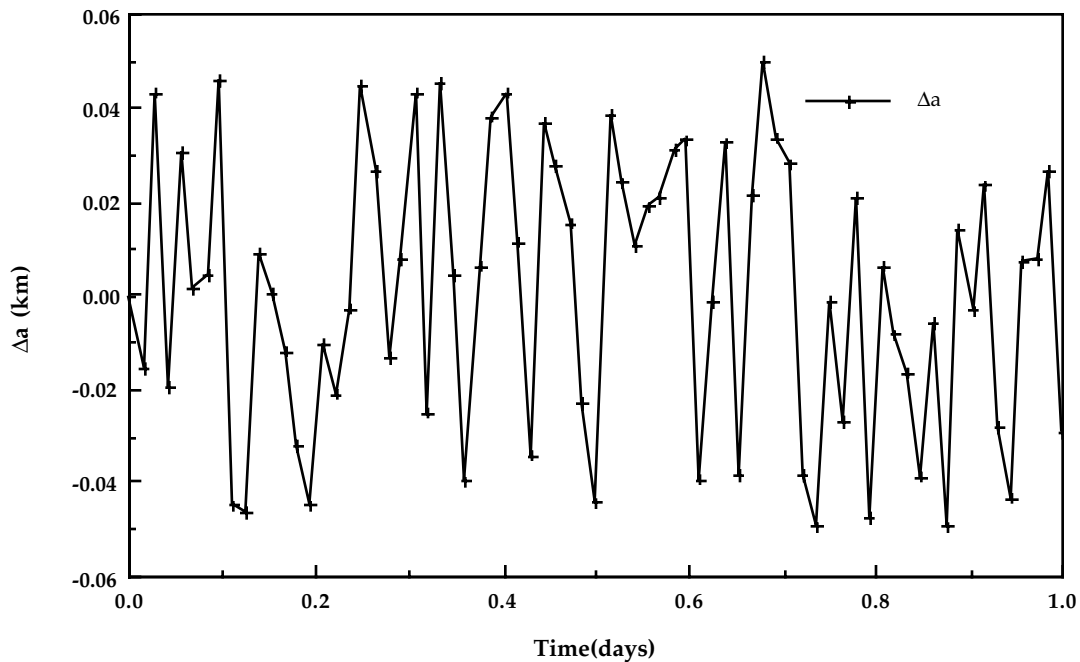


Fig. 1 Error in Semi-major Axis vs Time

The results are shown in representative plots of the error in semi-major axis (Fig. 1) and the error in the argument of latitude (Fig. 2). These results are similar to those

obtained by Aksnes⁵ with the exception of a long term 2nd order effect in mean mean anomaly that is shown as a linear growth of the error in

Along-Track Angular Error vs Time (1st Order Satellite Theory)

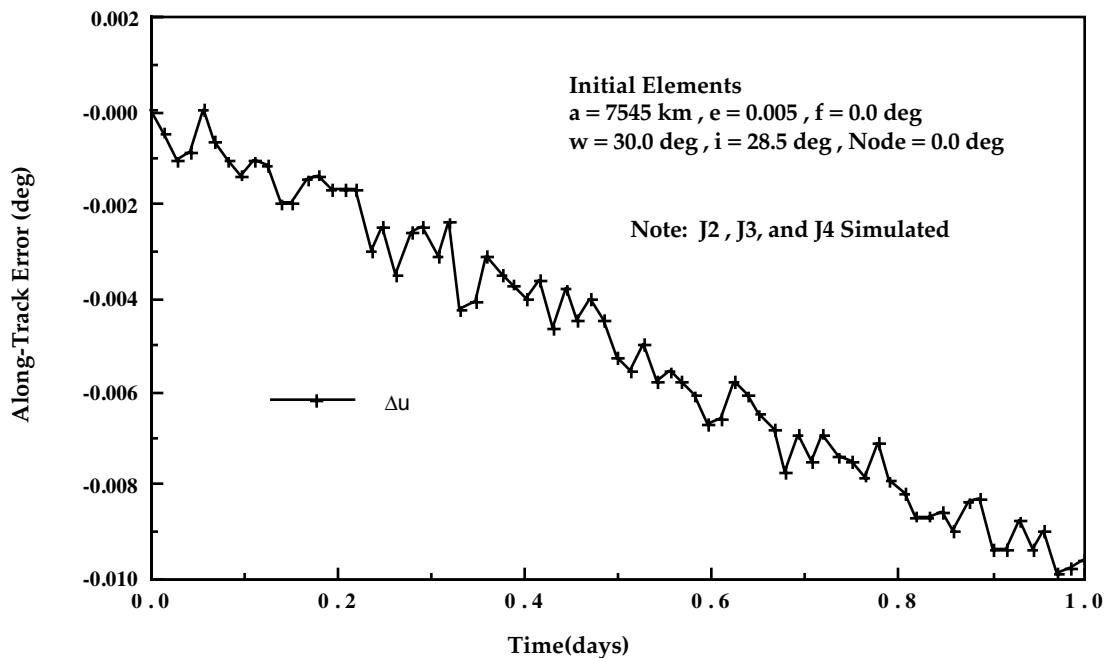


Fig. 2 Error in Argument of Latitude vs Time

the along-track angle. This is to be expected as Aksnes pointed out. He removed this error by calibrating each case with a short numerical integration to adjust the mean semi-major axis so as to minimize the root-sum-square of the residuals. In the current version of the KOZTST program, this error is not removed and is evident in Fig. 2. Thus, in using KOZTST, one can expect along-track error propagation of the order of a little less than 1 km ($a\Delta u$) per day of propagation and errors of the order of 60 meters in the cross-track and orbit-normal directions. In future versions of this development, we hope to remove the secular term analytically if possible and by a calibration integration otherwise.

CONCLUSIONS

An analytic theory of artificial satellites has been implemented for use in preliminary studies of orbit missions near the Earth. The theoretical development includes the effects of J_2 and J_2^2 while the software distributed includes, in addition, the effects of J_3 and J_4 . The implementation has been tested against numerical integration and has been shown to have periodic errors of the order of 60 meters and a long-term growth of the along-track error of less than one km per day of propagation. This long-term error will be eliminated in future versions of the program. It is concluded that the program is adequate for most orbit prediction applications that do not suffer from strong drag effects or luni-solar perturbations. Instructions for using the program have been included and a complete listing is attached as an appendix.

REFERENCES

1. Kozai, Y. 1959 *Astron. J.* **64**, p 367.
2. Izsak, I. 1963 *ibid.* **68**, p 559.
3. Brouwer, D. 1959 *ibid.*, **64**, p 378.
4. Hill, G.W. 1913, *ibid.* **27**, p 171.
5. Aksnes, K. 1972 *Astron. & Astrophys.*, **17**, p 70
6. Kozai, Y. 1962 *Astron. J.* **67**, p 446.

APPENDIX

C PROGRAM KOZTST
C
C THIS IS A MAIN PROGRAM FOR USING SUBROUTINE KOZSAK, A
C FORTRAN SUBROUTINE TO CONVERT FROM MEAN TO OSCULATING
C ORBITAL ELEMENTS AND VICE-VERSA. THIS MAIN PROGRAM ALSO
C PROPAGATES THE MEAN STATE FORWARD IN TIME TO PROVIDE AN
C EPHEMERIS AND A TIME HISTORY OF THE MEAN AND OSCULATING
C ORBITAL ELEMENTS AND, IF REQUESTED, THE OSCULATING POSITION
C AND VELOCITY. THIS WHOLE PACKAGE IS DESIGNED TO BE AS
C MACHINE INDEPENDENT AS POSSIBLE; PLEASE EXCUSE THE
C NON-MACINTOSH INTERFACE WITH THE USER.
C INPUTS ARE AS FOLLOWS:
C
C OUTFIL CHARACTER*64 FILENAME PATH FOR UNIT 6 OUTPUT
C OUTFLX CHARACTER*64 FILENAME PATH FOR UNIT 99 OUTPUT
C THIS IS TAB-DELIMITED OUTPUT FOR PASTING INTO
C PLOTTING PACKAGES LIKE CRICKET GRAPH OR EXCEL.
C
C A,E,F,W,AI,OM INPUT ORBITAL ELEMENTS
C IN TRUE EQUATOR AND EQUINOX OF EPOCH
C
C A .. SEMI-MAJOR AXIS (KM) , E .. ECCENTRICITY
C F .. TRUE ANOMALY (DEG) , W .. ARG. OF PERICENTER (DEG)
C AI .. INCLINATION (DEG) , OM .. LONG. OF ASC. NODE (DEG)
C
C
C NOTE: THESE MAY BE EITHER OSCULATING OR MEAN ELEMENTS
C DEPENDING ON THE SETTING OF THE FLAG MNOSC.
C
C RUN CONTROL PARAMETERS
C
C MNOSC IS THE FLAG TO SPECIFY WHETHER THE INPUT ELEMENTS ABOVE
C ARE MEAN OR OSCULATING. IF MNOSC IS POSITIVE (OR ZERO)
C THEN WE ARE GOING FROM MEAN TO OSCULATING AND THE PROGRAM
C ASSUMES YOU INPUT MEAN ELEMENTS. IF MNOSC IS NEGATIVE, THE
C INPUT ELEMENTS WILL BE TREATED AS OSCULATING. AN ITERATION
C WILL BE PERFORMED ABS(MNOSC) TIMES TO GO FROM OSCULATING TO
C MEAN INITIAL ELEMENTS. SET TO -3 OR -4 FOR CONSISTENT RESULTS.
C SEE SUBROUTINE KOZSAK FOR USE OF MNOSC IN ITERATION.
C
C KPROP IS THE FLAG TO PROPAGATE MEAN ELEMENTS FORWARD AND GENERATE AN
C EPHEMERIS. IF KPROP = 0, THEN CONVERSION WILL BE DONE ONLY FOR
C INITIAL INPUT ELEMENTS.
C
C KOUT IS OUTPUT FLAG FOR EPHEMERIS. IF KOUT = 0 .. NO OUTPUT
C KOUT > 0 .. UNIT 6 OUTPUT
C KOUT < 0 .. UNIT 6 AND SCREEN
C
C KU99 IS OUTPUT FLAG FOR TAB-DELIMITED (UNIT 99) OUTPUT
C IF KU99 = 0 .. NO UNIT 99 OUTPUT
C KU99 = 1 .. OSCULATING ELEMENTS (KM , DEG)
C KU99 = 2 .. OSCULATING POSITION AND VELOCITY (KM,KM/S)
C KU99 = 3 .. BOTH 1 AND 2 (ON ONE LINE)
C

```

C      DELT IS THE TIME INCREMENT FOR THE EPHEMERIS (SECONDS)
C
C      TF IS THE FINAL TIME FOR THE EPHEMERIS (SECONDS)
C
C      TS IS THE STARTING TIME FOR THE EPHEMERIS (SECONDS RELATIVE TO
C      THE EPOCH OF THE INPUT ORBITAL ELEMENTS .. USUALLY ZERO.)
C
C      TCON IS TIME CONVERSION FACTOR FOR PRINTOUT.
C      OUTPUT WILL BE PRINTED AS T/TCON.
C
C      NODALP IS A FLAG TO CALCULATE THE NODAL PERIOD OF THE INPUT ORBIT.
C
C      NREV IS THE NUMBER OF REVS FOR WHICH NODAL PERIOD IS DESIRED.
C
C      J2SQ IS FLAG TO INCLUDE J2 SQUARED TERMS IN MEAN ELEMENTS PROPAGATION.
C
C      THIS CODE WAS DEVELOPED ON THE MACINTOSH PLUS, SE, AND II USING
C      THE ABSOFT MACFORTRAN/020 VERSION 2.3. REFERENCES TO LONG(362)
C      ARE TO GET TO THE SYSTEM CLOCK. CHANGE FOR PC'S OR OTHER MACHINES.
C      THIS SHOULD BE THE ONLY CHANGE REQUIRED UNLESS YOUR COMPILER WANTS
C      TO USE UNIT 9 (CODED AS * HERE) TO REFER TO THE SCREEN OR CONSOLE.
C
C      GOOD LUCK ... IF YOU HAVE TROUBLE .. CALL CHAUNCEY UPHOFF X6480
C
C      NOTE: J2 ONLY IN 1988 NOV 25 VERSION
C
C
C      1988 NOV 25
C
C      IMPLICIT REAL*8 (A-H,O-Z)
C      CHARACTER*64 OUTFIL, OUTFLX
C      CHARACTER*4 Q
C      DIMENSION ELMOUT(6),ELMIN(6),ELMT(6),ELMTOT(6)
1    , XOUT(3),XDOUT(3),ELMDUM(6)
C
C      COMMON /CONST/ RAD,PI,PI2,AU
C      RAD = DEG/RADIAN, PI,PI2 = 2*PI, AU = EARTH TO SUN (KM)
C
C      COMMON /GRVMOD/GM,REQ,SPIN,AJ2,AJ3,AJ4,AJ5
C      GM = GRAV CONST OF CENTRAL PLANET ( KM**3/S**2)
C      REQ = EQ. RADIUS OF CENTRAL PLANET (KM)
C      SPIN = ROTATION RATE OF CENTRAL PLANET (RAD/SEC)
C      AJ2,AJ3,AJ4,AJ5 = ZONAL HARMONICS OF CENTRAL PLANET
C      SEE BLOCK DATA SUBROUTINE FOR VALUES LOADED.
C
C      SETUP AND INPUT
C
C      Q = CHAR(09)
C      WRITE(*,1010)
C      READ (*,*) OUTFIL
C      WRITE(*,1030)
C      READ(*,*) OUTFLX
C      OPEN (6,FILE=OUTFIL)
C      OPEN (99,FILE=OUTFLX)
1010 FORMAT ('Please input the FILENAME path for UNIT 6 output:')
1030 FORMAT ('Please input the FILENAME path for UNIT 99 output:')
1    WRITE(*,*) 'Enter Elements (a,e,f,w,i,Node) of Spacecraft:'
    READ(*,*) A,E,F,W,AI,OM

```

```

      WRITE(6,444) A,E,F,W,AI,OM
444  FORMAT('INPUT ORBITAL ELEMENTS: (a,e,f,w,i,Node) '
1     / F11.4,F10.7,4F10.4)
C     WRITE(99,*) 'INITIAL S/C ORBIT: ',A,Q,E,Q,F,Q,W,Q,AI,Q,OM
C
      IF (A .LT. 0.D0) GO TO 999
C
C     LOAD ELEMENTS INTO ELMIN
      ELMIN(1) = A
      ELMIN(2) = E
      ELMIN(3) = F/RAD
      ELMIN(4) = W/RAD
      ELMIN(5) = AI/RAD
      ELMIN(6) = OM/RAD
      WRITE(*,*) 'Please enter run control values:'
      WRITE(*,*) 'MNOSC,KPROP,KOUT,KU99,DELT,TF,TS,TCON,NODALP,NREV,J2SQ'
      READ(*,*) MNOSC,KPROP,KOUT,KU99,DELT,TF,TS,TCON,NODALP,NREV,J2SQ
      WRITE(6,*) 'MNOSC,KPROP,KOUT,KU99,DELT,TF,TS,TCON,NODALP,NREV,J2SQ'
      WRITE(6,*) MNOSC,KPROP,KOUT,KU99,DELT,TF,TS,TCON,NODALP,NREV,J2SQ
      KSS = LONG(362) !SECRET CODE FOR GETTING TIME TO 1/60 SEC
C     DO 3 I=3,6
C     3 ELMIN(I)=ELMIN(I)/RAD
C
      IF (KPROP.NE.0) GO TO 4
C
C     ELEMENTS NOT PROPAGATED, CONVERSION OF INPUT ELEMENTS ONLY
C
      CALL KOZSAK(ELMOUT,EM,ELMIN,MNOSC)
      DO 2 I=3,6
      ELMOUT(I) = ELMOUT(I)*RAD
2  CONTINUE
      EM=EM*RAD
      WRITE(6,111) ELMOUT,EM
      WRITE(*,111) ELMOUT,EM
111  FORMAT(1H ,F11.4,F10.7,5F10.4)
C
C     MEAN ELEMENTS PROPAGATED
C
      4 IF (KPROP.EQ.0) GO TO 1
      IF (MNOSC.LT.0) GO TO 5
      DO 6 I=1,6
      6 ELMOUT(I) = ELMIN(I)
      GO TO 7
      5 CALL KOZSAK(ELMOUT,EM,ELMIN,MNOSC)
      7 PBAR= ELMOUT(1)*(1.D0-ELMOUT(2)**2)
      CON = AJ2*REQ*REQ/PBAR/PBAR
      SI= DSIN(ELMOUT(5))
      CI = DCOS(ELMOUT(5))
      SEPS= DSQRT(1.D0-ELMOUT(2)**2)
      FACT = (1.D0- CON*(1.D0-1.5D0*SI*SI)*SEPS)
      ENBAR= DSQRT(GM/ELMOUT(1)**3)
      ENBAR= ENBAR*(1.D0-0.75*CON*SEPS*(1.D0-3.D0*CI*CI))
C
C     NOTE THAT ENBAR IS KOZAI'S MEAN MEAN NOTION BUT THE
C     SEMI-MAJOR AXIS IS NOT MULTIPLIED BY FACT AS IN KOZAI'S
C     1959 PAPER. THE MEAN A HERE IS BROUWER'S A''.ELMT(1).
C
C     NOTE: ELMOUT NOW CONTAINS THE INITIAL MEAN ELEMENTS

```

C

```

ELMT(1)=ELMOUT(1)
ELMT(2)=ELMOUT(2)
ELMT(5) = ELMOUT(5)
OM0 = ELMOUT(6)
W0 = ELMOUT(4)

```

C

C

```

GET INITIAL MEAN MEAN ANOMALY
EM0 = TRMN(1,ELMOUT(3),ELMT(2))
T = TS
GAM4 = AJ4/AJ2**2
IF(J2SQ.NE.0) OD2 = ENBAR*(3.D0/32.D0)*CON*CON*CI
1      *(4.D0-40.D0*CI*CI-(9.D0-5.D0*CI*CI)*ELMT(2)**2
2      + 12.D0*SEPS*(1.D0-3.D0*CI*CI)
3 - 5.D0*GAM4*(3.D0 - 7.D0*CI*CI)*(2.D0 + 3.D0*ELMT(2)**2))
IF(J2SQ.NE.0) WD2 = -ENBAR*0.75D0*CON
1      *CON/32.D0*((10.D0+86.D0*CI*CI)*(1.D0-5.D0*CI*CI)
2      +(25.D0-126.D0*CI*CI+45.D0*CI**4)*ELMT(2)**2
3      -24.D0*SEPS*(1.D0-8.D0*CI*CI+15.D0*CI**4)
4      +20.D0*GAM4*(3.D0-36.D0*CI*CI+49.D0*CI**4)
5      +45.D0*GAM4*(1.D0-14.D0*CI*CI+21.D0*CI**4)*ELMT(2)**2)
IF(J2SQ.NE.0) ENBAR=ENBAR+0.75D0*CON*CON*SEPS/32.D0
1      *((10.D0-60.D0*CI*CI+130.D0*CI**4)
2      -(25.D0-90.D0*CI*CI+25.D0*CI**4)*ELMT(2)**2
3      +16.D0*SEPS*(1.D0-6.D0*CI*CI+9.D0*CI**4)
4      -15.D0*GAM4*(3.D0-30.D0*CI*CI+35.D0*CI**4)*ELMT(2)**2)*ENBAR

```

C

C

C

```

TOP OF LOOP FOR EPHEMERIS GENERATION

```

```

8 ELMT(6) = OM0 - 1.5D0*CON*CI*ENBAR*T
ELMT(4) = W0 + 1.5D0*CON*ENBAR*(2.D0-2.5D0*SI*SI)*T
IF(J2SQ.NE.0) ELMT(4) = ELMT(4) + WD2*T
IF(J2SQ.NE.0) ELMT(6) = ELMT(6) + OD2*T
ELMT(4) = DMOD(ELMT(4),PI2)
ELMT(6) = DMOD(ELMT(6),PI2)
EMT= EM0 + ENBAR*T
EMT = DMOD(EMT,PI2)
ELMT(3) = TRMN(-1,EMT,ELMT(2))
ELMTOT(1) = ELMT(1)
ELMTOT(2) = ELMT(2)
DO 14 I=3,6
14 ELMTOT(I) = ELMT(I)*RAD

```

C

C

C

C

C

```

MEAN ELEMENTS NOW UPDATED

```

```

NOW GET OSCULATING ELEMENTS AT TIME T

```

```

CALL KOZSAK(ELMDUM,EM,ELMT,1)
ELMOUT(1) = ELMDUM(1)
ELMOUT(2) = ELMDUM(2)
DO 9 I=3,6
ELMDUM(I) = DMOD(ELMDUM(I),PI2)
9 ELMOUT(I) = ELMDUM(I)*RAD
TIME= T/TCON
EMTOUT = ELMT(3) + ELMT(4)
EMTOUT=DMOD(EMTOUT,PI2)*RAD

```

C

C

```

ELMOUT NOW HAS OSC. ELEMENTS FOR OUTPUT (KM , DEG)

```



```

C      ELMTOT HAS MEAN ELEMENTS FOR OUTPUT (KM,DEG)
C      EMTOUT IS MEAN ARG. OF LATITUDE, I.E. MEAN W + F (DEG)
C
      IF (KOUT .NE. 0 ) WRITE(6,222) TIME,ELMOUT,ELMTOT,EMTOUT
      IF (KOUT .GT. 0 ) WRITE(*,222) TIME,ELMOUT,ELMTOT,EMTOUT
222  FORMAT(1H , F8.2,F11.4,F9.7,4F10.4/1H ,8X,F11.4,F9.7,5F10.4)
C
C      TAB-DELIMITED OUTPUT TO UNIT 99
C
      IF (KU99 .EQ. 0) GO TO 12
      IF (KU99.EQ.1) WRITE (99,555) TIME,Q,ELMOUT(1),Q,ELMOUT(2),Q,
1      ELMOUT(3),Q,ELMOUT(4),Q,ELMOUT(5),Q,ELMOUT(6)
555  FORMAT(1H ,F8.2,A1,F11.4,A1,F9.7,4(A1,F10.4))
C
      IF (KU99 .LT.2) GO TO 12
C
C      GET POSITION AND VELOCITY FROM OSCULATING ELEMENTS
C
      ELMDUM(1) = ELMDUM(1)*(1.D0-ELMDUM(2)**2)
      CALL ORBIT(3,-2,XOUT,XDOUT,GM,ELMDUM)
      IF (KU99 .EQ.2) WRITE(99,666) TIME,Q,XOUT(1),Q,XOUT(2),Q,
1      ,XOUT(3),Q,XDOUT(1),Q,XDOUT(2),Q,XDOUT(3)
666  FORMAT(1H ,F8.2,3(A1,F10.2),3(A1,F10.6))
      IF (KU99 .EQ. 3) WRITE (99,777) TIME,Q,ELMOUT(1),Q,ELMOUT(2)
1      ,Q,ELMOUT(3),Q,ELMOUT(4),Q,ELMOUT(5),Q,ELMOUT(6),Q,XOUT(1)
2      ,Q,XOUT(2),Q,XOUT(3),Q,XDOUT(1),Q,XDOUT(2),Q,XDOUT(3)
777  FORMAT(1H ,F8.2,A1,F11.4,A1,F9.7,4(A1,F10.4),
1      3(A1,F10.2),3(A1,F10.6))
C
12  T = T + DELT
      IF ( T.LT. TF+.000001) GO TO 8
C
      KSE = LONG(362)
      TTIME = KSE - KSS
      TTIME = TTIME/60.
      WRITE(6,*) ' RUN TIME = ', TTIME, ' SECONDS '
      WRITE(*,*) ' RUN TIME = ', TTIME, ' SECONDS '
      IF (KU99 .NE. 0) WRITE(99,*) ' RUN TIME = ', TTIME, ' SECONDS '
C
      IF(NODALP.EQ.0) GO TO 1
C
C      ITERATE FOR NODAL PERIOD
C
      IDIOT = 0
      TN = PI2*NREV/ENBAR
      WRITE(6,*) ' NODAL PERIOD ITERATION '
      WRITE(*,*) ' NODAL PERIOD ITERATION '
11  DW = 1.5D0*CON*ENBAR*(2.D0-2.5D0*SI*SI)*TN
      IF(J2SQ.NE.0) DW = DW + WD2*TN
      EMB = TRMN(1,-W0,ELMT(2))
      WW = -W0 - DW + PI2*NREV
      EMTOP = TRMN(1,WW,ELMT(2))
      TNN = (EMTOP-EMB)/ENBAR
      DELTN = TNN-TN
      IDIOT = IDIOT + 1
      WRITE(6,333) TNN,TN,DELTN,IDIOT
      WRITE(*,333) TNN,TN,DELTN,IDIOT
333  FORMAT(1H , 3F15.3,I4)

```

```
      IF (IDIOT.GT.10.OR. DABS (DELTN) .LT. 1.D-3) GO TO 1
      TN= TN+DELTN
      GO TO 11
999 STOP
      END
```

C
C

```
      BLOCK DATA
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CONST/ CONST(4)
      COMMON /GRVMOD/ GRVMOD(7)
      DATA CONST /57.29577951308232D0 , 3.141592653589793D0 ,
1 6.283185307179586D0 , 149597893.D0 /
      DATA GRVMOD / 398600.8 , 6378.14 , 7.29212361D-5,
1 1.082D-3 , -2.56D-6, -1.58D-6, 0.D0 /
      END
```

C
C

```
      SUBROUTINE KOZSAK (ELMOUT, EM, ELMIN, MNOSC)
      IMPLICIT REAL*8 (A-H,O-Z)
```

C
C

```
      THIS ROUTINE CALCULATES THE SHORT-PERIODIC VARIATIONS IN
      ORBITAL ELEMENTS DUE TO J2 IN THE MAIN PROBLEM OF THE
      ARTIFICIAL SATELLITE THEORY, (J2 ONLY). THE NAME IS A
      COMBINATION OF KOZAI AND IZSAK FROM WHOSE WORK THE
      FOLLOWING TRANSFORMATION IS DERIVED. THE AUTHOR IS GRATEFUL
      TO PROF. J.V. BREAKWELL FOR THE REFERENCE TO IZSAK'S NOTE.
      C. UPHOFF 1984/APRIL/25
```

C
C

```
      UNITS ARE KILOMETERS, SECONDS, RADIANS.
      COORDINATE FRAME IS TRUE EQUATOR AND EQUINOX OF EPOCH.
```

C
C

```
      MNOSC IS THE FLAG TO TELL WHETHER TO GO FROM MEAN TO
      OSCULATING OR VICE-VERSA. IF MNOSC IS NON-NEGATIVE,
      THE TRANSFORMATION WILL BE FROM MEAN TO OSC, AND
      NO ITERATION WILL BE PERFORMED. IF MNOSC IS NEGATIVE,
      THE TRANSFORMATION IS FROM OSC. TO MEAN AND THIS
      ROUTINE WILL TRY ABS(MNOSC) TIMES TO GET THE MEAN
      ECCENTRICITY TO CONVERGE TO A VALUE WITHIN TOL
      OF THE PREVIOUS ITERATION.
```

C
C

```
      ELMOUT WILL ALWAYS CONTAIN THE TRANSFORMED ELEMENTS,
      EM IS THE MEAN ANOMALY ASSOCIATED WITH ELMOUT,
      ELMIN ARE THE INPUT ELEMENTS AND REMAIN UNCHANGED.
      GOOD LUCK
```

C
C

```
      REFERENCES:  BROUWER, D. 1959 ASTRON. J. VOL. 64, P 378.
                   KOZAI, Y. 1959 IBID. VOL. 64, P 367.
                   KOZAI, Y. 1962 IBID. VOL. 67, P 446.
                   IZSAK, I. 1963 IBID. VOL. 68, P 559.
                   AKSNES, K. 1972 ASTRON. & ASTROPHS. VOL. 17, P 70
```

C
C

```
      DIMENSION ELMOUT(6), ELMIN(6)
      COMMON /CONST/ RAD, PI, PI2, AU
      COMMON /GRVMOD/ GM, REQ, SPIN, AJ2, AJ3, AJ4, AJ5
```

C

```
      DATA TOL/1.D-7/
```

```

C
C
C      AJ2 (J2=-C20) , REQ, AND GM ASSUMED SET IN MAIN OR BLOCK DATA
C
C      NOTE THAT GM IS NOT REQUIRED FOR CONVERSION, ANY VALUE WILL
C      GIVE SAME ANSWERS.....
C
      IDIOT = 1
      SAVE = 1.D10
      DO 1 J=1,6
1     ELMOUT(J) = ELMIN(J)
C
      PIN = ELMIN(1) * (1.D0-ELMIN(2)**2)
      SFIN = DSIN(ELMIN(3))
      CFIN = DCOS(ELMIN(3))
      RIN = PIN/(1.D0 + ELMIN(2)*CFIN)
      RDOTIN = DSQRT(GM/PIN)*ELMIN(2)*SFIN
C
C      GET MEAN ANOMALY FROM TRUE ANOMALY
      EM = TRMN(1,ELMOUT(3),ELMOUT(2))
C
C      TOP OF LOOP FOR OSC. TO MEAN INTERATION (MNOSC < 0)
C
2     P = ELMOUT(1) * (1.D0-ELMOUT(2)**2)
      E= ELMOUT(2)
      U= ELMOUT(4) + ELMOUT(3)
C      SET UP SINES AND COSINES
C
      SEPS = DSQRT(1.D0-E*E)
      RMOP = DSQRT(GM/P)
      SU = DSIN(U)
      CU = DCOS(U)
      C2U = DCOS(U+U)
      S2U = DSIN(U+U)
      SI = DSIN(ELMOUT(5))
      CI = DCOS(ELMOUT(5))
      CON = AJ2*REQ*REQ/P/P
C
C      ADD LONG-PERIODIC AND 2ND ORDER STUFF FROM AKSNES,
C      NOTE CON = GAMA
C
      GAM3 = AJ3/(AJ2**2*REQ/P)
      GAM4 = AJ4/AJ2**2
      A1 = 16.D0*GAM3
      A2 = (1.D0-15.D0*CI*CI + 5.D0*GAM4*(1.D0-7.D0*CI*CI))
1     / (1.D0-5.D0*CI*CI)
      A2PR = (5.D0*A2-15.D0-35.D0*GAM4)/(1.D0 - 5.D0*CI*CI)
C
C      ELEMENTS IN ORDER A,E,F,W,I,NODE,
C      F = TRUE ANOMALY , W=ARG, OF PERIAPSIS
C
      CF = DCOS(ELMOUT(3))
      SF = DSIN(ELMOUT(3))
      CW = DCOS(ELMOUT(4))
      SW = DSIN(ELMOUT(4))
      S2W = 2.D0*SW*CW
      C2W = CW*CW-SW*SW

```

```

C      C3W = DCOS (3.D0*ELMOUT (4) )
C
C      CALC. SINES AND COSINES OF F+2W AND 3F+2W.
C
      SF2W = DSIN (ELMOUT (3) +2.D0*ELMOUT (4) )
      CF2W = DCOS (ELMOUT (3) +2.D0*ELMOUT (4) )
      S3F2W = DSIN (3.D0*ELMOUT (3) +2.D0*ELMOUT (4) )
      C3F2W = DCOS (3.D0*ELMOUT (3) +2.D0*ELMOUT (4) )
C
C      CALC. PERTURBATION IN SEMI-LATUS RECTUM, P
C
      DP= P*1.5D0*CON*SI*SI* (C2U+E*CF2W+E*C3F2W/3.D0)
1      +P* (CON/16.D0) * (-2.D0*A2*SI*SI*E*E*C2W
2      +A1*SI*E*SW)
      DELI = 0.5D0* (DP/P) *CI/SI
      DNODE = -1.5D0*CON*CI* (ELMOUT (3) -EM+E*SF- .5D0*S2U
1      - .5D0*E*SF2W-E*S3F2W/6.D0)
2      - (CON*CI/16.D0) * ( (A2-A2PR*SI*SI) *E*E*S2W
3      +0.5D0*A1*E*CW/SI)
C
C      NOW GET SHORT-PERIODIC VARIATION IN RADIUS
C
      DELR = -P*CON/4.D0* ( (3.D0*CI*CI-1.D0) * (2.D0*SEPS/ (1.D0+E*CF)
1      +E*CF/ (1.D0+SEPS) +1.D0) -SI*SI*C2U)
2      + (P*CON/32.D0) * (-2.D0*A2*SI*SI*E*CF2W
3      + A1*SI*SU)
C
C      CALC. SHORT PERIODIC VARIATION IN RADIAL SPEED
C
      DRDOT = 0.25D0*CON*RMOP* ( (3.D0*CI*CI-1.D0) *E*SF
1      * (SEPS+ (1.D0+E*CF) **2/ (1.D0+SEPS) )
2      -2.D0*SI*SI* (1.D0+E*CF) **2*S2U)
3      - (CON*DSQRT (GM*P) *P/32.D0/RIN/RIN) *
4      (-2.D0*A2*SI*SI*E*SF2W - A1*SI*CU)
C
C      GET VARIATION IN ARG. OF LATITUDE
C
      DELU = -CON/8.D0* (6.D0* (1.D0-5.D0*CI*CI) * (ELMOUT (3) -EM)
1      +4.D0* ( (1.D0-6.D0*CI*CI) + (1.D0-3.D0*CI*CI) /
2      (1.D0+SEPS) ) *E*SF + (1.D0-3.D0*CI*CI) / (1.D0+SEPS)
3      *E*E*2.D0*SF*CF +2.D0* (5.D0*CI*CI-2.D0) *E*SF2W
4      + (7.D0*CI*CI-1.D0) *S2U + 2.D0*CI*CI*E*S3F2W)
5      -CON/32.D0* (2.D0* (A2PR*SI*SI-A2) *CI*CI*E*E*S2W
6      -4.D0*A2*SI*SI*E*SF2W
7      -2.D0*A1*SI*CU - A1* (CI*CI/SI/SI - 1.5D0) *SI*E*CW
7      -A1/2.D0*SI*E*DCOS (U+ELMOUT (3) ) )
C
C      PUT 'EM TOGETHER AND WHATTA YA GOT?
C
C
C      BIBBIDY
C
      FLIP = MNOSC
      IF (MNOSC .EQ. 0) FLIP = 1.D0
      FLIP = FLIP/DABS (FLIP)
      POUT = PIN + FLIP*DP
      ROUT = RIN + FLIP * DELR
      RDTOUT = RDOTIN + FLIP * DRDOT

```

```

C
C      BOBBIDY
C
      BIGA = POUT/ROUT - 1.D0
      BIGB = RDTOUT/DSQRT(GM/POUT)
      EOUT = DSQRT(BIGA**2 + BIGB**2)
      FOUT = DATAN2(BIGB,BIGA)
C
C      BOO
C
      UOUT= ELMIN(4) + ELMIN(3) + FLIP*DELU
      ELMOUT(1) = POUT/(1.D0-EOUT**2)
      ELMOUT(2) = EOUT
      ELMOUT(5) = ELMIN(5) + FLIP*DELI
      ELMOUT(6) = ELMIN(6) + FLIP * DNODE
      ELMOUT(4) = UOUT -FOUT
      ELMOUT(3) = FOUT
      EM = TRMN(1,ELMOUT(3),ELMOUT(2))
C
C      HERE THEY COME
C
      TEST = DABS(EOUT-SAVE)
      SAVE = EOUT
      IF(MNOSC.GE.-1) RETURN
C
C      THERE THEY GO
C
      IF (MNOSC.LT. 0 .AND. TEST.LT.TOL) GO TO 998
      IF(IDIOT.EQ.ABS(MNOSC)) GO TO 999
      IDIOT = IDIOT + 1
      GO TO 2
999 WRITE (6,111) TEST,DRDOT,DELR
111 FORMAT(1H , 'OSCULATING TO MEAN ITERATION FAILS, TEST = '3G12.4)
998 RETURN
      END
      SUBROUTINE CROSS (X,Y,Z)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(3),Y(3),Z(3),DUM(3)
      DUM(1) = X(2)*Y(3) - X(3)*Y(2)
      DUM(2) = X(3)*Y(1) - X(1)*Y(3)
      DUM(3) = X(1)*Y(2) - X(2)*Y(1)
      DO 1 J = 1,3
1 Z(J) = DUM(J)
      RETURN
      END
C
C
      DOUBLE PRECISION FUNCTION DARCOS(X)
      DOUBLE PRECISION X
      DARCOS = DACOS(X)
      RETURN
      END
C
C
      DOUBLE PRECISION FUNCTION DARSIN(X)
      DOUBLE PRECISION X
      DARSIN = DASIN(X)
      RETURN

```

```

      END
C
C
      DOUBLE PRECISION FUNCTION DOT (X,Y)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(3),Y(3)
      DOT = X(1)*Y(1) + X(2)*Y(2) + X(3)*Y(3)
      RETURN
      END
C
C
      DOUBLE PRECISION FUNCTION DVMAG(X)
      IMPLICIT REAL*8 (A-H,O-Z)
C.....FUNCTION DVMAG CALCULATES THE MAGNITUDE OF A DOUBLE PRECISION
C      VECTOR
      DIMENSION X(3)
      DVMAG = DSQRT(X(1)**2 + X(2)**2 + X(3)**2 )
      RETURN
      END
C
C
      SUBROUTINE INRTRT(ROT,PHI)
      IMPLICIT REAL*8 (A-H,O-Z)
C      SETS UP ROT. MAT. FOR XFM FROM INERTIAL TO ROTATING FRAME
      DIMENSION ROT(3,3)
      CPHI = DCOS(PHI)
      SPHI = DSIN(PHI)
      ROT(1,1) = CPHI
      ROT(1,2) = SPHI
      ROT(1,3) = 0.D0
      ROT(2,1) = -SPHI
      ROT(2,2) = CPHI
      ROT(2,3) = 0.D0
      ROT(3,1) = 0.D0
      ROT(3,2) = 0.D0
      ROT(3,3) = 1.D0
      RETURN
      END
C
C
      SUBROUTINE MVTRN(A,B,C,M,N)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(9),B(1),C(1)
      N3 = 3*N
      DO 5 I=1,N3,3
      B1 = B(I)
      B2 = B(I+1)
      B3 = B(I+2)
      IF (M .NE. 1) GO TO 3
C      C = A B
      2 D1 = A(1)*B1 + A(4)*B2 + A(7)*B3
      D2 = A(2)*B1 + A(5)*B2 + A(8)*B3
      D3 = A(3)*B1 + A(6)*B2 + A(9)*B3
      GO TO 4
C      C = A(TRANPOSED) B
      3 D1 = A(1)*B1 + A(2)*B2 + A(3)*B3
      D2 = A(4)*B1 + A(5)*B2 + A(6)*B3
      D3 = A(7)*B1 + A(8)*B2 + A(9)*B3

```

```

4 C(I) = D1
  C(I+1) = D2
  C(I+2) = D3
5 CONTINUE
  RETURN
  END

C
C
  SUBROUTINE ORBIT(K,J,X,XD,GM,ELM)
  IMPLICIT REAL*8 (A-H,O-Z)
C      REPLACES ORBIT3-5-6 FOR K=3, 5 OR 6
C      ORBIT5 MUST SPLIT X INTO X,X(4) IN ARGUMENT LIST FOR ORBIT
  DIMENSION X(3),XD(3),ELM(6)
  COMMON /CONST/ RAD,PI,PI2,AU
  IF (J .LT. 0) GO TO 111

C
C      GET ELEMENTS FROM POSITION AND VELOCITY
C
11 R= DSQRT(X(1)*X(1) + X(2)*X(2) + X(3)*X(3))
  RDOT = (X(1)*XD(1) +X(2)*XD(2)+X(3)*XD(3)) / R
  WXP = X(2)*XD(3) - X(3)*XD(2)
  WYP = X(3)*XD(1) - X(1)*XD(3)
  WZP = X(1)*XD(2) - X(2)*XD(1)
  P = (WXP*WXP + WYP*WYP + WZP*WZP) / GM
  H = DSQRT(P*GM)
  SU = RDOT * P / H
  CU = (P - R)/R
  E = DSQRT(CU**2 + SU**2)
  WZ = WZP/H
  IF (DABS(WZ) .GE. 1.D0) GO TO 2
1 CI = DARCOS(WZ)
  O = DATAN2(WXP, (-WYP))
  SN = X(3)*H*(1.0D0-WZ*WZ) - WZ*(X(1)*WXP + X(2)*WYP)
  CN = X(2)*WXP - X(1)*WYP
  UN = DATAN2(SN,CN)
  GO TO 3
2 CI = DATAN2(0.D0,WZ)
  O = 0.0D0
  UN = DATAN2((X(2)*WZ), X(1))
3 IF (E .LE. 2.D-7) GO TO 7
4 U = DATAN2(SU,CU)
  W = UN - U
  IF (W .LT. 0.D0) W = W + PI2
  GO TO 8
7 W = 0.0D0
  U = UN
8 ELM(1) = P
  ELM(5) = CI
  ELM(6) = O
  IF (K .NE. 6) GO TO 300
  ELM(2) = E * DSIN(W)
  ELM(3) = E * DCOS(W)
  ELM(4) = W + U
  RETURN
300 ELM(2) = E
  ELM(3) = U
  ELM(4) = W
  IF (K .LT. 4) RETURN

```

```

      ELM(7) = DCOS(W)
      ELM(8) = DSIN(W)
      ELM(9) = DCOS(CI)
      ELM(10) = DSIN(CI)
      ELM(11) = DCOS(O)
      ELM(12) = DSIN(O)
      RETURN
C
C   GET POSITION AND VELOCITY (J = -2) FROM ELEMENTS
C   (J = -1) RETURNS POSITION ONLY
C
111  IF (K - 5) 333,555,666
555  CW = ELM(7)
      SW = ELM(8)
      CINC = ELM(9)
      SINC = ELM(10)
      CO = ELM(11)
      SO = ELM(12)
      GO TO 334
333  CW = DCOS(ELM(4))
      SW = DSIN(ELM(4))
666  CINC = DCOS(ELM(5))
      CO = DCOS(ELM(6))
      SINC = DSIN(ELM(5))
      SO = DSIN(ELM(6))
      IF (K .GT. 5) GO TO 667
334  E = ELM(2)
      U = ELM(3)
      GO TO 668
667  E = DSQRT(ELM(2)**2 + ELM(3)**2)
      W = DATAN2(ELM(2), ELM(3))
      SW = ELM(2)/E
      CW = ELM(3)/E
      U = ELM(4) - W
668  P = ELM(1)
      CU = DCOS(U)
      SU = DSIN(U)
      A11 = CW * CO - CINC*SO * SW
      A12 = -SW*CO - CINC*SO * CW
      A21 = CW*SO + CINC*CO * SW
      A22 = -SW*SO + CINC*CO * CW
      A31 = SINC * SW
      A32 = SINC * CW
      IF(J.EQ.-3) GO TO 66
      R = ELM(1)/(1.D0 + E*CU)
      XPR = R*CU
      YPR = R*SU
      X(1) = A11*XPR + A12*YPR
      X(2) = A21*XPR + A22*YPR
      X(3) = A31*XPR + A32*YPR
      IF (J .EQ. -1) RETURN
12  V = DSQRT(GM * (2.D0/R - (1.D0 - E*E)/ELM(1)))
      GAMA = DATAN(E*SU/(1.D0 + E*CU))
67  VXPR = -V * DSIN(U-GAMA)
      VYPR = V * DCOS(U-GAMA)
      XD(1) = A11*VXPR + A12*VYPR
      XD(2) = A21*VXPR + A22*VYPR
      XD(3) = A31*VXPR + A32*VYPR

```



```

13 RETURN
66 V = DSQRT(GM*(E**2 - 1.D0)/ELM(1))
   GAMA = PI2/4.D0
   GO TO 67
   END
C
C
   SUBROUTINE ROTATE (M,A,B,C)
   IMPLICIT REAL*8 (A-H,O-Z)
   DIMENSION A(9),B(3),C(3)
C   ROTATE DOES MATRIX-VECTOR MULT
   IF (M .NE. 1) GO TO 3
C   C = A B
2 C(1) = A(1)*B(1) + A(4)*B(2) + A(7)*B(3)
  C(2) = A(2)*B(1) + A(5)*B(2) + A(8)*B(3)
  C(3) = A(3)*B(1) + A(6)*B(2) + A(9)*B(3)
  GO TO 4
C   C = A(TRANPOSED) B
3 C(1) = A(1)*B(1) + A(2)*B(2) + A(3)*B(3)
  C(2) = A(4)*B(1) + A(5)*B(2) + A(6)*B(3)
  C(3) = A(7)*B(1) + A(8)*B(2) + A(9)*B(3)
4 CONTINUE
  RETURN
  END
C
C
   DOUBLE PRECISION FUNCTION TRMN (J,QQ,E)
   IMPLICIT REAL*8 (A-H,O-Z)
   COMMON /CONST/ RAD,PI,PI2,AU
   F = QQ
   D = DABS(1.D0-E)
   U = F
   IF(J)2,1,1
1 IF (E.GT.1.D0) GO TO 20
  CF = DCOS(F)
  IF (DABS(F).LT. 1.D-8) CF = 1.D0 - F*F/2.D0 + F**4/24.D0
  ECAN = DARCOS((E + CF)/(1.D0 + E * CF))
  MTST = F/PI
  TST = MTST
  ONE = 1.D0
  IF (F .LT. 0.D0) ONE = -ONE
  MTST = MOD(MTST,2)
  IF (MTST .NE. 0) ECAN = TST * PI + ONE*(PI-ECAN)
  IF (MTST .EQ. 0) ECAN = TST * PI + ONE*ECAN
  TRMN = ECAN - E * DSIN(ECAN)
  RETURN
20 ULIM = PI + DARCOS(1.D0/E)
  BLIM = PI2 - ULIM
  IF(F.GE.BLIM.AND.F.LE.ULIM) GO TO 21
  FARG = DSQRT((E - 1.D0)/(E + 1.D0)) * DSIN(F/2.D0)/DCOS(F/2.D0)
  FAN = DLOG(1.D0 + FARG) - DLOG(1.D0 - FARG)
  TRMN = -FAN + E * DSINH(FAN)
  RETURN
21 WRITE(6,111)
111 FORMAT(1H ,26HTRUE ANOMALY NOT POSSIBLE )
   TRMN = 0.0D0
   RETURN
2 IF(E-1.D0) 5,6,7

```

```

5  U = F
   DO 8 M=1,50
     DEM = F - U + E * DSIN(U)
     ECOSU = E * DCOS(U)
     DELU = DEM / (1.D0 - ECOSU + .01D0*ECOSU**3)
     IF (ECOSU .GT. .99D0 .OR. M .GT. 10 ) DELU = DEM / (1.D0 - ECOSU)
     IF (DABS(DELU) .GT. 2.D0) DELU=DSIGN(2.D0,DELU)
     U = U + DELU
     IF ( DABS(DEM) - 2.D-12) 9,9,8
8  CONTINUE
   WRITE (6,119) QQ,E,DEM
119 FORMAT (23H TRMN FAILS, E.LT. 1.0      3D10.4 )
9  COSU = DCOS(U)
   DEM1 = 1.D0 - E*COSU
   CS = (COSU -E) /DEM1
   SS = DSQRT(1.D0 -E*E) * DSIN(U) /DEM1
   GO TO 99
6  E = E + .0001D0
   GO TO 2
7  U = F/D
   IF (D.LE.DABS(F) ) U=DSIGN( (6.D0*DABS(F) ) ** (1.D0/3.D0) ,F)
   IF (DABS(F) .GT.1.D0) U=DSIGN(DLOG(1.D0+DABS(2.D0*F/E) ) ,F)
   DO 12 L=1,100
     DEM = F + U - E * DSINH(U)
     ECOSU = E * DCOSH(U)
     DELU = -DEM / (1.D0-ECOSU)
     U = U + DELU
     IF (DABS(DEM) - 2.D-12) 11,11,12
12  CONTINUE
   WRITE (6,120) QQ,E,DEM
120 FORMAT(23H TRMN FAILS, E.GT. 1.0      3D10.4 )
11  COSU = DCOSH(U)
   DEM1 = 1.D0 - E*COSU
   CS = (COSU - E) / DEM1
   SS = - DSQRT(E*E - 1.D0) * DSINH(U) /DEM1
99  TRMN = DATAN2(SS,CS)
   RETURN
   END

```

Errata**for****Orbit Propagator for Close Orbiters in an Oblate Field****by****C. Uphoff****Memorandum dated 1988 Dec 16**

p. 4 line 12 of section entitled Osculating and Mean Elements

the phrase "the subtract " should read "then subtract"

p. 11 add the following two lines to the top of the page

obtained by Aksnes⁵ with the exception of a long term 2nd order effect in mean mean anomaly that is shown as a linear growth of the error in

p. 14 line 9 (part of the Fortran listing)

T^*TCON should read $T/TCON$