```
C      TEST FOR TANGENTIAL THRUST
C
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON/STEP/X,X0,H,H0,ERRC,GAIN,STMIN,STMAX,KER,KCYCL
       COMMON/INTVAR/Y(20,6)
       COMMON/RATES/YDOT(20,6)
       COMMON/SAVE/Y0(20,6),YDOT0(20,6),TE(20,6)
       COMMON/CONST/ RAD,GM
       COMMON/FORCE/ T0,TDOT,C0,CDOT,ACC0,EM0
       EQUIVALENCE (Y(1,1),P),(Y(1,2),E),(Y(1,3),F),(Y(1,4),W),
      1 (Y(1,5),DELV),(Y(1,6),EM)
       EQUIVALENCE (YDOT(1,1),PDOT),(YDOT(1,2),EDOT),(YDOT(1,3),FDOT)
      1 ,(YDOT(1,4),WDOT),(YDOT(1,5),ACC),(YDOT(1,6),EMDOT)
       OPEN(UNIT=99,FILE='BURNOUT')
       RAD = 57.2957795D0
       GM = 398600.44D0
C2345678901234567890123456789012345678901234567890123456789012345678901 2
     1 WRITE(9,888)
   888 FORMAT(1H ,'ENTER P,E,F,W,ERRC,GAIN,H,TMAX,DTP,T0,TDOT,C0,CDOT,
      1 EM0')
       READ(9,*) P0,E0,F0,W0,ERRC,GAIN,H,TMAX,DTP,T0,TDOT,C0,CDOT,EM0
       RP0 = P0/(1.D0 + E0)
       WRITE(99,777) P0,E0,F0,W0,ERRC,GAIN,H,TMAX,DTP,T0,TDOT,
      1 C0,CDOT,EM0,RP0
   777 FORMAT(1H ,6G14.6)
       X = 0.D0
       Y(1,1) = P0
       Y(1,2) = E0
       Y(1,3) = F0/RAD
       Y(1,4) = W0/RAD
       Y(1,5) = 0.0D0
       Y(1,6) = EM0
       C30 = -GM*(1.D0-E0*E0)/P0
       VP0 = DSQRT(C30 + 2.D0*GM/RP0)
       TP = 0.D0
       KOUNT = 0
       IF(ERRC.LT.0.D0) GO TO 999
       KSS = LONG(362) !SECRET CODE FOR GETTING TIME TO 1/60 SEC
       STMIN = .0001D0
       STMAX = 10.D0
     2 CALL BRNEQNS(1,6)
       CALL RKHULL(1,6)
       KOUNT = KOUNT+1
       IF(DABS(X).GT.DABS(TMAX).OR. KER .NE. 0 ) GO TO 3
       IF(DABS(X) .LT. DABS(TP)) GO TO 2
       TP = TP + DTP
       C3 = GM*(E*E - 1.D0)/P
C
C      GET  IMPULSIVE DELTA-V AT ORIGINAL PERIGEE FOR CURRENT C3.
C
C2345678901234567890123456789012345678901234567890123456789012345678901 2
       VP = DSQRT(C3 + 2.D0*GM/RP0)
       DELV0 = (DABS(VP) - VP0)*1000.D0
       DELDLV = (DELV*1000.D0 - DELV0)
       DELVOUT = DELV*1000.D0
       ACCOUT = ACC*1000.D0
       FOUT = F*RAD
       fout = dmod(fout,360.d0)
       WOUT = W*RAD
       wout = dmod(wout,360.d0)
       T = T0 + TDOT*X
       WRITE(99,111) X,char(9), P,char(9),E,char(9),FOUT,char(9),WOUT
      1 ,char(9),ACCOUT,CHAR(9),EM,CHAR(9),DELVOUT,char(9),C3,char(9),
      2   DELDLV,char(9),DELV0,char(9),T,char(9),KOUNT
       WRITE(9,111) X,char(9), P,char(9),E,char(9),FOUT,char(9),WOUT
      1 ,char(9),ACCOUT,CHAR(9),EM,CHAR(9),DELVOUT,char(9),C3,char(9),
      2   DELDLV,char(9),DELV0,char(9),T,char(9),KOUNT
   111 FORMAT(1H ,F8.1,A1,F10.2,A1,F8.4,A1,4(F8.3,A1),5(F9.3,A1),I5)
C     WRITE(99,222) H,TE(1,1),TE(1,2)
   222 FORMAT(1H ,38X3G12.4)
       GO TO 2
     3 KSE = LONG(362)
       TTIME = KSE-KSS
       TTIME = TTIME/60.D0
       WRITE(9,666) TTIME,KOUNT
       WRITE(99,666) TTIME,KOUNT
   666 FORMAT(1H ,'RUNTIME = ',F8.2,' SEC    STEPS = ',I5)
       GO TO 1
   999 CONTINUE
       CLOSE(99)
       STOP
       END
C
C
       SUBROUTINE BRNEQNS(NPT,NEQ)
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON/STEP/X,X0,H,H0,ERRC,GAIN,STMIN,STMAX,KER,KCYCL
       COMMON/INTVAR/Y(20,6)
       COMMON/RATES/YDOT(20,6)
       COMMON/SAVE/Y0(20,6),YDOT0(20,6),TE(20,6)
       COMMON/CONST/ RAD,GM
       COMMON/FORCE/ T0,TDOT,C0,CDOT,ACC0,EM0
       EQUIVALENCE (Y(1,1),P),(Y(1,2),E),(Y(1,3),F),(Y(1,4),W),
      1 (Y(1,5),DELV),(Y(1,6),EM)
       EQUIVALENCE (YDOT(1,1),PDOT),(YDOT(1,2),EDOT),(YDOT(1,3),FDOT)
      1 ,(YDOT(1,4),WDOT),(YDOT(1,5),ACC),(YDOT(1,6),EMDOT)
C
       EMDOT = -DABS((T0 + TDOT)/(C0 + CDOT))
       ACC = (T0 + TDOT*X)/(EM0 + EMDOT*X)/1000.D0
       CF = DCOS(F)
       SF = DSIN(F)
       R = P/(1.D0 + E*CF)
       V = DSQRT(GM*(1.D0 + E*E + 2.D0*E*CF)/P)
       PDOT = 2.D0*P*ACC/V
       EDOT = 2.D0*(E + CF)*ACC/V
       FDOT = DSQRT(GM*P)/R/R - 2.D0*SF*ACC/V/E
       WDOT = 2.D0*SF*ACC/V/E
       RETURN
       END
C
C
       SUBROUTINE RKHULL(NPT,NEQ)
       IMPLICIT REAL*8(A-H,O-Z)
C
```

```
C     THIS ROUTINE IS A FOURTH-ORDER RUNGE-KUTTA INTEGRATOR
C     WITH AN EMBEDDED SECOND-ORDER SOLUTION USED FOR AUTOMATIC
C     STEP CONTROL.  THE METHOD APPEARS IN THE AIAA JOURNAL
C     OCT. 1977 IN A NOTE BY DAVID G. HULL, UNIV. OF TEXAS,AUSTIN.
C
C     THE CODE BELOW PERMITS THE SIMULTANEOUS INTEGRATION OF
C     NPT SETS OF NEQ FIRST-ORDER ORDINARY DIFFERENTIAL EQUATIONS.
C
C     USE AS FOLLOWS:
C
C     ERRC .... ERROR CONTROL, TOLERANCE ON LOCAL ALLOWABLE
C               RELATIVE TRUNCATION ERROR.  IF ERRC = 0.D0,
C               A FIXED STEP, H,IS USED.
C     GAIN .... MULTIPLIER OF AUTOMATICALLY SELECTED STEP, ADJUST
C               TO CONTROL NO. OF REPEATED STEPS.
C
C     STMIN,STMAX .....  MIN. AND MAX ALLOWABLE STEP SIZE.
C
C     H   ..... CURRENT STEP SIZE
C
C     H0 .....  STORED VALUE OF STEP AT THE BEGINNING OF A CYCLE.
C
C     X0 .....  VALUE OF THE INDEPENDENT VARIABLE AT START OF
C               CURRENT STEP.
C     X ......  VALUE OF THE INDEPENDENT VARIABLE.  FOR USE
C               DURING EVALUATION OF ALL DERIVATIVES.  UPDATED
C               ACCORDING TO RUNGE-KUTTA FORMULA AND COMMON
C               TO ALL ROUTINES THAT NEED IT TO EVALUATE THE
C               DERIVATIVES.
C     KER ....  ERROR FLAG,SET TO 1 FOR STEP TOO SMALL OR TOO
C               MANY CYCLES IN AUTO STEP MODE.
C
C     KCYCLE .. COUNTER ON NO. OF STEP REDUCTIONS
C
C     ALL VARIABLES ABOVE ARE STORED IN /STEP/ COMMON
C
C     /INTVAR/ COMMON CONTAINS THE DEPENDENT VARIABLES Y(NPT,NEQ)
C
C     /RATES/ COMMON CONTAINS THE DERIVATIVES OF Y WITH RESPECT TO
C             THE INDEPENDENT VARIABLE X. DERIVATIVES ARY YDOT(NPT,NEQ)
C             AND ARE ASSUMED TO BE CALCULATED AS FUNCTIONS OF Y AND X
C             BY A SINGLE CALL TO SUBROUTINE BRNEQNS(NPT,NEQ).
C     /SAVE/ COMMON CONTAINS THE INITIAL STATE AND THE CURRENT VALUES OF
C            THE ESTIMATED TRUNCATION ERRORS.
C
C      NOTE .... IT IS ASSUMED THAT YDOT IS CONSISTENT WITH Y AT ENTRY,
C                THE LAST CALL TO BRNEQNS IN THIS CODE WILL UPDATE YDOT FOR
C                THE NEXT STEP BUT THE USER SHOULD MAKE SURE THAT YDOT IS
C                PROPERLY INITIALIZED BY A CALL TO BRNEQNS BEFORE STARTING AN
C                INTEGRATION.
C
C                                       C. UPHOFF     BASG 88/4/4
C
C
C
C
C
```

```
C
      DIMENSION YS(20,6)
C
      COMMON/STEP/X,X0,H,H0,ERRC,GAIN,STMIN,STMAX,KER,KCYCL
      COMMON/INTVAR/Y(20,6)
      COMMON/RATES/YDOT(20,6)
      COMMON/SAVE/Y0(20,6),YDOT0(20,6),TE(20,6)
      DATA YMIN/.0001D0/
C
      KER = 0
C     DEFAULT VALUES
      IF (GAIN .EQ.0.D0) GAIN = .9D0
      IF(STMIN .EQ. 0.D0) STMIN = 1.D0
      IF(DABS(H).LT.STMIN) GO TO 11
      IF (STMAX.EQ. 0.D0) STMAX = 500.D0
C
C     SAVE INITIAL STATE
C
      DO 1 I = 1,NPT
      DO 1 J = 1,NEQ
      TE(I,J) = 0.D0
      Y0(I,J) = Y(I,J)
    1 YDOT0(I,J) = YDOT(I,J)
      X0 = X
      KCYCL = 0
C
C     CALCULATE THE F SUB K'S
C
    2 CONTINUE
      H0 = H
      DO 3 I = 1,NPT
      DO 3 J = 1,NEQ
      YS(I,J) = Y0(I,J) + H*YDOT(I,J)/6.D0
      TE(I,J) = H*YDOT(I,J)/6.D0
    3 Y(I,J) = Y0(I,J) + 0.5D0*H*YDOT(I,J)
      X = X0 + 0.5D0*H
C
      CALL BRNEQNS(NPT,NEQ)
C
      DO 4 I = 1,NPT
      DO 4 J = 1,NEQ
      YS(I,J) = YS(I,J) + H*YDOT(I,J)/3.D0
    4 Y(I,J) = Y0(I,J) + 0.5D0*H*YDOT(I,J)
C
      CALL BRNEQNS(NPT,NEQ)
C
      X = X0 + H
      DO 5 I = 1,NPT
      DO 5 J = 1,NEQ
      YS(I,J) = YS(I,J) + H*YDOT(I,J)/3.D0
      TE(I,J) = TE(I,J) - H*YDOT(I,J)/3.D0
    5 Y(I,J) = Y0(I,J) + H*YDOT(I,J)
C
      CALL BRNEQNS(NPT,NEQ)
C
C     FOURTH-ORDER SOLUTION
C
```

```
      DO 6 I = 1,NPT
      DO 6 J = 1,NEQ
      TE(I,J) = TE(I,J) + H*YDOT(I,J)/6.D0
    6 Y(I,J) = YS(I,J) + H*YDOT(I,J)/6.D0
C
C
C     AUTOMATIC STEP CALCULATIONS
C
      IF (ERRC.NE.0.D0) GO TO 12
      CALL BRNEQNS(NPT,NEQ)
      RETURN
C
C
   12 TEM = 0.D0
      DO 8 I = 1,NPT
      DO 8 J = 1,NEQ
      YTEST = DMAX1(DABS(Y(I,J)),YMIN)
      TEST = DABS(TE(I,J)/YTEST)
      IF(TEST.LT.TEM) GO TO 8
      TEM = TEST
    8 CONTINUE
      ARG = ERRC/TEM
      H = H*(ARG)**(0.3333333333333D0)
C
C     STEP ACCEPTANCE TEST
C
      IF(TEM .GT. 3.D0*ERRC) GO TO 9
      CALL BRNEQNS(NPT,NEQ)
C
C     LIMITING
      H = H*GAIN
      IF( DABS(H) .GT. STMAX) H = DSIGN(STMAX,H)
      RETURN
C
C     STEP TOO BIG ... RECYCLE
C
    9 IF(DABS(H) .LT. STMIN) GO TO 11
      KCYCL= KCYCL + 1
      IF (KCYCL .GT. 10) GO TO 11
C     RESET STATE
      DO 10 I = 1,NPT
      DO 10 J = 1,NEQ
      TE(I,J) = 0.D0
      Y(I,J) = Y0(I,J)
   10 YDOT(I,J) = YDOT0(I,J)
      X = X0
      GO TO 2
C
C     ERROR RETURN
C
   11 WRITE(9,111) H,KCYCL
  111 FORMAT(1H ,'RKHULL ERROR--STEP TOO SMALL OR TOO MANY CYCLES'/
     1  1H ,'H = ' G12.6, 'KCYCL = ' I3)
      KER = 1
      RETURN
      END
```