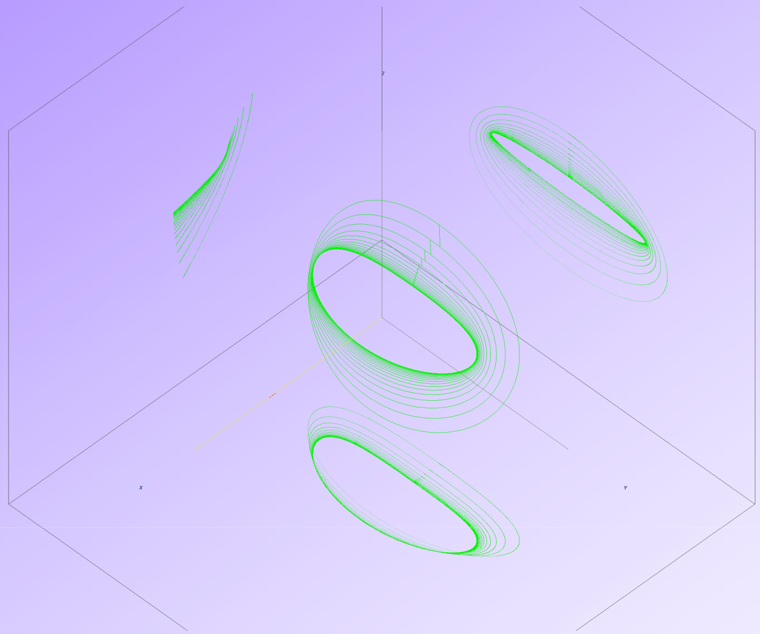
6th ICATT
New Tool for Finding Periodic Halo Orbits: the Solver of a Spacecraft Simulator
(ESPSS -Ecosimpro® European Space Propulsion System Simulation)

Darmstadt, 14th to 17th March 2016



Christophe R. Koppel

KopooS Consulting Ind., 75008 Paris, France

Summary

- Introduction
- The differential system
- Solution for periodic orbits
- Application to Halo orbits
- Conclusions

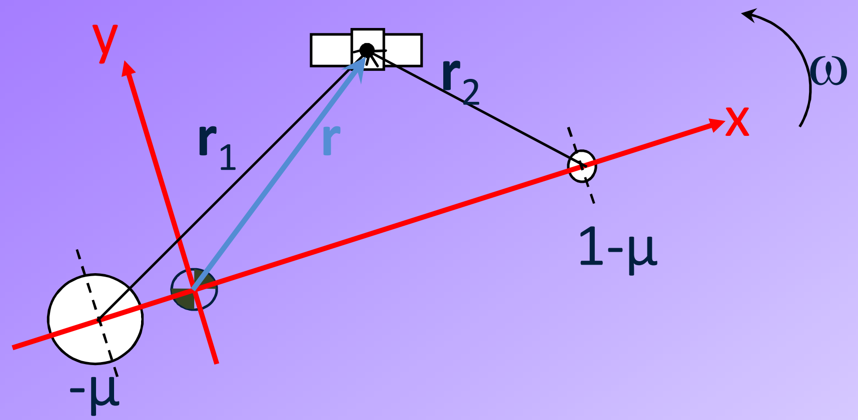
1. New tool periodic Halo, ICATT 6th , 14-17 March 2016, Darmstadt, Germany

Introduction

- The existing ESA developed tool EcosimPro® is a solver of differential algebraic equations
  - It is oriented for system simulations, not for mathematicians, nor for numerical analysts
  - But it can be used as well for solving some problem which are generally solved “only” with some **US tools**
- For engineers, it is sometime needed to assess some orbits
  - But the cost of a sub-contract to do this assessment can be simply out of the scope of normal engineering work
  - In addition time is needed by sub-contractors, and their answers may not be in line with the need (Very low cost)
- Hence, its has been found appropriate to check if a simple EcosimPro model could be used for solving efficiently the question of periodic orbits
  - Because Engineers use preferably EcosimPro, its is not out of the scope of their knowledge
  - But the most difficult part is to get the right equations
  - Unfortunately, this requires some efforts in order to clear the uncertainties in the set of equations
  - In addition high accuracy needed for the numerical resolution can be considered as a showstopper
- The problem will be presneted in simple word
- And the presentation will show the successful Halo orbit solutions

The differential system

- **Acceleration in the rotating frame with** $\mu = M_{earth}/M_{total}$



$$\ddot{\vec{r}} + \vec{\omega}\times\left(\vec{\omega}\times\vec{r}\right) + 2\vec{\omega}\times\dot{\vec{r}} = -\frac{(1-\mu)}{r_1^3}\vec{r}_1 - \frac{\mu}{r_2^3}\vec{r}_2$$

- **➔ lead to write the system straightforward when the non-dimensional rotation rate** $\omega = 1$
- **System surprisingly so simple**

$$\dot{X} = f(X)$$

$$X = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \quad f(X) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ x + 2\dot{y} - \frac{(1-\mu)(x+\mu)}{r_1^3} - \mu\frac{(x-(1-\mu))}{r_2^3} \\ y - 2\dot{x} - \frac{(1-\mu)y}{r_1^3} - \mu\frac{y}{r_2^3} \\ -\frac{(1-\mu)z}{r_1^3} - \mu\frac{z}{r_2^3} \end{bmatrix}$$

$$r_1 = \begin{bmatrix} x-(-\mu) \\ y \\ z \end{bmatrix} \quad \left\|r_1\right\| = \sqrt{(x-(-\mu))^2 + y^2 + z^2}$$

$$r_2 = \begin{bmatrix} x-(1-\mu) \\ y \\ z \end{bmatrix} \quad \left\|r_2\right\| = \sqrt{(x-(1-\mu))^2 + y^2 + z^2}$$

Solution for Periodic Orbits

$$X(0)=\begin{bmatrix}x_0\\0\\z_0\\0\\\dot{y}_0\\0\end{bmatrix}\quad X(T\tfrac{1}{2})=\begin{bmatrix}x\\0\\z\\0\\\dot{y}\\0\end{bmatrix}$$

Principles

- **Without deep analysis, simple periodic orbits cannot be unsymmetrical**
  - **In the plane XZ for example ➔After half orbit some state variable are the same**
- **Problem reformulated (x_0 fixed)**
  - **Search problem of the 3 zeros with** $g(Y)=0$
  - **Iteration by Newton Method**

$$Y=\begin{bmatrix}z_0\\\dot{y}_0\\T_{1/2}\end{bmatrix}\quad g(Y)=\begin{bmatrix}y(t)\\\dot{x}(t)\\\dot{z}(t)\end{bmatrix}_{t=T_{1/2}}$$

$$Y_{n+1}=Y_n-\left[\frac{\partial g}{\partial Y_{|Y=Y_n}}\right]^{-1}.g(Y_n)$$

  - **But that *Jacobian* becomes the real problem to solve…**

Solution for Periodic Orbits

Principles

- Search problem of the 3 zeros with $g(Y) = 0$ $\quad Y = \begin{bmatrix} z_0 \\ \dot{y}_0 \\ T_{1/2} \end{bmatrix} \quad g(Y) = \begin{bmatrix} y(t) \\ \dot{x}(t) \\ \dot{z}(t) \end{bmatrix}_{t=T_{1/2}}$
  - But that *Jacobian* becomes the real problem to solve…

$$Y_{n+1} = Y_n - \left[\frac{\partial g}{\partial Y}_{|Y=Y_n} \right]^{-1} .g(Y_n)$$

  - For $\begin{bmatrix} z_0 \\ \dot{y}_0 \end{bmatrix}$ $\frac{\partial g}{\partial Y}_{|Y=Y_n}$ is given by $\dot{M}(t,t_0) = \frac{d}{dt}\left[\frac{\partial X_{|t=t}}{\partial X_{|t=t_0}} \right]$; $M(t_0,t_0) = [Id]$ from $\dot{X} = f(X)$
  - For $\begin{bmatrix} T_{1/2} \end{bmatrix}$ is given by $\frac{dg}{dt}_{|Y=Y_n} = \dot{g}_{|Y=Y_n}$ i.e. the function f in $\dot{X} = f(X)$
  - Finally, a system of 42 variables to integrate into an iterative loop for finding one periodic orbit

Solution Periodic Orbits: EcosimPro practical approach

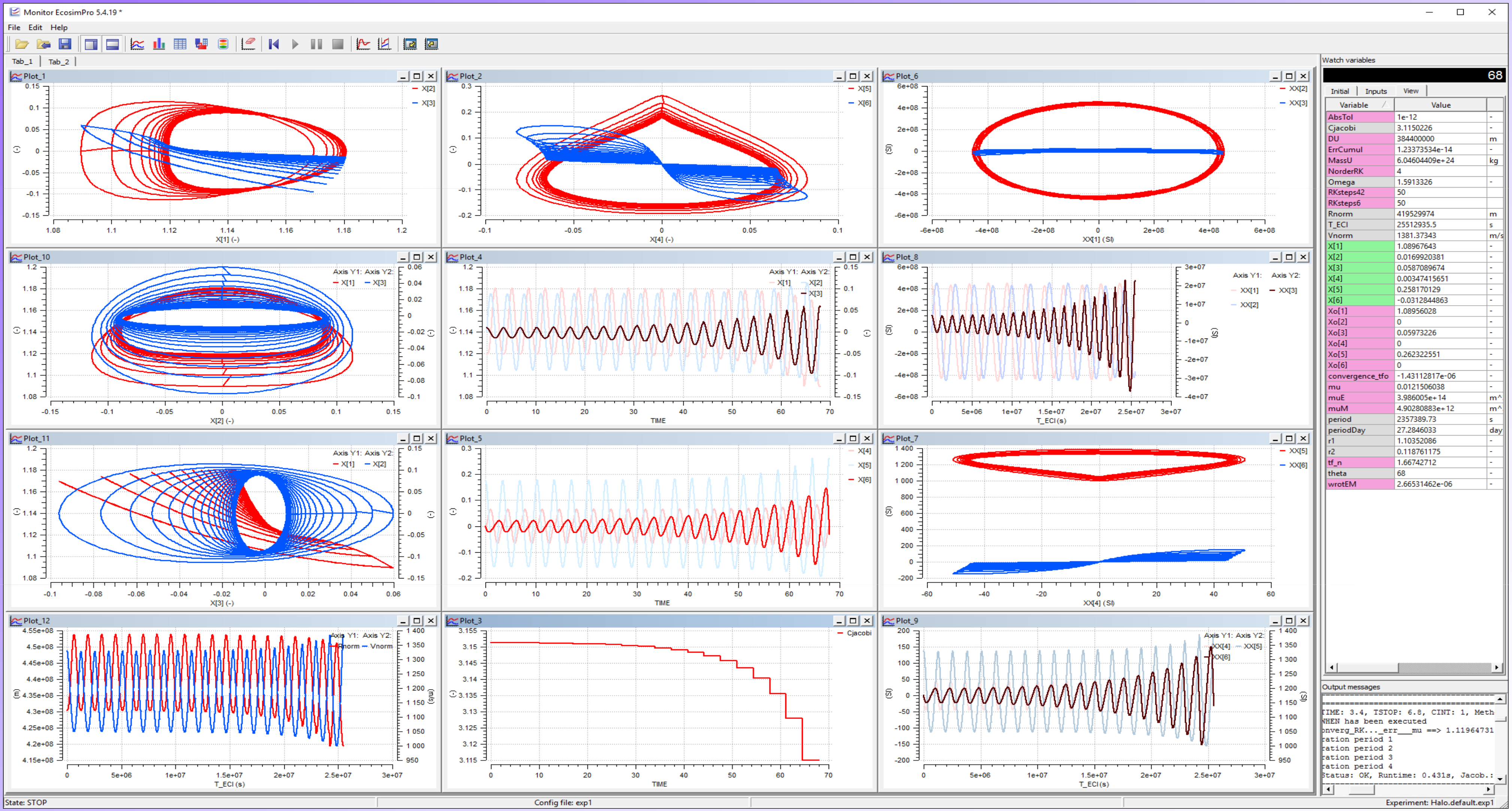
- Numbering the variables of the problem $\dot{X} = f(X)$ with index 1 to 6 and numbering the variable t to index 7
- $\frac{\partial g}{\partial Y}\Big|_{Y=Y_n}$ = $\left[Col.3\,5\ of\ rows\ 2\,4\,6\ of\ M(t,t_0)\right]\ \left[rows\ 2\,4\,6\ of\ f(X)\right]$
  - It was found that it was better to iterate on x_0 instead of z_0
  - ➔ just replace index 3 by 1 in above

- Loop on z_0 given values of each Halo
  - Loop on the 3 init conditions: solve the problem of 3 zeros
    - Integrate the 42 equation system
  - Iterate until zeros are found
- Further plots on the monitor the current Halo

$$\begin{pmatrix} x_0 = 1.12 \\ 0 \\ z_0 = 0.01\ldots \\ 0 \\ \dot{y}_0 = 0.17 \\ 0 \\ T_{1/2} = 1.7 \end{pmatrix}$$

Application to Halo orbits

Powerful plots under Monitor of EcosimPro

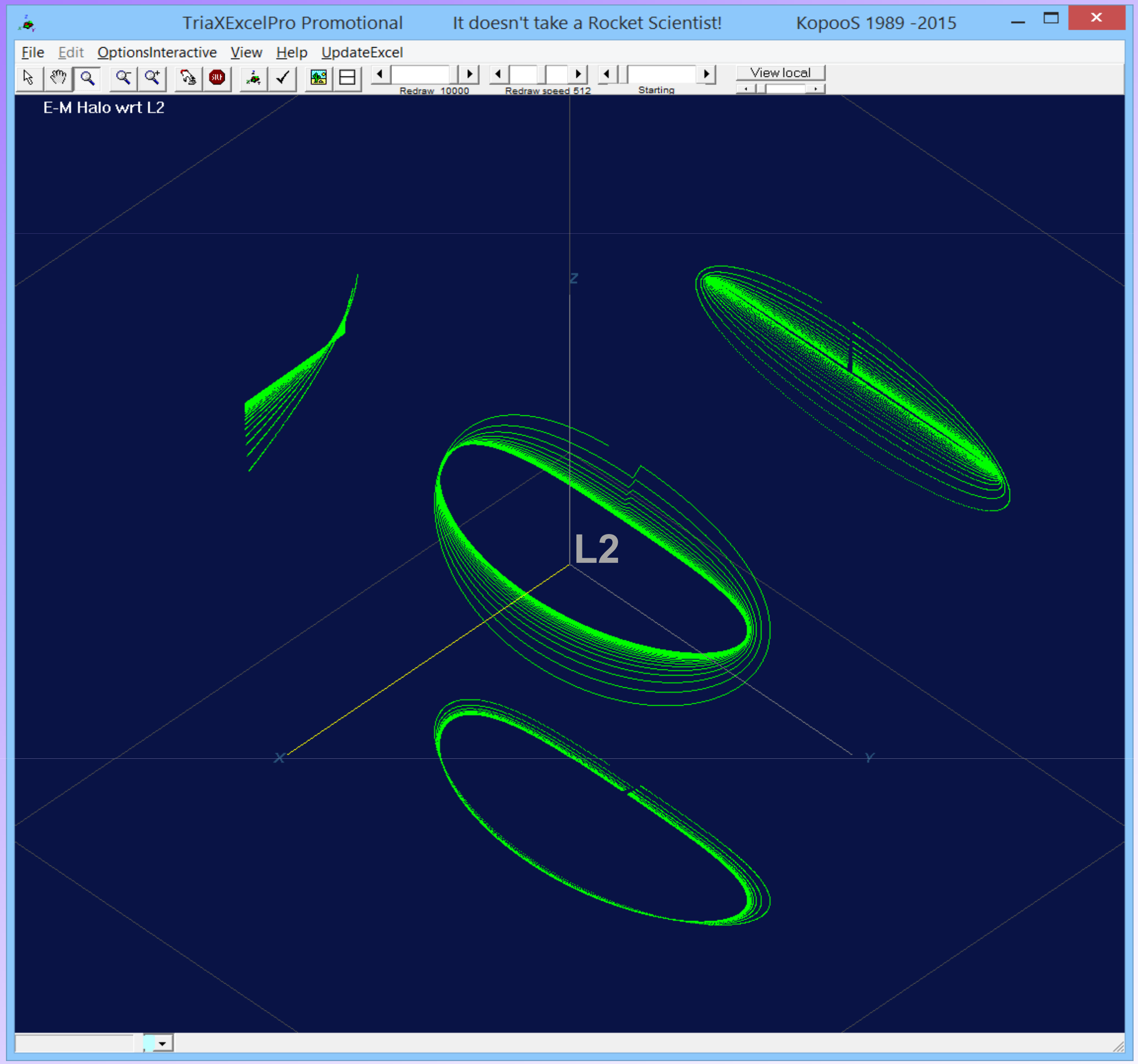


This document and the information contained are KopooS property and shall not be copied nor disclosed to any third party without KopooS prior written authorization

KopooS

Application to Halo orbits

And even better in 3D



Conclusions

- The paper has presented in simple words the mathematical problem of finding some Halo orbits
  - and the method implemented to solve it within the EcosimPro environment.
- The major advantages of the EcosimPro approach used successfully is to benefit of a real simulation framework based on models and on experiments
  - no mixing between the inputs\outputs needs and the real problem being to be solved.
- Hence the full model can be clearly and explicitly described
  - while the results coming from the experiments can be extensively assessed and analysed with simple EcosimPro monitor outputs.

Thanks for your attention

Questions?

Ackowledgments

- The research leading to these results is a KopooS funding

A taste of the EcosimPro model

•LISTING OF THE EXPERIMENT

```
/*---------------------------------------------------------------------------------
LIBRARY: MY_SAT
COMPONENT: Halo
PARTITION: default
EXPERIMENT: exp1
TEMPLATE: TRANSIENT
---------------------------------------------------------------------------------*/
EXPERIMENT exp1 ON Halo.default
DECLS
   REAL T_Halo
   STRING Filnam="Rep"
   INTEGER nbHalo=1
OBJECTS
INIT
BOUNDS
   MY_SAT.AbsTolM12 = 1e-012
   MY_SAT.NbSteps2000 = 50
   MY_SAT.NorderRK85 =4 -- 5
BODY
   GuessZ3notX1_o=1
   Xo1= 1.12 -- x_o
   Xo3=0.01 -- zo FIXED NOW
   Xo5= 0.17--ydot_o
   Thalfperiod_o=1.7-- t
   NloopNewtonHalo=15
   T_Halo=2*Thalfperiod_o
   nbHalo=20
   Filnam="Halo20.rpt"
   -- creates an ASCII file with the results in table format
   REPORT_TABLE(Filnam, " *X[*] *XX[*] *PHI* Cj* A* G* Halo* *12 *00 *85 *U *RK Om* R* T *
Teco* V* conv* mu* per* r* wrot*] dE* L* ")
   DEBUG_LEVEL= 1
   IMETHOD= DASSL
   setStopWhenBadOperation(FALSE)
   REL_ERROR = MY_SAT.AbsTolM12
   ABS_ERROR = REL_ERROR
   TOLERANCE =REL_ERROR    REPORT_MODE=IS_STEP
   TIME = 0
   FOR (i IN 1, nbHalo)
      FlagSearchPeriodicOrbit=TRUE
      INTEG_TO(TIME+T_Halo,1)
      -- Case of series of Halo orbits (evolution of z)
      IF i!=nbHalo THEN --change but not for the last one to keep all results of the last case
         Xo[3]=Xo[3]+i*Xo[3]*0.01
      END IF
   END FOR
END EXPERIMENT
```

•LISTING OF THE MODEL

```
COMPONENT Halo
DATA
   REAL Xo1=0.99197555537727 UNITS "DU" "xo"
   REAL Xo3=-0.00191718187218 UNITS "DU" "zo"
   REAL Xo5=-0.01102950210737 UNITS "DU/TU" "vyo"
   REAL Thalfperiod_o=1.52776735363559 UNITS "TU" "half period for periodic orbit, initial guess"
   INTEGER NloopNewtonHalo=0 UNITS "-" " 0 --no convergence-- else up to 14 is enough for convergennce"
   INTEGER GuessZ3notX1_o=3 UNITS "-" " flag=3 for xo fixed and zo guess ==>find a Lyapunov plan; flag=1 for zo fixed and xo guess ==>find Halo from a Lyapunov plan with some small zo "
   --REAL RunCode=2 UNITS "-" "code=0: J.D. Mireles James 1 Nick Truesdale 2: Earth Moon L2, 10: J.D. Mireles James L2 from Lyapunov, etc..."
DECLS
   BOOLEAN FlagSearchPeriodicOrbit=TRUE --directive for new search of periodic orbits
   CONST INTEGER LDIM=6
   INTEGER NorderRK,NbSteps, RKsteps42,RKsteps6,
GuessZ3notX1,Function_ODE_IVP --info
   INTEGER i462[3]={4,6,2}
   INTEGER i357[3]={3,5,7}
   REAL X[LDIM] UNITS "-" --position then velocity in barycentric rotating frame addim
   REAL theta UNITS "-"
   REAL T_ECI,period UNITS "s"
   REAL periodDay UNITS "day"
   REAL r1,r2,Omega,Cjacobi UNITS "-"
   EXPL REAL wrotEM3D[3] , wrotEMCrossXXrot[3] UNITS "-" --dim
   EXPL REAL XX[6], XXrot[3] UNITS "SI" --dim
   EXPL REAL Rnorm UNITS "m"
   EXPL REAL Vnorm UNITS "m/s"
   DISCR REAL Xf_n[LDIM] UNITS "-" --point then velocity in barycentric rotating frame addim
   DISCR REAL dX6_dt[LDIM] UNITS "-" --velocity then acceleration in barycentric rotating frame addim
   DISCR REAL Xo_n[7+10], Xo[7] UNITS "-" -- 6+added more rowse for compact information data
   DISCR REAL PHI[6,7] UNITS "-"
   DISCR REAL DF[3,3],D[3,3],XSo[3], XSo_star[3] ,Xff[3],ErrCumul UNITS "-"
   DISCR REAL muE,muS,muM UNITS "m^3/s^2"
   DISCR REAL dEM,AU,DU UNITS "m"
   DISCR REAL MassU UNITS "kg"
   DISCR REAL wrotEM UNITS "-"
   DISCR REAL mu UNITS "-"
   DISCR REAL G = 6.67384E-11 UNITS "m^3/(kg.s^2)"--+- 0.00080 m^3.kg^-1.s^-2
   DISCR REAL convergence_tfo UNITS "-"
   DISCR REAL to_n,tf_n,Thalfperiod UNITS "-"
   DISCR REAL AbsTol UNITS "-"
   DISCR REAL L1, L2, L3 UNITS "DU" --for info
INIT
   FOR (i IN 1,6)
     Xo[i] = 0
   END FOR
   GuessZ3notX1=GuessZ3notX1_o
   muE = 1*3.986005E14
   muS = 328902.82113001*3.986005E14--; % was Relative to earth
   muM = 0.0123000569113856 *3.986005E14
   mu=muM/(muE+muM)
   dEM=384400e3
   Xo[1]=Xo1 --GuessZ3notX1=3 --guess Z User to choose or default =3
   Xo[3]=Xo3
   Xo[5]=Xo5
   Thalfperiod=Thalfperiod_o
   DU=dEM
  MassU=(muE+muM)/G
   wrotEM=sqrt(G*MassU/DU**3)
   --for info here only because mu in known and allow computation of L1 L2 L2
   L1=findLagrangePoints(0.83, mu)-- init value not too far from the wanted roots
   L2=findLagrangePoints(1.15 , mu)
   L3=findLagrangePoints(-1.0, mu)
   PRINT (" for information: L1,L2,L3 in DistanceUnits,earthMoon= $L1 $L2 $L3 ")
   --Eco Normal Init of the derivatives
FOR (i IN 1,6)
   X[i]=Xo[i]
 END FOR
 Xo[7]= Thalfperiod --variable added
 i357[1]=GuessZ3notX1
DISCRETE
 WHEN FlagSearchPeriodicOrbit THEN -- this is like a program to be run before starting integratons by EcosimPro depending on the directive FlagSearchPeriodicOrbit .
   --Inputs : Xo[i] (including Xo[7]= Thalfperiod), NloopNewtonHalo , mu OUT: X[i] initialized by Xo which is set to the last converged Xo_n[i] (for a good starting guess for other periodic orbits)
   --Iteration on the suited IVP fulfilling the goal (with xo fixed (index 1) )
   -- goal: after a half_period vx,vz and y shall be all null (index 4,6,2) with free variables to guess: initial values of zo, vyo, half_period (index 3,5 and variable tf_n)
   FlagSearchPeriodicOrbit=FALSE --clear the condition for running this routine
   to_n=0 --never modified here
   FOR (i IN 1,7)
     Xo_n[i]=Xo[i] --here we work with IVP Xo_n (including Thalfperiod) because Xo is never modified inside the next loop
   END FOR
   --@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
   FOR (k IN 1,NloopNewtonHalo)
     --call ODE integration for the final state Xf_n from the given IVP Xo_n to see how good are the guesses and process the iterations
     AbsTol=AbsTolM12--1E-12
     NbSteps=NbSteps2000
     NorderRK=NorderRK85
     Function_ODE_IVP=LDIM
     tf_n=Xo_n[7] -- tf is a condition final for the ODE but it is as Thalfperiod an initial condition for the process of finding a periodic solution by convergence Newton
     ODE113 (LDIM, to_n, tf_n, Xo_n, Xf_n, NorderRK, AbsTol, NbSteps, mu, Function_ODE_IVP, RKsteps6 )--out Xf_n
     --Zero search by Newton method iterations
     FOR (i IN 1,3)
       XSo[i]=Xo_n[i357[i]]
     END FOR
     FOR (i IN 1,3)--Array with the 3 components results of ODE integration to be nullified by converging the IVP XSo to XSo_star
       Xff[i]=Xf_n[i462[i]] -- i462[3]={4,6,2} i357[3]={3,5,7}
     END FOR
     --Jacobian at current final point tf_n=Xo_n[7] wrt IVP initial Xo_n given for to_n -- IT INCLUDES THE ODE113 SIZE 42
     STMatrixCR3BP ( to_n, tf_n , Xo_n, PHI, mu , RKsteps42)-- out PHI = d FF / d xx = d xxdot_i / d xx_j
     --derivative of X6 wrt time at final point, needed for getting the time derivatives to fill the matrix DF (dFF/dxx)
     Function_ODE_IVP_6( 6, Xf_n, dX6_dt, mu )
     FOR (i IN 1,6)--extended PHI last column added with time derivatives d FF / d t = d xxdot_i / d t in column 7
       PHI[i,7] = dX6_dt[i]
     END FOR
     -- dFF/dxx Full derivative of XXf (to be nullified) wrt XXo (selected state variables and time) i462[3]={4,6,2} i357[3]={3,5,7}
   FOR (i IN 1,3)
     FOR (j IN 1,3)
       DF[i,j] =PHI[i462[i],i357[j]] -- i462[3]={4,6,2} i357[3]={3,5,7}
     END FOR
   END FOR
   InvMatrix( 3,DF, D , ErrCumul)
   --XSo_star The next solution guess : XSo_star = XSo-inv(dFF/dxx)*Xff
 FOR (i IN 1,3)--extended PHI with time derivatives
   XSo_star[i]=XSo[i]-SUM (j IN 1,3; D[i,j]*Xff[j])
   END FOR
 --New Xo_n = Xo_n+1 for iterations
   FOR (i IN 1,7)
     Xo_n[i]=Xo[i] --come back to the first init conditions before update of the selected ones
   END FOR
   FOR (i IN 1,3)
     Xo_n[i357[i]]=XSo_star[i]--update the selected ones with better guesses
   END FOR
 -- end for the new Xo_n, ready to go for iterations
     --PRINTa1 (3, XSo_star , "new guess")
     --convergence and for info
     convergence_tfo=XSo_star[3]-XSo[3]
     Xo_n[8]= convergence_tfo --for info only and printing
     Xo_n[9]= NorderRK --for info only and printing
     Xo_n[10]= RKsteps6 --for info only and printing
     Xo_n[11]= RKsteps42 --for info only and printing
     Xo_n[12]= ErrCumul --for info only and printing
     Xo_n[13]= mu --for info only and printing
   END FOR --k
   --@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
   PRINTa1 (13, Xo_n, "final_Xo_n--_tf_n_converg_RK..._err--_mu")
   FOR (i IN 1,7)--Update Xo from last converged Xo_n, and also memorized for starting other periodic orbit search if any
     Xo[i]=Xo_n[i] --including the time tf_n
   END FOR
   --Update wrt Init: New init conditions for derivative variables for EcosimPro integration: the right one for a periodic orbit
   FOR (i IN 1,6) --only 6 for X
     X[i]=Xo[i]
   END FOR
 END WHEN
CONTINUOUS
  r1=((mu+X[1])**2+X[2]**2+X[3]**2)**(1/2)--distance point to body1
  r2=((mu+X[1]-1)**2+X[2]**2+X[3]**2)**(1/2)--distance point to body2
  EXPAND (i IN 1,3) X[i+3] = X[i]'
  --dynamic f=ma in barycentric rotating frame, see for example J.D. Mireles James and many others
  X[4]'=+X[1]+2*X[5]-(X[1]+mu)*(1-mu)/r1**3-(X[1]+mu-1)*mu/r2**3
  X[5]'=+X[2]-2*X[4]-X[2]*(1-mu)/r1**3-X[2]*mu/r2**3
  X[6]'=-X[3]*(1-mu)/r1**3-X[3]*mu/r2**3
  --for info
  Omega=0.5*(X[1]**2+X[2]**2)+(1-mu)/r1+mu/r2
  Cjacobi=2*Omega-(X[4]**2+X[5]**2+X[6]**2)
  --Geocentric results in ECI with vector XX
  T_ECI=TIME/wrotEM --TIME is addim = 6.28 for 1 period
  period=2*3.14159265358979323846264338327950 /wrotEM
  periodDay=period/86400
  EXPAND (i IN 1,2) wrotEM3D[i]=0 -- only 2 first coordinates
  wrotEM3D[3]=wrotEM -- the 3rd coordinate
  --cross product
  wrotEMCrossXXrot[3]=wrotEM3D[1]*XXrot[2]-wrotEM3D[2]*XXrot[1]
  wrotEMCrossXXrot[1]=wrotEM3D[2]*XXrot[3]-wrotEM3D[3]*XXrot[2]
  wrotEMCrossXXrot[2]=wrotEM3D[3]*XXrot[1]-wrotEM3D[1]*XXrot[3]
  EXPAND_BLOCK (i IN 1,3)
     XXrot[i] = X[i]*DU
     XX[i+3] = X[i+3]*DU*wrotEM+wrotEMCrossXXrot[i]
  END EXPAND_BLOCK
  theta=TIME --wrotEM*T_ECI
  XX[1] = XXrot[1]*cos(theta)-XXrot[2]*sin(theta)
  XX[2] = XXrot[1]*sin(theta)+XXrot[2]*cos(theta)
  XX[3] = XXrot[3]
  -- useful
  Rnorm=sqrt(SUM(i IN 1,3; XX[i]**2))
  Vnorm=sqrt(SUM(i IN 4,6; XX[i]**2))
END COMPONENT
```


KopooS