A powerful example, differential system at 6 dimensions with periodic solutions, on EcosimPro

Sommaire

1. The differential system

We have $\dot{X} = f(X)$ with

$$X = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \qquad f(X) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ x + 2\dot{y} - \dfrac{(1-\mu)(x+\mu)}{r_1^3} - \mu \dfrac{(x-(1-\mu))}{r_2^3} \\ y - 2\dot{x} - \dfrac{(1-\mu)y}{r_1^3} - \mu \dfrac{y}{r_2^3} \\ -\dfrac{(1-\mu)z}{r_1^3} - \mu \dfrac{z}{r_2^3} \end{bmatrix} \text{ with } \begin{matrix} \|r_1\| = \sqrt{(x+\mu)^2 + y^2 + z^2} \\ \|r_2\| = \sqrt{(x-(1-\mu))^2 + y^2 + z^2} \end{matrix} \quad \begin{matrix} r_1 = \begin{bmatrix} x+\mu \\ y \\ z \end{bmatrix} \\ r_2 = \begin{bmatrix} x-(1-\mu) \\ y \\ z \end{bmatrix} \end{matrix}$$

For some periodic orbit it is sufficient to find initial conditions of X (position and velocity) such that after half an orbit some values remain null as initially: $y_{T½}=0=y_0$ and $\dot{x}_{T½}=0=\dot{x}_0$ and $\dot{z}_{T½}=0=\dot{z}_0$.

In clear, the starting point is in the x,z plane because $y_0=0$ with no velocity on x, z but $\dot{y}_0 \neq 0$.

That means that if $X(0) = \begin{bmatrix} x_0 \\ 0 \\ z_0 \\ 0 \\ \dot{y}_0 \\ 0 \end{bmatrix}$ and $X(T½) = \begin{bmatrix} x \\ 0 \\ z \\ 0 \\ \dot{y} \\ 0 \end{bmatrix}$ then such orbit is periodic. One notes that in the final condition there are 3 zeros.

2. Solutions of periodic orbits

To find a periodic orbit, we suppose that a first starting point not too far from the solution is given. In order to reduce the number of guesses to do, one can fix the initial value x_0 (and further make a loop on it) .

The new problem of the 3 zeros is to find Y such that $g(Y) = 0$ with $Y = \begin{bmatrix} z(0) \\ \dot{y}(0) \\ t \end{bmatrix}$ and $g(Y) = \begin{bmatrix} y(t) \\ \dot{x}(t) \\ \dot{z}(t) \end{bmatrix}$

(the value of t is $T½$ *, the time at which* $\mathbf{y}$ *is for the first time null after the integration 0 to t; i.e. the trajectory is back to the x,z plane).*

Iterations by Newton method can be used for finding right guesses: $Y_{n+1} = Y_n - \left[\dfrac{\partial g}{\partial Y}_{|Y=Y_n} \right]^{-1} .g(Y_n)$

Like for a 1-D curve: for finding a new guess u_1 after u_0 of $h(u)=0$, one follows the tangent of the curve $v=h(u)$ at point $u= u_0$ up crossing the line "$v=0$": $v_0=h(u_0)$ so $\dfrac{v_0 - 0}{u_0 - u_1} = \dfrac{dh}{du}_{|u=u_0}$. Hence $u_0 - u_1 = \left[\dfrac{dh}{du}_{|u=u_0} \right]^{-1} .(v_0 - 0)$ and finally $u_1 = u_0 - \left[\dfrac{dh}{du}_{|u=u_0} \right]^{-1} .h(u_0)$.

Here the tangent (or Jacobian differential) is a bit delicate because in the definition of *g* there is integration from 0 to T½. The derivative matrix $\frac{\partial g}{\partial Y}_{|Y=Y_n}$ is :

- for the sub set of variable *Y*: $\begin{bmatrix} z(0) \\ \dot{y}(0) \end{bmatrix}$, it comprises a subset of the **derivative** of the general first function $\dot{X} = f(X)$ for which one have (for the numerical analysts they call it the STM state transition matrix) $M(t,t_0) = \frac{\partial X_{|t=t}}{\partial X_{|t=t_0}}$ which is defined by a differential equation $\dot{M}(t,t_0) = \frac{d}{dt}\left[\frac{\partial X_{|t=t}}{\partial X_{|t=t_0}}\right]$ with an initial value of $M(t_0,t_0) = \frac{\partial X_{|t=t_0}}{\partial X_{|t=t_0}} = [Identity]$. *That is an impressive system of 36 differential equations to be integrated simultaneously from t=0 to t. Hopefully some of the equations are trivial...*

  Note that obviously the derivative is the null matrix at $t=t_0$ because the time is not explicitly appearing in the equations f(X) -- *the numerical analysts say that the system is autonomous*-- $\dot{M}(t_0,t_0) = \frac{d}{dt}\left[\frac{\partial X_{|t=t_0}}{dX_{|t=t_0}}\right] = \frac{d}{dt}[1] = [0]$ -- *Also for that reason, one have*

  $$\dot{M}(t,t_0) = \frac{d}{dt}\left[\frac{\partial X_{|t=t}}{\partial X_{|t=t_0}}\right] = \left[\frac{\partial \dot{X}_{|t=t}}{\partial X_{|t=t_0}}\right] = \left[\frac{\partial \dot{X}_{|t=t}}{\partial X_{|t=t}}\right] \cdot \left[\frac{\partial X_{|t=t}}{\partial X_{|t=t_0}}\right] = \left[\frac{\partial f}{\partial X}\right] \cdot M(t,t_0) \implies \dot{M}(t,t_0) = \left[\frac{\partial f}{\partial X}\right] \cdot M(t,t_0)$$

- And of course, $\frac{\partial g}{\partial Y}_{|Y=Y_n}$ for the sub set of variable *Y*: $[\,t\,]$ i.e. $\frac{dg}{dt}_{|Y=Y_n} = \dot{g}_{|Y=Y_n}$ it is **directly** a sub set of the function $\dot{X} = f(X)$ itself.

Just with the numbering of the variables of the first problem with index 1 to 6 and numbering the variable *t* to index 7, one get straight forward successively: *Y= 3 5 7* and *g(Y)=2 4 6*.

So $\frac{dg}{dY}_{|Y=Y_n} = \frac{d\,2\,4\,6}{d\,3\,5\,7}_{|3\,5\,7=Y_n} = [Column\ 3\ 5\ of\ lines\ 2\ 4\ 6\ of\ M(t,t_0)]$ and $[_{in\ last\ col}\ the\ lines\ 2\ 4\ 6\ of\ f(X)]$

Finally the solution of periodic orbits is performed by the integration of a system of 36 + 6= **42** differential equations and that within a loop for finding the solution $g(Y) = 0$ with Newton, which lead to periodic orbit. A further loop on the fixed variable allows plotting many halo orbits. With some tests, it was better to guess x_0 while keeping fixed z_0 so in the equations above it is just matter of replacing the index 3 by index 1.

Note: **The model and experiment is a stand alone model within EcosimPro "as-is"** *without need of sophisticated libraries like ESPSS.* Just in addition to the equations in EL (EcosimPro langage) shown below, a simple function so called "**ODE113**" has been implemented for the integration of the 6 and 42 differential equations (based on Runge-Kutta with possibility of error control and variable time steps) and also a matrix inversion routine with error quantification has been added. *Such features could be as well added by the Ecosimpro team to* **EcosimPro "as-is"!**

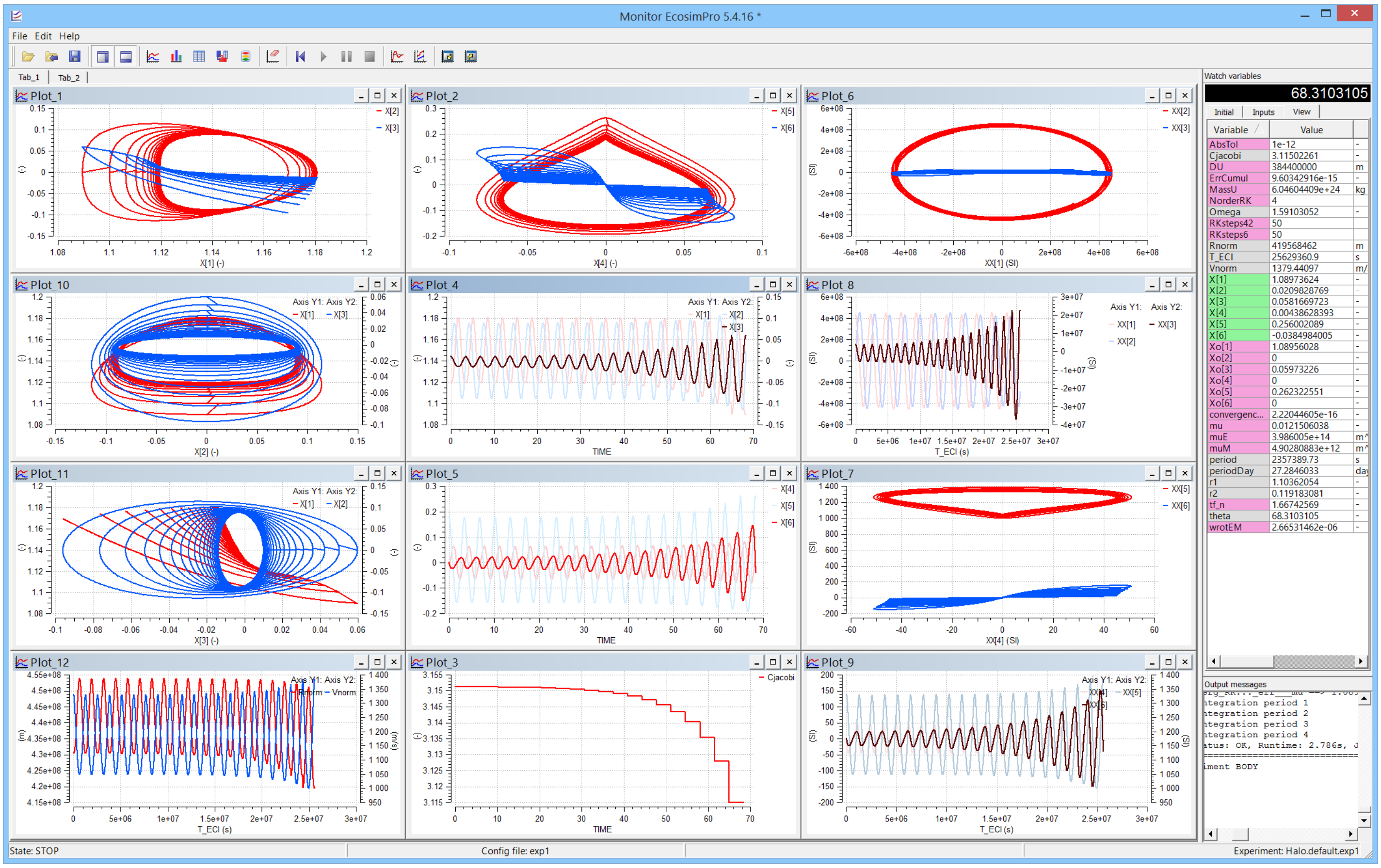
3. Application to Halo orbits Lagrange point L2 of the system Earth+Moon

The system of equations §1 represent the CRTBP (circular restricted 3 body problem). For the Earth+Moon system, μ = 0.0121506038.

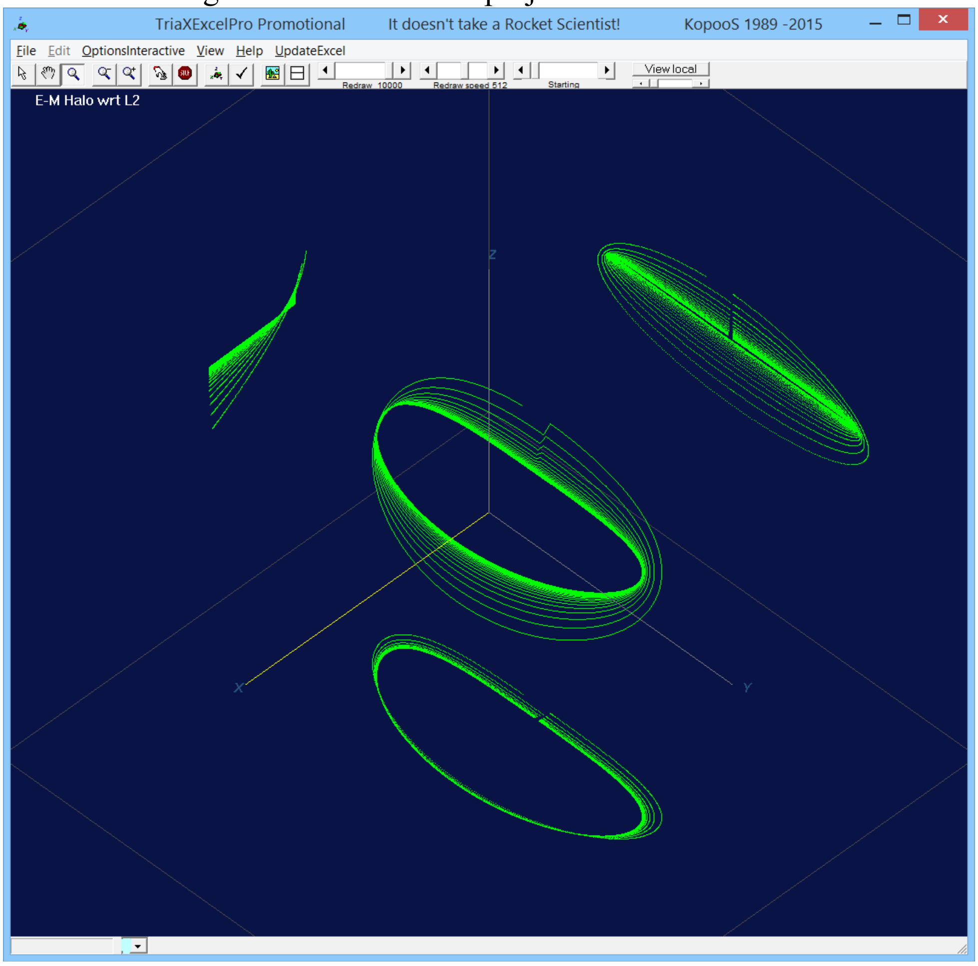
With initial values

Xo1= 1.12037906887683 -- x_o
Xo3=0.01 -- z_o FIXED NOW
Xo5= 0.176061510401881 --ydot_o
Xo7=Thalfperiod_o=1.70775776152685 -- t

we get the following simulation plots (with Xo[3]=Xo[3]+i*Xo[3]*0.01 for i=1 to 20) of 20 Halo orbits with run time around of 2 seconds for each Halo orbit.



Of course, the above plot is very useful for analysts, but for a first view, using a 3D visualisation tool feed by the data from EcosimPro we can get a cubic view with projections of the orbits on the reference planes:



Annex: Traceability

Listing of the experiment

```
-- ' 30/09/2015 17:34:16
/*--------------------------------------------------------------------------------------
LIBRARY: MY_SAT
COMPONENT: Halo
PARTITION: default
EXPERIMENT: exp1
TEMPLATE: TRANSIENT
CREATION DATE: 14/08/2015
--------------------------------------------------------------------------------------*/
EXPERIMENT exp1 ON Halo.default
DECLS
  REAL T_Halo
  STRING Filnam="Rep"
  INTEGER nbHalo=1
OBJECTS
INIT
  -- initial values for state variables
  BOUNDS
  -- Set equations for boundaries: boundVar = f(TIME;...)
  MY_SAT.AbsTolM12 = 1e-012
  MY_SAT.NbSteps2000 = 50
  MY_SAT.NorderRK85 =4 -- 5
BODY
  GuessZ3notX1_o=1
  Xo1= 1.12037906887683 -- x_o
  Xo3=0.01 -- zo FIXED NOW
  Xo5= 0.176061510401881--ydot_o
  Thalfperiod_o=1.70775776152685-- 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 -- set the debug level (valid range [0,4])
  IMETHOD= DASSL -- select default integration solver
  setStopWhenBadOperation(FALSE)-- Set flag to stop when bad numerical operation occurs (eg division by 0). By default do not stop.
  REL_ERROR = MY_SAT.AbsTolM12-- set relative and absolute tolerance for DASSL solver (transient solver)
  ABS_ERROR = REL_ERROR
  TOLERANCE =REL_ERROR -- 1e-006 -- set relative tolerance for algebraics solver (steady solver)
  REPORT_MODE=IS_STEP -- REPORT_MODE=IS_EVENT,IS_CINT,IS_STEP -- when to report results
  -- calculates a steady state
  --STEADY()
  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

```
-- ' 30/09/2015 17:24:42
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 convergence"
  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 DistanceUnitsEarthMoon= $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 (m IN 1,3; D[i,m]*Xff[m])
      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 teh 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*3.141592653589793238462643383279 5 /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
```