A Comparison of Regularization Methods for few-body interactions

S. Mikkola
Turku University Observatory, 21500 Piikkiö, Finland.

Abstract

In this work several methods to regularize the computation of few-body motions are discussed. These include the global regularization and the CHAIN-algorithm, which utilize the KS-transformation, and two other methods, the logarithmic Hamiltonian (logH) and time transformed leapfrog (TTL), which use a regularizing algorithm rather than coordinate transformation. All these methods are useful to advance strongly interacting subsystems in N-body simulations. Facility of use and programming are considered and accuracy tests in various systems, including some with large mass ratios, are conducted.
1 Extended phase space and time transformations

Poincare’s time-transformed extended phase space Hamiltonian can be written

\[ \Gamma = g(p, q)(H(p, q, t) + B), \]

where \( B \) is the momentum of time (also: binding energy), for which the initial value \( B(0) = -H(0) \) guarantees that \( \Gamma = 0 \) along the orbit. The equations of motion are

\[
\begin{align*}
\dot{p}' &= -\frac{\partial \Gamma}{\partial q} = -g \frac{\partial H}{\partial q} - \frac{\partial g}{\partial q}(H + B) \\
\dot{q}' &= \frac{\partial \Gamma}{\partial p} = +g \frac{\partial H}{\partial p} + \frac{\partial g}{\partial p}(H + B) \\
\dot{t}' &= \frac{\partial \Gamma}{\partial B} = g.
\end{align*}
\]

where \( H + B = 0 \) along the orbit, but these terms cannot be dropped.

In N-body computations the time-transformation function \( g(p, q) \sim r \), where \( r \) is the shortest distance. Functions having such quality in the N-body problem are

\[ g = 1/U \; ; \; U = \text{potential} \quad \text{or} \quad g = 1/L \; ; \; L = \text{Lagrangian}. \]

A recommendable choice is \( g = 1/L \) which gives

\[ \Gamma = (H + B)/L = (H - E)/L. \]
2 Regularization methods

In addition to a time transformation coordinate transformations and/or Hamiltonian manipulation and selection of a suitable algorithm can be used to regularize N-Body integrations.

**KS-transformation:**

1D: \( q = Q^2 \)

2D: \( x + iy = (Q_1 + iQ_2)^2 \) (Levi-Civita)

3D: \( ix + jy + kz = U^* i U \) (KS)

where \( U = iQ_1 + jQ_2 + kQ_3 + Q_4 \) is a quaternion.

These lead, with a suitable time transformation, to harmonic oscillator type behavior in near collisions.

**The logarithmic Hamiltonian**

\[
\Lambda = \ln(T + B) - \ln(U)
\]

and the leapfrog algorithm provide regularization of two-body collisions.

**Time-Transformed Leapfrog (TTL)**

is a non-canonical generalization of the logH-method. Here the Newtonian equations

\[
\dot{x} = v, \quad \dot{v} = F(x),
\]

are transformed to

\[
x' = v/W, \quad t' = 1/W, \quad v' = F(x)/\Omega(x), \quad W' = \frac{\partial \Omega}{\partial x} v
\]

for which a leapfrog-like algorithm can be constructed and regularization of two-body collisions is obtained if \( \Omega \sim 1/r \) near collision.
3 KS TRANSFORMATION

Matrix formulation

In three dimensions the KS-transformation of coordinates \( \mathbf{R} \) and momenta \( \mathbf{W} \) may be written

\[
\mathbf{R} = \mathbf{\bar{Q}} \mathbf{Q} ; \quad \mathbf{W} = \mathbf{\bar{Q}} \mathbf{P} / (2\mathbf{Q}^2)
\]  

(1)

Here \( \mathbf{\bar{Q}} \) is the KS-matrix (e.g. Stiefel and Scheifele (1972) p. 24)

\[
\mathbf{\bar{Q}} = \begin{pmatrix}
Q_1 & -Q_2 & -Q_3 & Q_4 \\
Q_2 & Q_1 & -Q_4 & -Q_3 \\
Q_3 & Q_4 & Q_1 & Q_2 \\
Q_4 & -Q_3 & Q_2 & -Q_1
\end{pmatrix}.
\]  

(2)

The KS momenta are given by

\[
\mathbf{P} = 2\mathbf{\bar{Q}}^t \mathbf{W}.
\]  

(3)

With the time transformation

\[
\frac{dt}{ds} = R = \mathbf{Q}^2
\]  

(4)

one obtains the Hamiltonian

\[
\Gamma = R(H - E) = \frac{1}{8} \mathbf{P}^2 - K^2 - E\mathbf{Q}^2,
\]  

(5)

where \( H = \frac{1}{2} \mathbf{W}^2 - K^2 / R \) is the two-body Hamiltonian and \( E = H(0) \) is the numerical value of the energy.
4 Simple (1d ) introduction to various regularization methods

The one-dimensional problem with the Hamiltonian

\[ H = \frac{p^2}{2} - \frac{1}{q}. \]

can be integrated with various regularization methods:

**1d-KS:**

Transformation \( q = Q^2, \ P = p/(2Q), \ t' = Q^2. \)

With \( B_0 = -H(0) \) the new regular Hamiltonian is

\[ \Gamma = t'(H + B_0) = \frac{1}{8}P^2 + B_0Q^2 - 1 \]

and the equation of motion

\[ P' = -\frac{\partial \Gamma}{\partial Q} = -2P_0Q, \]
\[ Q' = \frac{\partial \Gamma}{\partial P} = P/4, \]
\[ t' = \frac{\partial \Gamma}{\partial B_0} = Q^2, \]

can be integrated with any reasonable numerical method.
**1d-LogH:**

One may also use the logarithmic Hamiltonian

\[ \Lambda = \ln(p^2/2 + b_0) - \ln(1/q), \]

with \( b_0 = -H(0) \), and try to solve approximately the equations of motion

\[
\begin{align*}
q' &= p/(p^2/2 + b_0) \\
p' &= -1/q \\
t' &= 1/(p^2/2 + b_0),
\end{align*}
\]

using the leapfrog. Defining the two 'subroutines':

\[
X(s) : \begin{align*}
\delta t &= s/(p^2/2 + b_0) \\
q &\rightarrow q + \delta t \ p \\
t &\rightarrow t + \delta t
\end{align*}
\]

and

\[
V(s) : \begin{align*}
p &\rightarrow p - s/q
\end{align*}
\]

one can symbolize the leapfrog by

\[ X(h/2)V(h)X(h/2), \]

i.e. half step \( X(h/2) \), then full step \( V(h) \), followed by a half step \( X(h/2) \).

Surprisingly, this happens to produce the correct trajectory, having only a phase error.
The toy code: 1dLogH.f

```
implicit real*8 (a-h,o-z)
q=1  ! initialize
p=0
b=-(p*p/2-1/q)
h=0.1d0
1 continue
   call qm(h/2,p,q,b,t)
   call pm(h ,p,q)
   call qm(h/2,p,q,b,t)
   err=p*p/2-1/q+b  ! Energy error
   erru=q*err      ! This should remain constant (but for round-off)
   write(6,*)t,q,p*q,err,erru
   if(t.lt.100.)goto1
end

subroutine qm(s,p,q,b,t)
implicit real*8 (a-h,o-z)
dt=s/(p*p/2+b)
t=t+dt
q=q+dt*p
return
end

subroutine pm(s,p,q)
implicit real*8 (a-h,o-z)
p=p-s/q
return
end
```
1d-TTL:

The normal leapfrog method is often used for the Newtonian equations of motion, which in our simple case read

\[ \dot{p} = -1/q^2, \quad \dot{q} = p. \]

The collision \((q \to 0)\) is, however, singular and the basic leapfrog does not work.

The idea of TTL is to introduce a time transformation is a such a way that a modified leapfrog can be constructed. Write

\[ p' = \frac{\dot{p}}{\Omega(q)}, \quad q' = \frac{\dot{q}}{W}, \quad t' = \frac{1}{W} \]

where we consider \(W\) to be a 'velocity-like' variable and \(\Omega(q)\) is some suitable function of \(q\). Clearly we must have, in numerical value, \(W = \Omega(q)\), but to make the construction of a leapfrog possible we obtain the value of \(W\) from the differential equation

\[ \dot{W} = \dot{\Omega}(q) = \frac{\partial \Omega}{\partial q} p, \quad \text{or} \quad W' = \dot{W}/\Omega(q). \]

In the 1d-case a reasonable choice is \(\Omega = 1/q\), so that

\[ q' = p/W, \quad t' = 1/W, \quad p' = -1/q, \quad W' = -p/q. \]

The equations

\[ q' = p/W, \quad t' = 1/W \]

can be solved for constant \(p\) and \(W\) and

\[ p' = -1/q, \quad W' = -p/q \]

are solvable for constant \(q\), thus a leapfrog can be constructed as a composite of these two pairs.
The 'subroutines' are

\[ \mathbf{X}(s) : \]
\[
\begin{align*}
\delta t &= s/W \\
q &\rightarrow q + \delta t \ p \\
t &\rightarrow t + \delta t
\end{align*}
\] (9)

and

\[ \mathbf{V}(s) : \]
\[
\begin{align*}
\delta p &= -s/q \\
p_a &= p + \delta p/2 \\
p &\rightarrow p + \delta p \\
W &\rightarrow W - sp_a/q
\end{align*}
\] (10)

giving the leapfrog

\[ \mathbf{X}(h/2) \mathbf{V}(h) \mathbf{X}(h/2), \]
The toy code: 1dTTL:

```fortran
implicit real*8 (a-h,o-z)
q=1
p=0
b=p*p/2-1/q  ! =Energy
W=1/q        ! initial value
h=0.1d0
t=0

1 continue
    call qm(h/2,p,q,w,t)
    call pm(h,p,q,w)
    call qm(h/2,p,q,w,t)

c diagno
    err=p*p/2-1/q-b ! Energy error
    erru=q*err    !
    write(6,123)t,q,p,q,err,erru
123 format(1x,f10.4,2f10.4,1p,3g10.2)
    if(t.lt.100.)goto 1
end

subroutine qm(s,p,q,w,t)
    implicit real*8 (a-h,o-z)
    dt=s/w
    t=t+dt
    q=q+dt*p
    return
end

subroutine pm(s,p,q,w)
    implicit real*8 (a-h,o-z)
    dp=-s/q
    pa=p+dp/2
    p=p+dp
    w=w-s*pa/q
    return
end
```

10
Figure 1: **ENERGY ERRORS IN 1dTTL and 1dLogH.**

Figure 2: **Motion of q**
Some conclusions:

- Mathematically the logH and TTL methods are equivalent in this simple case [increment of $w$ is equivalent to that of the kinetic energy in logH].

- The difference in precision is due to numerical effects: there is more round-off in updating the $W$-variable. One notes that in collision $W \to \infty$ and then it return back to ‘normal’. All this with many increments that can be individually very large.

- In more general cases one may recommend the logH-method in case it is possible to use

$$\Omega = \text{potential}.$$ 

End of simplicity......
N-Body system in one dimension

One-dimensional CHAIN

The Hamiltonian of the problem is given by

\[ H = \sum_i \frac{p_i^2}{2m_i} - \sum_{i<j} \frac{m_im_j}{|x_i - x_j|}. \]

New coordinates \( X_k = x_{k+1} - x_k \) and the generating function

\[ S = \sum_{k=1}^{N-1} W_k X_k = \sum_{k=1}^{N-1} W_k (x_{k+1} - x_k), \]

together with the canonical transformation

\[ X_k = Q_k^2; \quad W_k = \frac{P_k}{2Q_k}, \]

and a suitable time transformation

\[ g = 1/L = 1/(T + U), \]

gives the new regularized Hamiltonian

\[ \Gamma = (T - U - E_0)/(T + U), \]

where

\[ T = \sum_{k=1}^{N-1} \left( \frac{1}{8} \frac{1}{m_k} + \frac{1}{m_{k+1}} \right) \frac{P_k^2}{Q_k^2} - \sum_{k=2}^{N-1} \frac{P_{k-1}P_k}{4Q_{k-1}Q_k m_k} \]

\[ U = \sum_{k=1}^{N-1} \frac{m_k m_{k+1}}{Q_k^2} + \sum_{i<j+1} \frac{m_im_j}{\Sigma_{k=i}^{j-1} Q_k^2}. \]

The equations of motion may be written

\[ H = T - U; \quad g = 1/(T + U); \quad \Gamma = g(H - E), \]
\[ \Gamma_T = (1 - \Gamma)g; \quad \Gamma_U = -(1 + \Gamma)g, \]
\[ \Gamma_{P_k} = \Gamma_T T P_k, \]
\[ \Gamma_{Q_k} = \Gamma_T T Q_k + \Gamma_U U Q_k, \]
\[ P_k' = -\Gamma Q_k, \]
\[ Q_k' = \Gamma P_k. \]
LogH and TTL (1, 2 and 3 dimensions....)

With

\[ T = \sum_i \frac{p_i^2}{2m_i}, \quad U = \sum_{i<j} \frac{m_im_j}{|x_i - x_j|} \quad (11) \]

and the binding energy \( B = U - T \) one may define the logarithmic Hamiltonian

\[ \Lambda = \ln(T + B) - \ln(U), \]

which gives the equations of motion

\[ p'_k = \frac{1}{U} \frac{\partial U}{\partial x_k}, \quad x'_k = \frac{1}{T + B} \frac{\partial T}{\partial p_k}. \quad (12) \]

and since \( p' \) only depends on \( x \) and \( x' \) only on \( p \), a leapfrog is possible. One expects this to give good approximations for close approaches because it is asymptotically exact for two-body collisions.

The TTL formulation is now

\[ t' = \frac{1}{W}, \quad x'_k = \frac{1}{W} \frac{\partial T}{\partial p_k} \quad (13) \]

\[ p'_k = \frac{1}{\Omega} \frac{\partial U}{\partial x_k}, \quad W' = \sum_k \frac{\partial \Omega \frac{\partial T}{\partial p_k}}{\partial x_k} / \Omega \quad (14) \]

and the leapfrog for this system can be constructed.

Note that the soft potential

\[ U = \sum_{i<j} m_im_j/\sqrt{|x_i - x_j|^2 + \epsilon^2}, \quad (15) \]

is ‘regularized’ by these algorithms.
The toy code 1dNBTTL.f

1dNBTTL.f
implicit real*8 (a-h,o-z)
real*8 q(6),p(6)
common/softening/eps,U
read(5,*)NB,h,tmx,eps
read(5,*) (q(k),k=1,nb)
read(5,*) (p(k),k=1,nb)
call energy(nb,p,q,b,w)
goto 2
1 continue
ts=t
do kie=1,10000
call qm(nb,h/2,p,q,w,t) ! LEAPFROG
call pm(nb,h,p,q,w) !
call qm(nb,h/2,p,q,w,t) !
if(t.gt.ts+.01)goto 2
end do
c diagno
2 continue
call energy(nb,p,q,bt,U)
er=-bt+b
erru=err/U
write(6,123)t,W/U-1,erru,(q(k),k=1,nb)
123 format(1x,f10.3,1p,8g12.4)
if(t.lt.tmx)goto 1
end
subroutine qm(nb,s,p,q,w,t)
implicit real*8 (a-h,o-z)
real*8 q(6),p(6)
dt=s/w
t=t+dt
do k=1,nb
q(k)=q(k)+dt*p(k)
end do
return
end
subroutine pm(nb,s,p,q,w)
implicit real*8 (a-h,o-z)
real*8 q(6),p(6),A(6)
common/softening/eps,U
do k=1,nb
A(k)=0
end do
U=0
do i=1,nb-1
do j=i+1,nb
xij=q(i)-q(j)
rij=sqrt(xij**2+eps**2)
aij=-xij/rij**3
U=U+1/rij ! pot
a(i)=a(i)+aij ! acc_i
a(j)=a(j)-aij ! acc_j
end do
end do
dlt=s/U
do k=1,nb
  pa=p(k)
p(k)=p(k)+dlt*a(k)
  pa=(pa+p(k))/2
w=w+dlt*a(k)*pa
end do
return
end

subroutine energy(nb,p,q,b,w)
  implicit real*8 (a-h,o-z)
  real*8 q(6),p(6)
  common/softening/eps,U
T=0
U=0
do k=1,nb
  T=T+p(k)**2/2
end do
do i=1,nb-1
do j=i+1,nb
  U=U+1/sqrt((q(i)-q(j))**2+eps**2)
end do
do end do
b=U-T
w=U
return
end
Figure 3: **Three-body motion in 1d (hard potential).**

![Figure 3](image)

Figure 4: **Three-body motion in 1d (soft potential).**

![Figure 4](image)

Figure 5: **Energy errors in hard and soft potential calculations**

![Figure 5](image)
Figure 6: KS regularized interactions in the global method of Heggie, CHAIN-method and Zare’s cartwheel method.
5  HEGGIE’S GLOBAL METHOD

Starting with the Hamiltonian

\[ H = \sum_{i=1}^{N} \frac{1}{2m_i} w_i^2 - \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{m_i m_j}{r_{ij}} \]  \hspace{1cm} (16)

Heggie (1974) introduces new independent variables \( r_{ij} = r_j - r_i \). The generating function

\[ S = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \mathbf{w}_{ij} \cdot (r_j - r_i) \], \hspace{1cm} (17)

defines the moments \( \mathbf{w}_{ij} \). The resulting expression for the Hamiltonian can be simplified by renaming the variables \((\mathbf{w}_{ij}, \mathbf{r}_{ij})\) to \((\mathbf{p}_k, \mathbf{q}_k)\), (in a loop 'do i=1,N-1; do j=i+1,N; k=k+1 ...'). The kinetic energy and potential can then be written

\[ T = \sum_{\mu \nu} T_{\mu \nu} P^t_{\mu} \bar{Q}_{\mu}^t \bar{Q}_{\nu} P_{\nu}/(4Q_{\mu}^2 Q_{\nu}^2), \hspace{1cm} U = \sum_k M_k / Q_k^2 \], \hspace{1cm} (18)

and with \( \Gamma = (T - U - E)/(T + U) \) a concise algorithm can be formulated:

(i) Calculate initial values and auxiliary quantities

\[ k = (i - 1)N - i(i + 1)/2 + j \]
\[ \mathbf{q}_k = \mathbf{r}_i - \mathbf{r}_j \]
\[ \mathbf{p}_k = (m_i \dot{r}_i - m_j \dot{r}_j)/N \]
\[ M_k = m_i m_j \] \hspace{1cm} (19)
\[ a_{ik} = 1, \hspace{0.5cm} a_{jk} = -1 \] [other components = 0]
\[ T_{\mu \nu} = \frac{1}{2} \sum_{n=1}^{N} a_{n \mu} a_{n \nu} / m_n \]
(ii) With the KS variables $P_k$ and $Q_k$ compute

\[
\begin{align*}
    p_k &= \frac{Q_k P_k}{(2Q_k^2)} \\
    A_k &= \sum_{\mu=1}^{K} T_{k\mu} p_\mu \\
    d_k &= A_k \cdot p_k \\
    T &= \sum_{k=1}^{K} d_k, \quad U = \sum_{k=1}^{K} M_k/Q_k^2 \\
    T_{P_k} &= \frac{Q_k^t A_k}{Q_k^2} \\
    T_{Q_k} &= \frac{(P_k^t A_k^* - 4d_k Q_k)}{Q_k^2} \\
    A_k^* &= (A_1, A_2, A_3, -A_4)_k^t, \\
    U_{Q_k} &= -2M_k Q_k/Q_k^4 \\
    L &= T + U; \quad H = T - U; \quad \Gamma = (H - E)/L \\
    \Gamma_T &= (1 - \Gamma)/L; \quad \Gamma_U = -(1 + \Gamma)/L. \\
    Q'_k &= \Gamma_T T_{P_k} \\
    P'_k &= -\Gamma_T T_{Q_k} - \Gamma U_{Q_k} \\
    t' &= 1/L.
\end{align*}
\]

(iii) Transformation from the KS variables to the physical ones:

\[
\begin{align*}
    r_{ij} &= q_k = \bar{q}_k Q_k \\
    w_{ij} &= p_k = \bar{q}_k P_k/(2Q_k^2) \\
    r_i &= \left(\sum_{j=i+1}^{N} m_j r_{ij} - \sum_{j=1}^{i-1} m_j r_{ji}\right)/\sum_{n=1}^{N} m_n \\
    \dot{r}_i &= \left(\sum_{j=i+1}^{N} w_{ij} - \sum_{j=1}^{i-1} w_{ji}\right)/m_i.
\end{align*}
\]
Suppose a chain of vectors connecting $N$ bodies has been selected and bodies relabeled 1, 2, .., $N$ along the chain. The generating function

$$S = \Sigma_{i=1}^{N-1} W_k \cdot (q_{k+1} - q_k) \quad (23)$$

defines a transformation to new chain-coordinates $R_k = q_{k+1} - q_k$ and momenta $W_k$. The Hamiltonian takes the form

$$H = \Sigma_{k=1}^{N-1} \frac{1}{2} \left( \frac{1}{m_k} + \frac{1}{m_{k+1}} \right) W_k^2 - \Sigma_{k=2}^{N} \frac{1}{m_k} W_{k-1} \cdot W_k - \Sigma_{k=1}^{N-1} \frac{m_k m_{k+1}}{R_k} - \Sigma_{1 \leq i < j \leq 2} \frac{m_i m_j}{R_{ij}} \quad (24)$$

The KS transformations $R_k = Q_k Q_k$, $W_k = Q_k P_k / (2 Q_k^2)$ and the 'standard' Hamiltonian

$$\Gamma = g(H - E) = (T - U - E)/(T + U)$$

gives

$$P'_k = -\partial \Gamma / \partial Q_k, \quad Q'_k = \partial \Gamma / \partial P_k.$$
Figure 7: Illustration of the chain and the checking of switching conditions. Distances like $R_{5,7}$ are compared with the smaller of the two distances $R_{5,6}$ and $R_{6,7}$ (marked by *). Interparticle distances like $R_{4,10}$ are compared with the smallest of those in contact with the considered distance (marked by $\times$).

Chain selection begins by finding the shortest interparticle vector. This is taken as the first part of the chain under construction. One proceeds by searching for the particle which is closest to one or the other end of the presently known part of the chain. Once identified, this particle is added to the chain (to the end which is closer). This operation is repeated until all particles are included in the chain. There is also a check for need of a new chain after every integration step.
8 LogH and TTL in small clusters

8.1 logH:

With the usual notation

\[ T = \sum_i \frac{p_i^2}{2m_i} \]  
\[ U = \sum_{i<j} \frac{m_i m_j}{|\mathbf{r}_i - \mathbf{r}_j|} + U_{extr}(\mathbf{r}, t) \]

and the logarithmic Hamiltonian

\[ \Lambda = \ln(T + B) - \ln(U), \]

the basic leapfrog subroutines \( \mathbf{X}(s) \) and \( \mathbf{V}(s) \) are

\[ \mathbf{X}(s) : \]
\[ \delta t = s/(T + B) \]
\[ \mathbf{r}_k \rightarrow \mathbf{r}_k + \delta t \mathbf{p}_k \]
\[ t \rightarrow t + \delta t \]

and

\[ \mathbf{V}(s) : \]
\[ \mathbf{p} \rightarrow \mathbf{p}_k + \frac{s}{U} \frac{\partial U}{\partial \mathbf{r}_k} \]
\[ B \rightarrow B + \frac{s}{U} \frac{\partial U}{\partial t} \]

and the leapfrog reads

\[ \mathbf{X}(\frac{h}{2})\mathbf{V}(h)\mathbf{X}(\frac{h}{2}), \]

Leapfrog alone is not satisfactory and it must to be used just as a basic algorithm to provide results for a **Gragg-Bulirsch-Stoer extrapolation method.**
A very simple two-body toy code (LogH-method)

    implicit real*8 (a-h,m,o-z)
    real*8 x(3),v(3)
    M=1.d0
    read(5,*)h,Tmx,x,v
    t=0
    r=sqrt(x(1)**2+x(2)**2+x(3)**2)
    vv=v(1)**2+v(2)**2+v(3)**2
    B=M/r-vv/2

    c
    c Integration of the two-body motion
    1 continue

    dt=h/(v(1)**2+v(2)**2+v(3)**2+2*B)
    do k=1,3
        x(k)=x(k)+dt*v(k) ! x-motion
    end do
    t=t+dt

    c
    dtc=h/(x(1)**2+x(2)**2+x(3)**2)
    do k=1,3
        v(k)=v(k)-dtc*x(k) ! v-motion
    end do

    dt=h/(v(1)**2+v(2)**2+v(3)**2+2*B)
    do k=1,3
        x(k)=x(k)+dt*v(k) ! x-motion
    end do
    t=t+dt
    t=t-h**3/12/(M)**2 ! +O(h^5) time correction (only for small h)

    c diagnostics
    r=sqrt(x(1)**2+x(2)**2+x(3)**2)
    vv=v(1)**2+v(2)**2+v(3)**2
    Bt=M/r-vv/2
    write(6,*)t,x,Bt-B
    if(t.lt.Tmx)goto 1
    end
8.2 TTL:

The TTL equations

\begin{align}
t' &= \frac{1}{W} \\
r_k' &= \frac{1}{W} \frac{\partial T}{\partial p_k} \\
v_k' &= \frac{1}{\Omega} A_k \\
W' &= \sum_k \frac{\partial \Omega}{\partial r_k} \cdot v_k / \Omega
\end{align}

where \( v, A \) are the velocity and acceleration, can be used to construct the leapfrog \( X(h/2) V(h) X(h/2) \), where

\begin{align}
X(s) : \\
\delta t &= s / W \\
r_k &\rightarrow r_k + \delta t \ p_k \\
t &\rightarrow t + \delta t
\end{align}

\begin{align}
V(s) : \\
\delta v_k &= \frac{s}{\Omega} A_k \\
W &\rightarrow W + \frac{s}{\Omega} \sum_k \frac{\partial \Omega}{\partial r_k} \cdot (v_k + \frac{1}{2} \delta v_k) \\
v_k &\rightarrow v_k + \delta v_k
\end{align}

For \( \Omega \) one may use any suitable function, usually

\[ \Omega = \sum_{i < j} \frac{\Omega_{ij}}{r_{ij}} \]

where

\[ \Omega_{ij} = 1, \quad \text{or} \quad \Omega_{ij} = m_i m_j, \]

the latter choice being recommendable iff the masses are comparable. Finally: use extrapolation method.
- Facility of programming (in author’s opinion)

**LogH < TTL < Heggie < CHAIN**

where \( A < B \) means ’A is simpler than B’.

-The algorithmic regularization methods **LogH** and **TTL** are fairly simple to program: differences compared with the most straightforward programming are minor in the simplest implementations.

- Comparing the methods that use KS-transformation suggests that in fact the Heggie’s global method is simpler than CHAIN. This is because all interactions are treated in the same way.

- The CHAIN has some complicated bookkeeping which make it harder to implement, however, it may be worth of it!
11 CONCLUSION

- For comparable masses the CHAIN algorithm is clearly preferable.

- If extreme many-body encounters are excluded, then LogH may compete with CHAIN. It is also enormously simpler to code!

- For very large mass ratios it becomes evident that TTL is more accurate and faster.
Literature


