TRIPLE & CHAIN User Manual

Sverre Aarseth

Email: sverre@ast.cam.ac.uk

Institute of Astronomy, University of Cambridge

1 Introduction

This is a User Manual for the few-body regularization codes *TRIPLE* and *CHAIN*. A complete description of the former method is given in Aarseth & Zare [1974, hereafter AZ] while chain regularization is treated in several papers [Mikkola & Aarseth 1990, 1993, 1996]. It is remarkable that going from $N = 3$ to $N = 4$ should take 17 years. On the other hand, the generalization from 2D to 3D motion by Kustaanheimo & Stiefel [1965, hereafter KS] was not achieved for another 60 years.

The computational task in three-body regularization is to integrate a set of 17 first-order differential equations. Thus there are eight coordinates and corresponding momenta, with the time as an additional variable (if desired). Since the resulting equations are well behaved away from the triple singularity, accurate solutions can readily be achieved in most cases.

2 Three-body implementation

Inspection of the final Hamiltonian reveals the underlying principle of two coupled harmonic oscillators which are expressed by the KS formalism. As far as decision-making is concerned, for comparable masses it is desirable (but not vitally important) to ensure that the choice of reference body m_3, satisfies the condition $R \equiv |\mathbf{R}_1 - \mathbf{R}_2| > \min(R_1, R_2)$, where $\mathbf{R}_1$ and $\mathbf{R}_2$ are the two-body vectors connecting m_3 with m_1 and m_2. In that case, the so-called singular contribution associated with the terms $m_1 m_2$ are *numerically* smaller than the corresponding regular terms.

The code structure is extremely simple. Thus the basic version (called *TRIPLE*) contains the following routines in standard *FORTRAN*:

Table 1: *TRIPLE routines.*

MAIN	Initialization, integration control and decision-making
DATA	Initial conditions and transformation to c.m. frame
DERQP	Derivatives for the equations of motion
DIFSY1	Bulirsch–Stoer high-order integrator
PERI	Determination of the closest two-body separation
TRANSF	Transformation to or from regularized quantities

Before discussing the overall strategy and decision-making, we comment on the tasks carried out by the subroutines listed here.

Routine *DATA* illustrates the case of doing just one example. The initial conditions (m, $\mathbf{r}$, $\mathbf{v}$) for each particle are given by the template input file `input`. Essential to the AZ (as well as the chain) method is that these variables must be expressed in the centre-of-mass (c.m.) rest frame; i.e. the six c.m. integrals have been used to reduce the order from 18 to 12. Once a few other initializations have been done the calculation can begin. Unless a specific application is intended, it is convenient to employ consistent scaled quantities of order unity. Generalization to a series of experiments in order to examine different parameters then requires repeated calls to this routine which can be automated. Note that it is helpful (but not required) to define the heaviest body initially as m_3. It is also useful to attach particle labels (so-called names) to each body which facilitates identification, i.e. switching of equal-mass bodies.

Routine *DERQP* evaluates the right-hand sides of the 17 equations of motion in terms of the masses and regularized variables ($\mathbf{Q}$, $\mathbf{P}$). These expressions have been derived in a more compact form than in the original code [cf. Aarseth 2003, hereafter AA03] and are therefore slightly more time-consuming. However, there is a gain in clarity and the respective terms in the regularized Hamiltonian can now be readily identified from the geometry of the problem (as defined above). Note that the standard version employs the time transformation $t' = R_1 R_2$.

The Bulirsch–Stoer [1966] integrator *DIFSY1* is very compact and may be considered a black box. It has been adapted to the present setup by Seppo Mikkola and obtains solutions of the differential equations for a specified tolerance (see below). For each call, there are a number of function evaluations depending on the tolerance and the solution at the end of each step is given in terms of a rational polynomial extrapolated to zero step. Hence there are usually quite large changes in the variables from one call to the next.

An additional routine (*PERI*) for determining the closest two-body separation has been included for convenience since this quantity is often of interest. It is based on an algorithm by Seppo Mikkola which employs regular variables [cf. AA03]. Calls to this routine are only made at the beginning of a new step if the smallest two-body separation is below a suitably small value. Note that the actual pericentre is obtained as the osculating solution which is reliable if the perturbation is small. However, if all three bodies are close, the smallest separation itself is a good measure because the steps are also small.

The transformation routine *TRANSF* serves three purposes. At the beginning, the total energy is evaluated from the physical variables. This quantity (actually *twice* the energy is saved as *ENERGY*) appears explicitly in the equations of motion. Certain mass factors used by the equations of motion are also evaluated initially and at every subsequent change of reference body. Likewise, a transformation from physical coordinates and momenta to the corresponding regularized variables is carried out. These transformations are of standard KS type, which makes for clarity. Finally, this routine is used in order to obtain the physical variables $\mathbf{r}$, $\mathbf{v}$ by the inverse KS transformations for diagnostic information as well as the energy check. In this connection, we emphasize that R_1 and R_2 play an important role in the decision-making. Since these quantities are not known exactly at the end of each step, the relation $R_k = \mathbf{Q}_k^2$ is evaluated explicitly in the main routine.

Having reviewed all the relevant subroutines, we now turn our attention to routine *MAIN* of program *TRIPLE* and focus on the basic version. The main input consists of two numbers: the tolerance, *TOL0*, and maximum integration time, *TCRIT*. The latter may be expressed in units of the crossing time, *TCR*. If desired, another criterion may be used to limit the extent of distant ejections (cf. *RMAX* in some versions). The initial time-step is constructed by taking a fraction of the minimum of the parabolic crossing time and the smallest travel time and converting to regularized units by $t' = R_1 R_2$. The input array for the integrator is initialized by copying the regularized coordinates and momenta, $\mathbf{Q}$, $\mathbf{P}$ as given by *TRANSF*. Although the time may be obtained explicitly by using a certain time transformation, we assume for simplicity that the full 17 equations are integrated. In principle there is a choice of time transformations and stabilized equations (c.f. AA03). The suppressed part of *DERQP* contains the alternatives $t' = R_1 R_2/(R_1 + R_2)^{1/2}$ and $t' = 1/U$ (potential energy), where the latter permits explicit time evaluation.

Several procedures are carried out following the completion of an integration step. First a check is made for a possible change of reference body using a simple distance test, i.e. $R < \min(R_1, R_2)$ which may be modified for large mass ratios. Any switching is performed by transformation to physical variables and appropriate relabelling of the reference body, followed by a transformation to new regularized variables. Likewise, the physical variables are obtained before the data analysis. Note that because of the large number of lengthy function evaluations, any such overheads represent a negligible part of the total effort.

For the standard three-body problem with negative total energy, the escape of a particle implies that the remaining binary is even more energetic. Hence the excess energy is directly related to the velocity at infinity for an isolated system. Although more rigorous escape criteria are available [cf. AA03], a factor of 20 in the semi-major axis ratio combined with the excess energy condition is sufficient in practice. A check for escape (or distant ejection) is carried out for significant distance ratios (defined by R_1/R_2) and the basic two-body elements are printed before termination. Additional parameters are summarized at the end, such as step number, the minimum pair-wise separations and relative energy error.

Other versions of the code are available (see *TRIPLE2*) which deal with time reversal or movie display based on X11 or *PGPLOT* software. Note that since the number of steps is relatively small even for an accurate calculation, the assigned step needs to be reduced significantly (say by a factor of 10) for giving a smooth display. Moreover, each movie frame must be accompanied by considerable extra calculations (depending on the CPU) in order to increase the viewing time. The *TRIPLE* code has proved effective in treating strong interactions and is therefore suitable for studying a variety of scattering problems. Questions of accuracy are a matter of taste but a tolerance of 10^{-10} is fairly standard and usually leads to relative energy errors of the same magnitude. However, a factor of 100 smaller is recommended for more demanding integrations.

Once the code is downloaded, it should first be uncompressed before the routines are extracted by '`tar xvf triple.tar`'. The command '`make triple`' which employs the `Makefile` should then compile and produce the executable, whereupon a test calculation may be tried with the test input provided.

3 Chain regularization

The concept of chain regularization is a straightforward generalization of the three-body formulation. However, the simplest case of adding one more particle allows the important problem of binary–binary interactions to be studied. Essentially the Hamiltonian function is constructed along analogous lines. On the other hand, the decision-making grows in complexity when more bodies are added. Once the initial two-body vectors have been introduced, the integration itself proceeds in a similar manner. Now there are $8(N-1)$ equations of motion for isolated systems, with one extra equation for the time. It turns out that the latter is more effectively treated by using the inverse Lagrangian energy rather than the product of all the distances. For generality, we also include one differential equation for the energy as well as integration of the c.m. coordinates and velocity due to possible external effects, making a total of $8N$ equations. Hence in the basic version there are seven redundant equations for the c.m. and energy for later use.

In the following we describe some relevant features of the routines which are listed in Table 3 of the Appendix. It can be seen that the *CHAIN* code consists mainly of three separate parts: decision-making to achieve optimal geometry, transformations (two types) and the integration itself where several procedures are implemented. Moreover, the overall decision-making is quite involved because of the need to consider a variety of situations. Consequently it is natural to adopt a logical order of the discussion which reflects the code structure. Although a certain amount of explanatory comments are included, much of the coding is quite compact. The present version consists of more than 30 routines and 3200 lines, compared to about 900 lines for the basic three-body code.

Initial conditions. Depending on the requirements, a total of nine parameters constitute the main input (cf. definitions and template in *CHAIN*). Here *RESC* (in units of the gravitational radius) is used for escape removal and *ECRIT* for time reversal tests if the indicator *IREV* is assigned a positive value. There are additional indicators for close encounters or collisions, special treatment (so-called slow-down) of super-hard binaries and movie display. The treatment of the initial conditions in routine *CHINIT* is based on the same principle as for three-body regularization. For computational convenience, the physical chain coordinates and velocities are described by one-dimensional arrays but conversion to the standard 2D form facilitates data analysis.

Chain selection. As in the three-body case, the total energy (and angular momentum) is evaluated first (routine *CONST*). The next task is to select appropriate chain vectors. This is achieved by routine *SELECT* which uses the heapsort algorithm of Press as well as routine *REDRAW*. The latter routine checks the chain connections which are derived by maximizing the relative forces rather than using distances only. Following some initializations, such as the gravitational radius $R_{\rm grav}$ (denoted *RGRAV*), crossing time and the first integration step, the calculation proper may begin.

Switching condition check. Upon return from the next Bulirsch–Stoer step, the integration variables are saved in *COMMON* by routine *YSAVE*. This is followed by transformation to physical variables by *TRANSX*. Since checking of the optimal chain structure requires little effort, this task is carried out after each call to routine *DIFSY1*. Any change of chain vectors is implemented by routine *SWITCH* which makes use of the dual routines *KSPHYS* and *PHYSKS* together with *SELECT*. Except for long-lived resonance systems, the number of new chain configurations is relatively small.

Integration method. The integration of the equations of motion by routine *DIFSY1* is based on the same principles as before. Explicit expressions for the right-hand sides are given elsewhere [cf. Mikkola & Aarseth 1990, 1993; AA03]. For computational convenience, the inverse two-body chain distances, *RINV*, are available in the same manner as R_1, R_2 of AZ (i.e. not given at the very end of a step). The regular and non-chained contributions to the derivatives formed by routine *DERQP* are evaluated separately for both types (i.e. potential and kinetic energy terms). If the solution converges for the specified tolerance, the next interval is chosen by an expression which limits the growth. As before, typical relative energy errors are comparable to the tolerance. Additional procedures for close encounters or collisions are described in detail below.

Escaper updating. Depending on the strategy, a calculation may be terminated after the particle number is reduced to a specified level or the maximum integration time has been exceeded. Since particle numbers up to about 10 are practical here, (cf. parameter *NMAX*) it is useful to include an algorithm for removal of distant members. In the simplest case, a particle may be defined as an escaper if it has positive energy with respect to the system c.m. and the central distance is significant. For convenience, routine *ESCAPE* compares the *radial* velocity with the parabolic velocity and uses an additional distance criterion of $5R_{\rm grav}$. The case of two particles escaping together as a binary is also considered (i.e. $R_b < 0.1R_{\rm grav}$), in which case the c.m. motion is examined. Reduction of the chain membership is carried out by routine *REDUCE* which eliminates any escaping particles consecutively if required, compared to the star cluster version where a binary is initialized as one body after removal from the chain. Following a successful updating of the basic variables and reduction to the c.m. rest frame, the calculation resumes with the standard initialization procedure so that the current parameters are consistent.

Three-body stability. Stable triples may form in several ways. The most probable channel is for binary–binary encounters with relatively small impact parameters in which one binary component is ejected and the other intruder ends up in a bound outer orbit by angular momentum conservation. Stable triples may also form as the result of escape or possibly collision. Hence it is of interest to provide a stability test to prevent lengthy calculations where no significant changes occur. In practice, this means a condition for *escape* of the outer component since the Zare exchange criterion is less stringent and also does not apply to inclined systems because of an angular momentum degeneracy [cf. AA03]. Routine *CHSTAB* includes a new stability criterion [Mardling 2008] valid for a range of outer mass ratios. It replaces an earlier criterion based on the binary tides problem which included a fitting constant as well as a heuristic correction factor for the inclination [Mardling & Aarseth 1999]. Thus a triple system where the outer pericentre distance exceeds the inner semi-major axis by a small factor (about 4 for modest outer eccentricity and similar masses) is deemed to be stable. If this termination is bypassed, continued calculation should show no secular change in the inner semi-major axis, while the eccentricity may exhibit Kozai cycles with amplitude depending on inclination.

Collision algorithm. Since the purpose of the chain code is to perform accurate calculations of close encounters, it is natural to consider particles of finite size in order to model compact systems. One application would be to hard binary–binary scattering where the radii may be an appreciable fraction of the semi-major axis. It is worth noting that a particle size places an upper limit on the ejection velocity resulting from three-body interactions, thereby providing an alternative outcome to collision. Given the large relative changes per step in the two-body separation during close encounters, a proper collision determination poses an interesting algorithmic challenge. In many cases it would be adequate to implement collision by employing the osculating pericentre distance. However, use of the latter technique based on physical variables ($\mathbf{r}$, $\mathbf{v}$) may not not be reliable at nearly singular separations. Instead an accurate iteration method has been developed which will be outlined below.

If the input indicator JCOLL is activated, the determination of the minimum separation is initiated at the first step past pericentre as defined by comparing the sum of all *inverse* square chain distances at three successive times. The first (of two) collision indicators used for the decision-making is activated if the smallest two-body separation is below a suitable factor times the sum of the two corresponding radii. Note that this is only done on the first function call of a new step. Regular two-body elements are then obtained by routine PERI which yields the osculating pericentre distance.

The iteration itself starts if the minimum distance is below eight times the sum of the respective radii (which may be set to zero if collisions are not of interest). Following the regular evaluation of the semi-major axis (routine EREL) and pericentre passage time (routine TPERI), a suitable negative step is taken [cf. AA03 for details]. Thus the iteration is continued according to the sign of the regularized radial velocity until convergence (i.e. a small radial velocity) is achieved, which typically occurs after about four steps. The calculation is continued by copying the state vectors ($\mathbf{Q}$, $\mathbf{P}$) at the collision instant and transforming to physical variables. Routine NEWSYS then combines the two bodies inelastically in the local c.m. frame, whereupon routine REDUCE performs the membership reduction (still in the c.m. rest frame). Lastly, the new chain variables are initialized as for the standard case. Since the collision energy is available (cf. EREL), the energy error can also be monitored after each such reduction.

Slow-down procedure. The formation of a close binary by multiple encounters usually leads to ejection by virtue of the recoil velocity. However, it may be of interest to consider one or more initial (or primordial) binaries with periods much less than the crossing time[1]. An unperturbed binary only requires two or three Bulirsch–Stoer integration steps per orbit. Still, the presence of just one super-hard binary may lead to increased function calls even in the absence of significant three-body interactions.

The principle of adiabatic invariance may be used to modify the equations of two-body motion by the so-called slow-down procedure [Mikkola & Aarseth 1996]. For computational simplicity, only one such binary is treated specially here but the procedure may be generalized. The dimensionless slow-down factor, κ, is obtained from an expression containing the relative perturbation due to the nearest particles in the chain (i.e. one

[1]Note that TCR itself may become very small if binary energies are included.

or two). Hence the essential task is to determine an accurate value of the smallest semi-major axis at each step so that κ represents a consistent value and the deviation from unity is a measure of the perturbation. Hence the physical interval corresponding to a regularized time-step is increased by κ. It should be emphasized that although strict energy conservation no longer applies, the solutions are essentially correct.

Discrete changes (which need not be small) are accommodated by a simple modification of the regularized Hamiltonian (cf. *EJUMP* in *SLOW* and *DERQP*). Thus at any time the transformation to the physical variables yield the actual values. Further details are given in the original paper [also cf. AA03]. For convenience, a positive value of the input indicator *ISLOW* is used to activate the procedure if $\min R_i < R_{\rm slow}$ with $R_{\rm slow} = 0.5 R_{\rm grav}$. In general, initial conditions without binaries may not benefit by this optional treatment, except possibly for large N and several fast escapers. A more careful evaluation of the perturbations based on vectors is employed in large N-body simulations where several hard binaries may be interacting.

Time reversal. A facility for time reversal may be useful in evaluating the reliability of a given solution. Thus instead of merely placing faith in the energy or angular momentum conservation, time reversal provides a stricter accuracy test. If the optional indicator *IREV* is positive at termination time, the regularized momenta and c.m. velocity (and thereby the velocities) are reversed and the time is set to zero. Hence integration over the elapsed interval may recover the initial conditions to some accuracy. An iteration is employed at the end in order to ensure that the correct time interval has been reached. If possible, the tolerance is reduced by a factor of 100 if the rms error in coordinates or velocities exceeds the prescribed value of *ECRIT*. Relevant results are produced on unit #7 or #8 according to whether the solution is acceptable. The success of such tests depends to a large extent on the time interval and complexity of the interactions. Since the growth of errors is exponential, one cannot expect to reproduce the initial conditions of relatively long-lived systems, especially if the phase plays an important role.

Movie displays. Routines have been provided for generating movies in colour using X11 graphics. However, calls to *MOVIE*0 and *MOVIE* sometimes give problems since the `Makefile` needs information about the X11 library which is installation dependent (see Appendix). Some control variables are read as an additional input line if the movie indicator is active (note the choice *IMOVE* = 1, 2, 3 for step reduction). The extra effort in delaying the next movie frame (cf. *WAIT* loop) should be adjusted according to the CPU speed. In principle, the movie can be combined with time reversal, or slow-down, escape and collisions.

Data analysis. A small amount of relevant output is provided at termination as well as for escape and collisions. Routine *BINARY* signals termination if the energy of the dominant binary exceeds the total energy in absolute value but this feature can be suppressed if not suitable. It is envisaged that the display of results will be modified according to requirements. Likewise, the code can be readily changed to deal with a series of experiments where the parameter space is explored.

4 Basic chain formulation

Given the presence of a versatile chain regularization code, it is an easy task to construct a compact version which only features the essential procedures. Such a code is particularly useful for educational purposes where the primary objective is to demonstrate the effectiveness of the method.

In the following we discuss some aspects of the compact code *CHAIN*0 which is also freely available. For convenience, the routines are displayed in Table 2. As can be seen, there is a considerable reduction in complexity and the total number of lines (about 1220) is also much smaller, which makes for simplification.

The relevant algorithms have been described in some of the subsections above. Hence it suffices to comment briefly on aspects of input and output. To start with, the initial input line is reduced to just three parameters: the particle number, tolerance and termination time (in units of the crossing time). Otherwise the generation and treatment of the initial conditions are exactly as before. Relevant parameters are initialized and the chain vectors selected in the standard way.

The main question when using the basic code is what type of results should be produced. This is related to the purpose of the calculation and therefore difficult to anticipate. It is therefore mostly up to the user to design suitable data analysis. Procedures for X11 movie displays are available in a slightly larger code (c.f. `movie.tar.Z`), as well as the full chain code described above. It contains a reduction of the time-step and extra arithmetic to increase the viewing time. Alternatively, it is a simple matter to include a suitable graphics package with the compact version.

Only a few counters and the relative energy error are given at termination, as well as the final solutions. The CPU time for a standard few-body calculation is usually very short. Hence the question arises about automating the generation of initial conditions in order to explore parameter space. At this stage it may be advantageous to adopt the more versatile code where other features are available.

The code may be downloaded in the usual way and the routines obtained by the command '`tar xvf chain0.tar`'. After checking the `Makefile`, compilation is achieved by '`make chain0`' and the code is ready for testing.

5 Graphics

Some versions of *LINUX* fail to run when compiling with X11 graphics. The reason is a complaint about the colour table. A work-around was devised by Peter Teuben as follows:

```
CTRL-alt-F2
login
startx8
```

This script starts `xwindows` in 8-bit mode and looks like:

```
cd $HOME
cp .xinitrc8 .xinitrc
startx -- :1 -depth 8
rm .xinitrc
```

To go back, press `CTRL-alt-F7`.

The Makefile (also designed by P.J.T) allows compilation of two different executables, `xtriple, ptriple` or `xchain, pchain` by the command '`make all`'. Alternatively, each executable may be compiled separately. The *PGPLOT* graphics package can be downloaded from `http://www.astro.caltech.edu/~tjp/pgplot`.

6 Plummer potential

An optional external potential was included in early 2006. This necessitates reading the Plummer mass and softening as the second input line. The relevant perturbations are skipped on zero mass (cf. *IPERT*). The modified equations of motion in AZ (given in AA03) were implemented, together with integration of the c.m. which was not included at first. Even without the latter, successful time reversal was achieved and the triple code also showed detailed agreement with *CHAIN*. Eventually a comparison with Seppo Mikkola's version pointed to the omission and an important lesson was learnt concerning time reversal. In AZ there are now 24 equations to be integrated - an extra six for the c.m. as well as one for the energy. The Plummer model has also been included in *CHAIN*.

In the presence of an external field there is the choice of assigning initial conditions at an arbitrary position or transforming to the c.m. rest frame, as adopted here. If the former is preferred instead, the initialization to zero of the c.m. variables *CMX*, *CMV* at the end of routine *DATA* or *CHINIT* should be suppressed. Note that unlike in a harmonic potential, the c.m. force in a Plummer model does not remain zero for a system at rest even if we have the condition $\sum m_i \mathbf{r}_i = 0$. Consequently, all coordinates and velocities are expressed with respect to the origin. In both codes, formal energy conservation is checked by comparing the internal system energy (evaluated explicitly) with the integrated value, where the latter changes if external perturbations are applied.

7 Post-Newtonian terms

Simulations involving massive black holes are currently attracting much attention. Post-Newtonian terms were added to the code *TRIPLE2* as an exercise during the Cambody School in August 2006. Two extra parameters are now read on the same line as the Plummer model input: *CLIGHT* and an option *IPN* for including the post-Newtonian terms up to PN3. The former parameter denotes the speed of light (c) in model units such that relatively small values represent strong GR effects, while the Newtonian formulation is adopted above 10^5. Because of the high power of the dependence on separation, small changes to the value of c affect the time-scale for inspiralling considerably.

The parameter dependence for GR coalescence in a three-body system is most readily investigated for a dominant binary and distant perturber since otherwise different close encounters may affect the result spuriously. Following common practice we define gravitational coalescence between two bodies when the separation falls below 3 Schwarzschild radii, given by $R_{\rm coll} = 6(M_1 + M_2)/c^2$, whereupon the calculation is terminated. Note that the time for coalescence is somewhat less than the initial expression (given as *TZ*) because of the subsequent speeding up.

The routine *XTPERT* contains the relevant expressions for the post-Newtonian terms. This implementation is based on recent papers by Blanchet & Iyer and Mora & Will (see *XTPERT* for references) which split the contributions into two parts connected with the relative coordinates and velocity. While the GR perturbation is active there is a choice of including the so-called 'Mercury' (or classical) precession (*IPN* $= 1$) and also the more complicated second-order precession (*IPN* $= 2$) together with the relativistic radiation. However, only the latter gives rise to a secular energy change while the first two affect the perihelion motion, with minor energy fluctuations during each orbit. Relevant expressions for PN3 (*IPN* $= 3$) were added at a later stage, together with the corresponding two-body energies and PN eccentricity terms. Now the treatment also includes hyperbolic encounters. In a large simulation, the progressive inclusion of the respective terms according to time-scales may be justified with the higher orders added at a later stage.

The formulation allows an optional Plummer force to be included at the same time as the post-Newtonian expressions. An implementation of the three-body regularization code calls for a careful treatment of the perturbing force as applied to the dominant motion. Thus there are three contributions from the Plummer model but only two due to Relativity. Here the latter term is split into two parts according to the mass ratios $M_3/(M_k + M_3)$ and $M_k/(M_k + M_3)$ of the two participating bodies (M_k, M_3) such that the difference equals the full term provided by *XTPERT*. On the other hand, there is no contribution to the change in the local two-body energy from the Plummer mass distribution. Finally, the c.m. motion of the triple system is integrated according to the mass-weighted accelerations of the Plummer model since the GR effect is only internal.

An experimental scheme for decision-making has been implemented. Here the main idea is to switch on the post-Newtonian terms progressively, depending on the standard time-scale $a/\dot{a}$. However, the lowest value of the indicator *IPN* is specified as input.

SJA, 14/10/06

References

Aarseth, S.J. [2003], *Gravitational N-Body Simulations* (Cambridge University Press).

Aarseth, S.J. & Zare, K. [1974], 'A regularization of the three-body problem', *Celes. Mech.* **10**, 185–205.

Bulirsch, R. & Stoer, J. [1966], 'Numerical treatment of ordinary differential equations by extrapolation methods', *Num. Math.* **8**, 1–13.

Kustaanheimo, P. & Stiefel, E. [1965], 'Perturbation theory of Kepler motion based on spinor regularization', *J. Reine Angew. Math.* **218**, 204–19.

Mardling, R.A. [2008], MNRAS in preparation (see also Capri Proceedings).

Mikkola, S. and Aarseth, S.J. [1990], 'A chain regularization method for the few-body problem', *Celes. Mech. Dyn. Ast.* **47**, 375–90.

Mikkola, S. & Aarseth, S.J. [1993], 'An implementation of N-body chain regularization', *Celes. Mech. Dyn. Ast.* **57**, 439–59.

Mikkola, S. & Aarseth, S.J. [1996], 'A slow-down treatment for close binaries', *Celes. Mech. Dyn. Ast.* **64**, 197–208.

Table 2: *Basic CHAIN routines.*

Routine	Called by	Description
CHAIN		Main routine for chain regularization
CHINIT	*CHAIN*	Initial conditions & reduction to c.m. frame
CONST	*CHAIN*	Constants of motion
DERQP	*DIFSY1*	Derivatives of chain variables
DIFSY1	*CHAIN*	Bulirsch–Stoer integrator
HPSORT	*SELECT*	Heapsort algorithm of Press
KSPHYS	*SWITCH* & *TRANSX*	Transformation from KS to physical variables
PHYSKS	*SWITCH* & *TRANSQ*	Transformation from physical to KS variables
QFORCE	*DERQP*	Transpose of $\mathbf{Q}\cdot 2\mathbf{F}$
REDRAW	*SELECT*	Checking of chain connections
SELECT	*CHAIN* & *SWITCH*	Determination of chain indices
SWCOND	*CHAIN*	Switching condition check
SWITCH	*CHAIN*	Switching of chain
TRANSQ	*CHAIN*	Transformation from chain variables to KS
TRANSX	*CHAIN*	Transformation from KS to chain variables
VECTOR	*DERQP*	Matrix multiplication of $\mathbf{Q^T}\cdot\mathbf{P}$
YCOPY	*CHAIN* & *SWITCH*	Copy Y-array from COMMON
YSAVE	*CHAIN* & *SWITCH*	Save Y-array in COMMON

Table 3: CHAIN *routines.*

Routine	Called by	Description
BINARY	*CHAIN*	Final binary determination
CHAIN		Main routine for chain regularization
CHINIT	*CHAIN*	Initial conditions & reduction to c.m. frame
CHSTAB	*CHAIN*	Chain stability test (Mardling criterion; $N = 3$)
CONST	*CHAIN*	Constants of motion
CPUTIM	*CHAIN*	Elapsed CPU time (secs)
DERQP	*DIFSY1*	Derivatives of chain variables $\mathbf{Q}$, $\mathbf{P}$
DIFSY1	*CHAIN*	Bulirsch–Stoer integrator
EREL	*DERQP* & *SLOW*	Dominant two-body energy
ESCAPE	*CHAIN*	Removal of escaping particle
HPSORT	*SELECT* & *ESCAPE*	Heapsort algorithm of Press
INCLIN	*CHSTAB*	Inclination of hierarchical system
KSPHYS	*SWITCH* & *TRANSX*	Transformation from KS to physical variables
MOVIE0	*CHAIN*	Initialization of movie window
MOVIE	*CHAIN*	Movie display (X11)
NEWSYS	*CHAIN*	Inelastic two-body collision
PERI	*DERQP*	Determination of pericentre distance
PHYSKS	*SWITCH* & *TRANSQ*	Transformation from physical to KS variables
QFORCE	*DERQP*	Transpose of $\mathbf{Q}{\cdot}2\mathbf{F}$
R2SORT	*CHAIN* & *CHSTAB*	Sorting of square two-body separations
REDRAW	*SELECT*	Checking of chain connections
REDUCE	*ESCAPE* & *NEWSYS*	Reduction of chain member
SELECT	*CHAIN* & *SWITCH*	Determination of chain indices
SLOW	*CHAIN*	Slow-down treatment of binary
SWCOND	*CHAIN*	Switching condition check
SWITCH	*CHAIN*	Switching of chain
TPERI	*DERQP*	Pericentre time for two-body motion
TRANSK	*EREL*	Transformation to physical momenta & separations
TRANSQ	*CHAIN*	Transformation from chain variables to KS
TRANSX	*CHAIN*	Transformation from KS to chain variables
VECTOR	*DERQP*	Matrix multiplication of $\mathbf{Q}^{\mathbf{T}} \cdot \mathbf{P}$
XTF	*DERQP*	External chain forces (inactive)
XTPERT	*XTF*	External perturbation (inactive)
XTRNLU	*CHAIN*	Harmonic test potential (suppressed)
YCOPY	*CHAIN* & *SWITCH*	Copy Y-array from COMMON
YSAVE	*CHAIN* & *SWITCH*	Save Y-array in COMMON