NBODY6 User Manual

Sverre Aarseth

Email: sverre@ast.cam.ac.uk

Institute of Astronomy, University of Cambridge

1 Introduction

This is a User Manual for *NBODY6*, a code that has been extensively tested since it was first developed around 1992. This code is mainly intended for laptops and workstations with the `UNIX` operating system. A parallel implementation called *NBODY6++* for Beowulf PC clusters and the T3E supercomputer is also available [Spurzem *et al.* 2003].

The code relies on many features of the classical *NBODY5* which dates back to the late 1970s. The subsequent change from a divided difference formulation to Hermite integration [Makino 1991] led to a complete revision. Briefly stated, single particles and centre-of-mass systems are integrated by the Ahmad–Cohen [1973] neighbour scheme using the fourth-order Hermite method [Makino & Aarseth 1992] (for the quantized version see Aarseth 1999). Binaries and close two-body encounters are studied by the Stumpff version of Kustaanheimo–Stiefel [1964, KS] regularization [Mikkola & Aarseth 1998], while interactions of compact subsystems are described by the chain regularization method [Mikkola & Aarseth 1990, 1993, 1996]. Moreover, strong interactions in unperturbed triples and quadruples are treated by three-body [Aarseth & Zare 1974, AZ] and Heggie [1974] global regularization [Mikkola 1985]. Finally, hard triples and higher-order systems satisfying a stability criterion [Mardling & Aarseth 1999] are reduced to two-body configurations (so-called mergers as opposed to collisions). All the relevant equations of motion are derived and discussed extensively in a recent book [Aarseth 2003], together with algorithms which may be helpful when examining the *FORTRAN* procedures. Hence familiarity with the book is very beneficial for understanding the code.

Several aspects of synthetic stellar evolution, such as mass loss, tidal circularization and collisions have been incorporated while still maintaining energy conservation by means of appropriate correction procedures. The early scheme used fast fitting functions for the radii and luminosities of single stars of solar metallicity [Eggleton, Fitchett & Tout 1989], with imposed wind and supernova mass loss. Binary evolution and collisions were subsequently included [Tout *et al.* 1997]. More recently, a unique extension to lower metallicities was implemented [Hurley, Pols & Tout 2000]. In this connection note that currently the *GRAPE–6* code *NBODY4* contains several important astrophysical processes not yet included here (*e.g.* Roche lobe overflow and spin–orbit coupling). On the other hand, *NBODY6* contains new procedures for a general 3D galactic potential. The code itself can be downloaded from the web in the form of a compressed tar file.[1]

[1] http://www.ast.cam.ac.uk/∼sverre.

2 Code structure

The whole code consists of some 34,500 lines including comments and layout space. It is written in *FORTRAN* and is F77 compliant but also compiles with Intel or F95 (ignore warning messages about non-standard expressions). There are about 236 routines altogether, with mnemonic names of maximum six characters. Likewise, almost all the *FORTRAN* statements are in upper case while the comments are in lower case. The coding conforms to a strict style and layout for clarity.

The main code relies heavily on a general `common` block called `common6.h` which contains a large number of arrays (mostly size N) and many useful scalars. This enables a calculation to be split into several parts by saving all the `common` variables after a specified `CPU` time (or by a '`touch STOP`' facility at arbitrary times, see below), followed by a restart. Except for some special situations, this gives rise to reproducible results which are essential for experimental purposes as well as problem investigation.

The code consists of three main parts: input, output and integration, with the latter split into several large routines employing different methods. Thus a smooth running relies on treating a range of special cases by the appropriate algorithm. However, the decision-making requires very little overheads. A number of optional procedures are included but care is needed to avoid mutual inconsistencies since only some parameter values and no options are checked. Thus the main choice is between isolated point-mass calculations and realistic star cluster models which include relevant astrophysical processes.

3 Getting started

Once the file `nbody6.tar.Z` is downloaded and uncompressed, the routines are extracted by '`tar xvf nbody6.tar`' and copied to four directories for convenience: `Ncode, Chain, Nchain, Docs`. The first contains the main code, while `Chain` holds the basic chain procedures and `Nchain` deals with the corresponding N-body interface. A number of useful files, such as test input templates, listing in TeX of all routines and the structure of the old code *NBODY5* are included in the directory `Docs`.

The size of most large `common` arrays are given by the parameter file `params.h` and defined in Table 1. Depending on available memory, it is recommended to limit the maximum array sizes somewhat but still leave room for bigger calculations; this will facilitate examining `common` blocks using the same `read` statements for different memberships. Note that the file `params.h` in directory `Nchain` should be identical to that in the main directory. Once these parameters have been specified, the correct size of the `common` block is calculated automatically in routine `mydump.f`. The present parameters ($N_{\rm max} = 4010$) produce a `common` block of about 4 Mb.

Before compiling, check the *FORTRAN* directives in the `Makefile`, i.e. whether `f77` or `g77` and use the highest optimization level. Compile by the command '`make nbody6`' which should produce the executable `nbody6`. One possible reason for failure could be the `CPU` timer function `etime` in routine `cputim.f` which is system dependent. Any other complaints and strange run-time behaviour should be reported after making every effort to ascertain the problem. For frequent usage it is recommended to create separate working directories. It is also a good idea to save the original files when making changes.

To start a test run, place the executable and a template input file `input` in one directory and type the command '`nbody6 < input > output &`'. Although the results are machine- and compiler-dependent, it is expected that the run will finish normally but there is always a chance that some difficult configuration may occur.

The restart facility can be tested as follows. First specify option # 1 = 1 as input. This will ensure a `common` dump on `fort.1` if the `CPU` time is exceeded or `TIME >= TCRIT`, with `TCRIT` given in the input file. The calculation can then be continued from the `common` save by '`nbody6 < rs >` output2 &', using the restart input file '`KSTART TCOMP`'. Here `KSTART = 1` denotes a new run (followed by the required input) or = 2 for standard restart, and `TCOMP` is the `CPU` time in minutes. Also note the possibility of reading some new parameters at restart if `KSTART > 2` (see routine `modify.f` and the restart file `rs`). Before restarting in the same directory, any important output files in capitals (*e.g.* `ESC, OUT9`) must be renamed and/or deleted (likewise any fort.82 and fort.83).

The code includes an optional provision for automatic error checking. Thus if option # 2 = 2, any output interval with relative energy error $|\Delta E/E| > 5\,Q_{\rm E}$, with $Q_{\rm E}$ the tolerance, is restarted from the previous time with reduced values of the basic integration parameters η_I and η_R. If instead the energy error lies in the interval $[Q_{\rm E}, 5\,Q_{\rm E}]$, the accuracy parameters are reduced by an appropriate amount. Likewise, an increase (up to the initial values) is carried out for relative errors below $Q_{\rm E}/5$ (see routine `check.f`). Note the use of two options (with `common` saves on `fort.1` and `fort.2`) in order to employ the energy check independently of terminations at arbitrary times. Consequently, in the case of a halted calculation, the previous good `common` save must be copied from `fort.2` to `fort.1` before the restart (possibly with modified accuracy parameters).

4 Input parameters

A standard input file consists essentially of six lines defining the membership, accuracy and decision-making parameters, options, KS integration, IMF power-law exponent and virial theorem scaling. Some of these quantities are listed in Table 2 and routine `define.f` contains a complete listing. Many are dimensionless while others have an astrophysical meaning (see Aarseth 2001a for the neighbour scheme in *NBODY2*). Likewise, the options can be selected by consulting Table 3. Hence only a few values need to be changed in the input template once the first calculation has been performed. Input files for primordial binaries (`inbins`) and 3D tidal field (`inlog`) are included in the directory `Docs`.

For most purposes, and a wide range of particle numbers, the standard choice of the accuracy parameters η_I, η_R and η_U should suffice. Having decided on the membership N, a maximum neighbour number of $n_{\rm max} \simeq 2N^{1/2}$ is usually adequate in the absence of primordial binaries. One good strategy for choosing the output intervals is to adopt $\Delta t_{\rm adj} = 2.0$ for energy checks and escaper removal and take $\Delta t_{\rm out} = 10.0$ for main output and data analysis. Finally, the choice of the relative energy tolerance $Q_{\rm E}$ is a matter of taste and a slightly larger value than quoted in Table 2 may be acceptable.

5 Initial conditions

Different choices of initial conditions are available. These include the Plummer model, two orbiting clusters or a massive bound/unbound binary (`KZ(5) = 1, 2, 4`). Several types of IMF (Salpeter or Kroupa, Tout & Gilmore 1993) may be generated by option # 20. However, it is envisaged that the user will provide original initial conditions for new investigations ($m_i, \mathbf{r}_i, \mathbf{v}_i,\ i = 1, 2, .., N$). One possibility is to use the stand-alone code `aking6.f` and input `inking` for generating King models [Heggie & Ramamani 1995] (no warranty!) included in the directory `Docs`. A package for producing initially rotating cluster models is also available [Spurzem & Einsel 2002, private communication].

The specially generated initial distribution is read from `fort.10` with option # $22 = 2$ before scaling to standard internal units of total energy -0.25 (for bound clusters) and $\sum m_i = 1$. Unless special precaution is taken, the same scaling is applied generally. Thus in scaled units and overall virial equilibrium the mean square velocity is $< v^2 >= 1/2$ and the crossing time (denoted `TCR`) is $t_{\rm cr} = 2\sqrt{2}$, independent of N.

With the initial length unit `RBAR` specified in pc and the mean mass `ZMBAR` (which may not be preserved with option # 20 >= 2) in $M_\odot$, any internal quantity may be converted to astrophysical values. A convenient set of conversion factors is given by `RBAR, SMU, VSTAR, TSTAR` for distances (pc), masses ($M_\odot$), velocities (km sec^{-1}) and times (in Myr). Binary periods in years or days can be obtained from the N-body units using `YRS` or `DAYS` in the standard Keplerian expression without 2π. Moreover, length scales may also readily be converted to solar radii (`SU`) or astronomical units (`AU`).

6 Decision-making

Although individual time-steps are used in the general integration, the Hermite scheme employs discrete block-steps which enables many particles to be advanced in tandem. Consequently the main integration cycle comprises a small number of calls to the next level of routines, with a few other important tasks performed either before or at the end. This structure facilitates the investigation of any strange behaviour such as infinite looping since the offending routine can be identified, whereupon the principle of bisecting can often be applied to locate the problem.

Essentially the integration cycle consists of the following procedures:

- Determine the next block of particles due for advancement
- Update any close encounter solutions (KS and/or chain)
- Predict coordinates and velocities (neighbours or full N)
- Advance the solution for each body (neighbour force or total force)
- Deal with any close encounter terminations (KS, chain or mergers)
- Check stellar evolution (updating of radii or mass loss)

The strategy for the first step relies on the simple device of determining the smallest value of $t + \Delta t_i$ for the members in the block, denoted by $t_{\rm new}$, where t is the current time. The corresponding time-steps are obtained by a relative force criterion of the form

$$\Delta t_i = \left(\frac{\eta |\mathbf{F}_i|}{|\ddot{\mathbf{F}}_i|} \right)^{1/2}, \tag{1}$$

where η is a small dimensionless constant ensuring convergence. To fit in with the more efficient block-step scheme, the 'natural' step is truncated (or quantized) to the nearest factor 2 value commensurate with `TIME`. This requirement implies that $\texttt{mod(TIME,}\Delta t_i) = 0$ for both the old and new time-step at the updating of a Hermite solution. A more sensitive expression is obtained by including the other force derivatives. Given a list of particles that are due for treatment in some larger interval, the actual block-step members are determined by identifying those with $T_0 + \Delta t_j = t_{\rm new}$, where T_0 is the time of the last neighbour (so-called irregular) force evaluation.

The basic integration cycle is given by the following lines of code:

```
*       Advance the pointer (<= NXTLEN) and select next particle index.
   50 LI = LI + 1
      IF (LI.GT.NXTLEN) GO TO 1
      I = NXTLST(LI)
      TIME = T0(I) + STEP(I)
*
*       See whether the regular force needs to be updated (IR > 0).
      IF (T0R(I) + STEPR(I).LE.TIME) THEN
          IR = 1
      ELSE
          IR = 0
      END IF
*
*       Advance the irregular step.
      CALL NBINT(I,IKS,IR,XI,XIDOT)
*
*       See whether the regular step is due.
      IF (IR.GT.0) THEN
          CALL REGINT(I,XI,XIDOT)
      END IF
*
*       Determine next block time (note STEP may shrink in REGINT).
      TMIN = MIN(TIMENW(I),TMIN)
```

As discussed in subsequent sections, the special case of KS and chain regularization merits special treatment. Following the relevant predictions, new solutions of the irregular and (if required) regular force polynomials are obtained for each particle in turn. After all the members on the current block have been advanced, procedures relating to any terminations are carried out. This may necessitate an occasional return to the main routine for the specified task, defined by the flow control indicator (cf. Table 5). The last step of the main cycle deals with the optional stellar evolution and is only examined at small quantized intervals of $\leq 10^3$ yr.

7 Data management

It is the purpose of a code to produce results. However, there is a wide choice of quantities that can be constructed from the basic particle data. Here a dual-purpose strategy has been adopted and if this is not convenient, special data analysis can readily be included either during integration or at output times (i.e. called from routine `adjust.f`).

The data structure itself forms the backbone of the code and will be summarized first (also see book p.117). Given a number of KS solutions $N_{\rm p}$, the particle arrays for the components of each pair $I_{\rm p}$ are placed in locations $2I_{\rm p}-1$ and $2I_{\rm p}$, with the corresponding centre of mass (c.m.) in $N + I_{\rm p}$. Hence all KS pairs appear contiguously in the order in which they are initialized. This scheme facilitates a sequential treatment of all the particles, with force summations and neighbour lists referring to increasing locations in $[2N_{\rm p}+1,\ N+N_{\rm p}]$. It entails relabelling particle references in neighbour or perturber lists after each new or terminated KS pair but is a small cost for preserving simplicity. The structure of a given list for particle i is of the form `LIST(k,i)`, where $k = 1$ is used for the neighbour number. The neighbour strategy in *NBODY2* is similar [Aarseth 2001a]. In the case of force summations involving neighbours with $j > N$, the c.m. is replaced by the components (which must be obtained by a KS transformation) if the c.m. approximation is not satisfied.

In order to introduce more complex systems, such as stable triples or temporary chain subsystems, the device of 'ghost' particles of zero mass is used (see below). This facilitates recovery of the original state without affecting the overall data structure. Pointers to the individual members are saved and any relevant information may be obtained as required, although the identification of multiple hierarchies is non-trivial (see Appendix C of book).

Two types of output are produced. Important results are presented in file names using capitals, while supplementary diagnostics appear in `fort.n` with $n > 2$. The main files provide the following information in mathematical notation:

- `ESC` $t\,({\rm Myr})$, $m\,(M_\odot)$, $v^2/<v^2>$, v $({\rm km\,s^{-1}})$, k^*, $\mathcal{N}_i$ for each escaper
- `OUT9` $E_{\rm b}$, e, $E_{\rm cm}$, r, m_k, m_l, $P\,({\rm days})$, $\mathcal{N}_k$, $\mathcal{N}_l$, k_k^*, k_l^* for each KS binary
- `HIARCH` t, a_0, a_1, e_1, $a_1(1-e_1)$, P_1/P_0, $m_{\rm bin}/m_3$, $R_{\rm pcrit}$, $m_{\rm bin}$, m_1/m_2, $\mathcal{N}_{1,2,3}$
- `HIDAT` $\mathcal{N}_k$, $\mathcal{N}_l$, $\mathcal{N}_m$, k^*, m_1, m_2, m_3, r, $e_{\rm max}$, e_0, e_1, P_0, P_1 for hierarchies
- `OUT3` m, x, y, z, $\dot{x}$, $\dot{y}$, $\dot{z}$, $\mathcal{N}_i$ for $i = [1,\ N+N_{\rm p}]$ (binary format)

Here $\mathcal{N}_i$ represent original particle names, $E_{\rm cm}$ is the specific c.m. binding energy, P denotes the period, k^* is the stellar type, $a_1(1 - e_1)$ is the outer pericentre and $R_{\rm pcrit}$ the corresponding stability boundary. The maximum eccentricity in the Kozai cycle is given by $e_{\rm max}$, while e_0 and e_1 refer to the respective eccentricities. Note that the file `HIARCH` contains information on the formation and termination of hierarchical systems whereas `HIDAT` provides a summary at each main output. All the above are optional (cf. the multi-valued # 8) and the file `OUT3` is the data bank which may be produced at main output with specified frequency. Because the binary format is currently saved in single precision, subsequent reading and analysis of this data must be consistent. Note the use of *two* records for each output which enables arrays of variable length to be read.

A variety of results also appear as standard output. This takes the form of an error check at intervals $\Delta t_{\rm adj}$, (with `DE` = `ΔE/E`) when the density centre is updated and escapers removed. Considerably more information is given at intervals $\Delta t_{\rm out}$ which are best kept commensurate with $\Delta t_{\rm adj}$. In this connection, note the facility (# 32) to increase the output intervals if the energy binding the cluster changes by factors of 2.

We summarize some of the most important quantities given at main output. This information is organized in distinct (optional) groups. The first line gives the particle number, average neighbour number (< `NNB` >), KS solutions (`KS`), number of mergers (`NM` and `MM`, standard and higher order) and single stars (`NS`). Among useful quantities describing the cluster evolution are the half-mass radius (< `R` >), tidal radius (`RTIDE`), core radius (`RC`) and membership (`NC`), energy in binaries and mergers (`EB/E` and `EM/E`) and the time in Myr (`T6`). Escapers are removed outside `2*RTIDE` with option # 23.

If # 8 is active, there is a summary of original and exchanged binaries, the average and maximum eccentricity (< `E` > and `EMAX`), as well as distributions of stellar population types and binary binding energies. The energy budget is also summarized. For historical reasons, the energy binding the cluster is saved in `E(3)`, while `E(1)` and `E(2)` give the energy of primordial and dynamically formed binaries. For the definition of the total energy see Eq. (9.29) of the book. A large number of counters are displayed; see Table 4 for a list of the most important. In large calculations, some time-step counters may exceed the integer limit and are reset to zero above 2×10^9.

Finally, note the optional time offset provision (#35) which prevents large values of `TIME/STEP`. Thus if the limit `DTOFF` is exceeded, `TIME` is reset to zero and `TOFF` is used to obtain the total time `TTOT`. This is only of practical relevance when printing the current value of the time.

The question of adding extra variables to the `common` blocks often arises. Here we mention two useful strategies. The simplest case is to introduce a new labelled `common` in the header file `common6.h` so that these variables become available in most routines. At the same time, the size of the `common` save must be increased accordingly (cf. routine `mydump.f`) and the whole code recompiled. A slight drawback is that old `common` saves cannot be read in the usual way. The alternative is to create dummy variables in an existing labelled `common` which can be used for newly created variables without affecting the overall size. However, the latter method anticipates future demands.

8 Two-body regularization

The KS method first appeared in a standard N-body code in time for the Cambridge IAU Colloquium # 10 [Aarseth 1972]. It was an instant success and has proved a mainstay for treating binaries and close two-body encounters ever since. Like any versatile algorithm, it has progressed through several distinct versions until ending up with the highly accurate Stumpff formulation [Mikkola & Aarseth 1998]. Although the underlying mathematics is very precise, its implementation is something of a black art, employing a variety of heuristic procedures. This is particularly the case when dealing with more complicated interactions in compact subsystems.

An arbitrary number of KS solutions are treated at the same time. Decision-making is essentially controlled by two input parameters which are modified at each output if option # 16 is active. A search for close encounter candidates is carried out if the time-step becomes suitably small; i.e. $\Delta t_i < \Delta t_{\rm cl}$, subject to the distance test $R < R_{\rm cl}$ in order to ensure dominant two-body motion (see book for definitions). If necessary, the actual regularization is delayed until both the components are advanced to the same time. Note that the relative time-step criterion gives very similar values for different masses during close encounters. Implementations of the KS transformations and polynomial initialization have been described in considerable detail elsewhere (Aarseth 1985, 2001b, 2003). It is therefore sufficient to discuss some aspects of the decision-making.[2]

As noted in a previous section, all relevant KS solutions are considered at the start of the integration cycle and not advanced beyond the end of the block-step. For this purpose, the regularized time-step is converted to physical units (book Eq. (11.1)). In practice most primordial binaries are unperturbed, with time intervals often exceeding the typical block-step, and an efficient sorting and insert procedure is employed (see book p.143). The exceptional case of a physical collision is treated differently and requires a new block-step to be created (for a discussion of time quantization see book Eq. (12.16)). An algorithm for specifying intervals of unperturbed motion is also given in the book.

Termination is essentially controlled by the relative perturbation, except that soft binaries and hyperbolic flybys are also subject to a distance test in terms of the initial separation. The strategy for terminating a strongly perturbed hard binary depends on the suitability of a switch to chain regularization. The ideal case is that the latter treatment may be adopted but there are many situations when such configurations are not sufficiently compact (see below). Hence we may instead have repeated switching of particle pairs according to dominant two-body solutions, which is less satisfactory.

Following a KS termination, new polynomials and time-steps are assigned to the components. All relevant neighbour and perturber lists must also be updated in order to be consistent with the new particle sequence. However, the latter task deals with integer arithmetic and is quite fast. Since the c.m. time-step is inevitably small for strong perturbations, the new steps may be comparable and hence do not result in smaller block-steps. In any case, small values may increase rapidly if the situation permits (i.e. doubling every other step). Finally, we emphasize that the use of regularization places a lower limit on the Hermite time-steps, thereby saving significant computational efforts.

[2]The data structure is described in section 7.

9 Hierarchical systems

The presence of binaries often lead to the formation of long-lived hierarchies. The computational requirements for direct integration of the inner binary may be quite severe; yet the semi-major axis hardly changes even for modest distance ratios. Since this is the most important binary element, it would seem justified to adopt the c.m. approximation and recast the KS solution for the outer component, thereby increasing the period and replacing one direct integration. This procedure neglects short-term fluctuations and assumes no secular change, in qualitative agreement with first-order perturbation theory.

Although simple stability criteria were already introduced in the mid 1980s, the chaos approach provides greater theoretical justification and has given rise to a semi-analytical criterion [Mardling & Aarseth 1999]. Extensive tests of triple systems have confirmed the practical usefulness of the criterion for a wide range of masses and outer eccentricities, while the enhanced stability of inclined orbits is accounted for by a small correction factor. We also mention that the code *NBODY4* includes an averaging method to model the eccentricity oscillations (Kozai cycles) induced by high inclinations.

The identification of suitable configurations is based on small c.m. time-steps and the first part of the procedure employs the same algorithm as for chain regularization (see below). Subsequently, the division of labour is essentially made by comparing the outer pericentre, $R_{\rm p} = a_1(1 - e_1)$, with the inner semi-major axis, a_0. Roughly speaking, the case $R_{\rm p} > 3a_0$ justifies a search for stable systems. Further conditions, such as the requirement of the outer binary being hard and not too strongly perturbed must also be satisfied in addition to the stability test.

A new hierarchy is initialized in a similar way as for standard KS. However, when examining the data structure we distinguish between the outer component being a single particle or binary. In the case of a triple, the outer body is defined as a ghost particle after combining it with the two KS components of the old pair. However, with a second binary, both the KS solution and associated c.m. are made inactive by prescribing large values of T_0 and setting the relevant masses to zero. Here the latter plays the role of the ghost particle for triples and once the c.m. location $j > N$ has been identified from the ghost name, the pair index is simply given by $I_{\rm p} = j - N$. For stability purposes, triples may be considered as degenerate quadruples and a small correction term is included for the smallest binary. Although rare, higher-order systems also occur and are treated in an analogous manner (their data structure is discussed in the book).

Further stability checks are made at each apocentre passage because the outer orbit may have changed due to perturbations. Mass loss from the inner binary reduces the spacing ratio and necessitates a stability test which is performed *in situ*. Following termination of a triple, the inner binary is initialized as a KS solution while the outer component already has the standard form (but is not in the correct location). In the case of a quadruple, the second KS pair and ghost c.m. are initialized *in situ* in the usual way after the neighbour list is formed. The force discontinuity arising from the changed configurations is handled in several ways with appropriate differential corrections to the energy budget. Note that the initialization and reconstruction of hierarchies employ the small labelled `common` block *BINARY* which contains original masses and particle names (as well as the basic KS variables for any second pair).

A new stability criterion has now been developed from first principles [Mardling 2003]. This formulation includes the effect of the inner eccentricity as well as the inclination and is also appropriate for planetary systems. Although the stability boundaries are qualitatively similar, the new criterion is more general. Hopefully this implementation will be available in a future code release.

10 Chain regularization

Several methods are available for treating strong interactions involving more than two particles. The original AZ three-body regularization was implemented first, followed by Heggie's global method for four particles [cf. Aarseth 1985]. For simplicity, these formulations did not include external perturbations. The versatile chain regularization [Mikkola & Aarseth 1990, 1993, 1996] which includes perturbations has proved more effective. Because of the increased complexity, the relevant routines are placed in the separate directories `Chain` and `Nchain`. Since only one configuration at a time is considered at present, the two unperturbed treatments are still maintained but rarely needed.

Close multiple encounters are characterized by one or more small c.m. time-steps. It is therefore convenient to perform the main decision-making for initiating multiple regularization at apocentre (defined as the turning point of radial velocity when $R > a$). In the case of a single particle approaching a hard binary, the condition for acceptance depends on the separation as well as the impact parameter. Thus too long delay leads to KS termination by large perturbation, while wide separations may be inefficient (unless for ultra-compact configurations). Algorithms for initialization and integration of chain subsystems are described in the book, together with specific procedures for changing the membership (reduction or increase). Such interactions tend to be short lived. Note the importance of including various stability tests (see above) since the ejection of a particle may otherwise produce a long-lived hierarchical system which can be studied more efficiently in another way.

Termination usually occurs when the third body escapes or two surviving binaries become well separated and are more suitable for KS treatment. The initialization of one or two KS solutions is carried out at termination rather than in the usual way during integration. A more difficult situation arises when the most distant particle is still bound to the subsystem. It then becomes a question of whether to retain such a member as part of the chain or include its effect as a strong perturber. In this connection, note that the system size is used for perturber selection and also that the perturbers are predicted at every function evaluation (i.e. many times per step) by the Bulirsch–Stoer [1966] integrator. Both the close encounter separation and the characteristic gravitational radius (defined by the mass products and total energy) are used to limit the system size.

The data structure of the basic chain algorithm employs quantities expressed in the local c.m. frame. Transformations to global values must therefore be made before including the effect of perturbers, and likewise at termination. Since the device of ghost particles is adopted to preserve the sequential arrangement, the actual masses are saved together with the corresponding names. It is then a simple matter to recover the relevant global locations for initialization. In the case of three initial members, the binary components are placed in the first two single particle locations, as for standard KS termination, while the third body is turned into a ghost particle *in situ*. The termination of two binaries, on the other hand, yields the initial members in the first four locations. However, this arrangement cannot be assumed to persist since a new KS may be formed during the intervening interval and in any case, a four-body system may be reduced by escape. Another point worth noting is that the particle assigned as the c.m. may in fact escape, in which case a new reference body is determined. We also mention that the current scheme now caters for up to six members, although more than four is quite rare.

The treatment of the chain c.m. requires special care. Thus the force and its first derivative are first obtained in the usual way, whereupon differential corrections are added. This entails subtracting the standard contributions from any perturbers and adding the respective individual mass-weighted terms. For consistency, similar corrections are carried out when dealing with the perturbers. Again the overheads of checking the neighbour lists for identification is modest compared with the actual function evaluations. Moreover, note that the accumulated duration of all the chain regularizations only represents a small fraction of the total time.

Some comments on the time management may be helpful. The choice of intervals for the internal integration is based on the principle of convergence for coordinate prediction. Essentially the maximum interval for advancing the solutions is determined by examining $T_0 + \Delta t_j$ for the relevant perturbers as well as the c.m. itself. Typically, the Bulirsch–Stoer time-steps are somewhat smaller than this interval. An inversion from physical to regularized time usually ensures that the maximum is barely exceeded (see book for the algorithm). Additional topics, such as slow-down and quantization of time are also discussed extensively in the book.

11 Stellar evolution

For greater realism, the code includes several options for mass loss and finite-size effects. The treatment of stellar evolution is based on fast look-up functions which provide information on the stellar type, radius as well as core mass for a given initial mass, age and metallicity [Tout *et al.* 1997, Hurley *et al.* 2000]. The current position in the HR diagram is checked at frequent intervals determined by the evolution rate and the remaining time of each characteristic stage. A list of stellar evolution indicators k^* is included in routine `define.f` and two look-up times (i.e. previous and next value) are used for decision-making. Reimers-type wind loss is adopted in addition to supernova events, where the latter result in neutron star or even black hole formation.

For convenience, mass loss corrections are implemented when the accumulated wind loss exceeds 1%. This entails modifying the force and first derivative of each neighbour, as well as subtracting the potential energy change assuming instantaneous mass loss from the cluster. In the case of KS solutions, the orbit is expanded at constant eccentricity together with force updating of the c.m. neighbours. Neutron stars are currently assigned a kick velocity sampled from a Maxwellian with relatively large dispersion [cf. Hansen & Phinney 1997] which usually leads to disruption of close binaries and escape of single stars. An optional procedure (# 18) is included in order to avoid a sudden close approach by high-velocity particles.

The code contains a simple recipe for physical collisions, implemented for KS solutions and all multiple regularizations. At present the masses are combined inelastically without invoking common envelope evolution, as in *NBODY4*, although the evolutionary time-scale (i.e. age) is updated for main-sequence or giant stars. In the case of a KS pair, the new c.m. body is initialized as a single particle and the second component turned into a massless escaper. With chain regularization, an iterative procedure is used to determine the exact pericentre if the closest two-body separation lies inside a suitably small value. The energy of the collision pair is evaluated indirectly using well defined variables rather than from the two-body elements. For more than three chain members, the membership is reduced and the calculation continued, otherwise standard termination follows.

Finally, we mention optional tidal circularization (# 27). For simplicity, a sequential procedure is employed [Portegies Zwart *et al.* 1997]. Here discrete changes of the orbital elements a, e are implemented, whereas *NBODY4* also treats continuous circularization [Mardling & Aarseth 2001]. Relevant modifications of the KS variables are carried out if $a(1-e) < 4r_1^*$, with r_1^* the largest radius (see book for detailed algorithm). Several adjustments are usually made because the stellar radii tend to increase with time. Note that for very large eccentricities, angular momentum conservation gives rise to considerable shrinkage. One way to reach large eccentricity is for the inner binary to experience favourable Kozai oscillations.

12 External fields

The code includes two types of external tidal field which will be described (for details see chapter 8 of book). Linearized equations are appropriate for nearly circular orbits with small vertical displacements and are therefore suitable for simulating open clusters. Two optional variants are available: (i) # 14 = 1 for the standard case based on Oort's constants, and (ii) # 14 = 2 which applies to a galactic point mass (also linearized).

The relevant tidal terms (converted from the length unit `RBAR` and total mass) are initialized in routine `xtrnl0.f` and the perturbations are added in `xtrnlf.f` and also in `xtrnlp.f` for KS. The additional terms in the equations of motion are simple and enables explicit force derivatives to be employed for the Hermite integration. The corresponding contributions to the total energy are included (cf. `ETIDE` in `xtrnlv.f`), thereby facilitating conservation. However, the linearized form of the equations of motion are not appropriate well outside the tidal radius. Hence a more detailed exploration of ejected stars is restricted to modest distances.

In order to broaden the scope for studying more general cluster motion, a full 3D galactic model has been implemented (# 14 = 3). Note that the external effect is only included in `xtrnlf.f` for the direct integration, consistent with treating binaries in the c.m. approximation here. We adopt a composite galaxy model consisting of three components: (i) central point mass, (ii) Miyamoto–Nagai (1975) disk, and (iii) logarithmic potential. This model gives a good representation of the Galaxy and satisfies the requirement of being computationally convenient for the Hermite scheme. In order to provide more flexibility, any combination of the components may be used by varying some of the input parameters (cf. routine `xtrnl0.f` and the special input template in directory `Docs`).

The tidal effect on cluster members is obtained by including the differential force of the Galaxy with respect to the cluster centre. This necessitates integrating the cluster guiding centre as a point mass orbiting the Galaxy (routines `gcinit.f` and `gcint.f`), using the total force functions. Consequently, the cluster motion is known to high accuracy throughout the calculation. As before, distant stars may be considered as escapers and are removed if # 23 > 0. Unless specified initially, a value `RTIDE` = 50 pc is adopted for the tidal radius. Alternatively, the ejected members may be added to a test population forming the tidal tail and integrated by a fast method (see below).

The equations of motion for the full galactic potential do not admit an energy integral as in the linearized case. This raises the question of an alternative way of accounting for the tidal energy change. From general principles, this contribution is provided by the integral of $\dot{w} = \mathbf{v} \cdot \mathbf{P}$, where $\mathbf{P}$ is the tidal force. Likewise, the terms for $\ddot{w}$ are available using the galactic force derivative. This enables a Taylor series solution to be obtained at the end of each regular step. The third-order terms are incomplete and their inclusion gave worse results. The individual mass-weighted terms are accumulated (cf. `ETIDE` in `regint.f`) and included in the expression for the total energy. Note that small output intervals (i.e. < 1) may introduce apparent spurious energy errors. So far, experience with this scheme has been favourable.

It may also be of interest to study a star cluster embedded in gas, with the possibility of including time-dependent decay. A Plummer sphere coinciding with the centre of mass has been adopted as an optional feature (# 14 = 3 or 4) which can be employed independently. Three extra input parameters are required (cf. routine `xtrnl0.f` and `define.f`), namely the total mass and length scale (saved as square) as well as the decay time for gas expulsion [Kroupa *et al.* 2001], all in N-body units.

The equations of motion are modified analogous to the case of linear external perturbation by including the force components in the regular polynomial. Likewise, the relative perturbations for KS solutions (force and its derivative) are evaluated in simplified form assuming the same softened central distance. The escape criterion is modified to include the background potential and likewise the total energy correction. Another aspect needing attention is mass loss due to evolving stars (routines `ficorr.f` and `fcorr.f`). Again the correction procedure is similar to the standard case (# 14 = 1).

Since the standard definitions of crossing time and virial ratio are not suitable in this formulation, alternative expressions are used. The former is now given by $2R_{\rm h}/V$, with $R_{\rm h}$ the half-mass radius and V the rms velocity. A consistent expression for the latter can be derived from first principles (based on summing $m\mathbf{r} \cdot \mathbf{F}$) which leads to combining the potential and virial energies in the denominator. Finally, total energy conservation is achieved by adding the sum of tidal energy contributions in the usual way at output time (cf. `ETIDE` in routine `adjust.f`).

13 Tidal tails

Procedures for studying the growth of tidal tails by fast integration have been added to the code. This facility can be particularly useful for cluster orbits in the 3D galactic potential since the tidal energy corrections may become less accurate at large distances. Consequently, a substantial speed-up may be achieved for large systems. The basic idea of the scheme is to employ the standard variables, with the tidal tail particles saved in unused parts of the arrays (which necessitates making a suitable declaration of maximum array size). In fact, the standard size of the `common` blocks remains unchanged, with four counters or pointers replacing part of a redundant integer array. There are only three new routines of modest size and another five are modified slightly.

Each tidal tail member is initialized for integration at the time of escaper removal, i.e. outside a distance `2*RTIDE` where `RTIDE` is specified (in N-body units) at input if non-zero. Data for the first member is saved in location `ITAIL0 = NZERO +` min`(KMAX,NBIN0+10)`, with `NBIN0` the initial number of primordial binaries. This is safely above the largest value used for direct integration. Routine `tail0.f` increases the membership and copies the current coordinates and velocities to appropriate locations, expressed with respect to the Galactic Centre. New time-steps are then assigned after obtaining the force and first derivative, whereupon the integration variables are initialized in the usual way. The membership is denoted by `NTAIL` and any loop over the whole population is made from `I = ITAIL0` to `NTTOT = ITAIL0 + NTAIL - 1`. Thus all stars in the tidal tail would require array sizes (i.e. `NMAX`) of at most `2*NZERO + KMAX`, depending on the number of primordial binaries.

Since the integration is done in the Galactic frame, the corresponding force and first derivative are obtained by the usual expressions without differential correction (routine `xtrnlt.f`). This approach assumes a full galactic model (i.e. # 14 = 3) but a linearized tidal field can readily be included if local coordinates are used instead. The integration itself is performed by routine `ntint.f`, called from `intgrt.f` at the end of each block-step for any member satisfying the usual condition $t_0 + \Delta t \leq t$. The algoritm is a simplified version of `nbint.f`, with standard prediction added. Note that the time-steps are usually of the maximum size, currently 1.0 time units and quantized values are used for a uniform treatment. This ensures that the data are available with the highest accuracy at main output times.

Some simple additions to other routines facilitate the decision-making. Although no new options are required, the tidal tail integration is initiated with option # 23 $\geq$ 3 and # 14 = 3. Any relevant treatment is therefore executed for an existing population, i.e. `NTAIL` > 0. Finally, option # 3 is used to control the output of results for data analysis and plotting. A value of 1 gives rise to the standard output file `OUT3` in binary format. Various multiple choices are available, with # 3 = 4 producing a formatted file `OUT34` in astrophysical units (pc and km s^{-1}) which contain all the stars with respect to the density centre (`ASCI` format), while # 3 = 5 also yields `OUT3`. A small header contains both memberships and the time in Myr.

14 Numerical problems

A code is never fully tested and the range of initial conditions may be much wider than has been considered so far. The question of validity is not an easy one, and there are also accuracy issues in spite of careful treatment.

Consider the simple case of an intermediate-mass cluster (say $N \simeq 500$) with realistic mass spectrum and the recommended maximum neighbour number, $2N^{1/2}$. Thus a massive wide KS binary may introduce significant errors because a perturber list containing all the neighbours would be too small; i.e. $a(1+e)/\gamma_{\rm min}^{1/3} > R_{\rm s}$ (with $R_{\rm s}$ the neighbour radius) and yet the binding energy may be large. This situation improves with increasing N since we have $R_{\rm cl} \propto 1/N$ while the interparticle separation goes as $1/N^{1/3}$.

Another difficult condition arises for borderline cases of hierarchies which may be long-lived but not accepted as stable. If it occurs, frequent switching of solutions also leads to inefficiency as well as loss of accuracy. It goes without saying that one cannot anticipate future problems since so much depends on the type of investigation. However, in general, the numerical task gets harder during advanced stages of evolution when the central density is smaller and more complex hierarchical configurations are formed. In order to investigate a given problem, it is useful to halt the calculation before including some diagnostics. This can be achieved by typing '`touch STOP`' at an arbitrary time, whereupon a `common` save will occur on `fort.1`, from which a restart can be made.

Although the external energy binding the cluster may be feeble, superhard binaries tend to have strong interactions enhanced by their gravitational focusing. Even if not present initially, such binaries may form via tidal circularization. In any case, some shrinkage by dynamical means is possible before ejection by recoil intervenes. Thus a small actual deviation of the total energy may give rise to a disproportionately large relative error because the external energy appears in the denominator. Finally, in this connection, note that the accumulated energy change is given at the end of a calculation.

15 Acknowledgements

The following colleagues have made substantial contributions to the code:

- Seppo Mikkola — Two-body and chain regularization
- Chris Tout & Jarrod Hurley — Stellar evolution and collisions
- Rosemary Mardling — Stability of hierarchical systems

References

Aarseth, S.J. [1972], 'Direct integration methods for the N-body problem', in *Gravitational N-Body Problem* ed. M. Lecar (D. Reidel), 373–87.

Aarseth, S.J. [1985], 'Direct methods for N-body simulations', in *Multiple Time Scales* ed. J.U. Brackbill & B.I. Cohen (Academic Press), 377–418.

Aarseth, S.J. [1999], 'From NBODY1 to NBODY6: the growth of an industry', *PASP* **111**, 1333–46.

Aarseth, S.J. [2001a], 'NBODY2: a direct N-body integration code', *New Astron.* **6**, 277–91.

Aarseth, S.J. [2001b], 'Regularization methods for the N-body problem', in *The Restless Universe*, ed. B.A. Steves & A.J. Maciejewski (Inst. Phys. Publ.), 93–108.

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.

Ahmad, A. & Cohen, L. [1973], 'A numerical integration scheme for the N-body gravitational problem', *J. Comput. Phys.* **12**, 389–402.

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

Eggleton, P.P., Fitchett, M.J. & Tout, C.A. [1989], 'The distribution of visual binaries with two bright components', *Astrophys. J.* **347**, 998-1012. (Also see Errata in *Astrophys. J.* **354**, 387.)

Hansen, B.M.S. & Phinney, E.S. [1997], 'The pulsar kick velocity distribution', *Mon. Not. R. Astron. Soc.* **291**, 569–77.

Heggie, D.C. [1974], 'A global regularisation of the gravitational N-body problem', *Celes. Mech.* **10**, 217–41.

Heggie, D.C. & Ramamani, N. [1995], 'Approximate self-consistent models for tidally truncated star clusters', *Mon. Not. R. Astron. Soc.* **272**, 317–22.

Hurley, J.R., Pols, O.R. & Tout, C.A. [2000], 'Comprehensive analytical formulae for stellar evolution as a function of mass and metallicity', *Mon. Not. R. Astron. Soc.* **315**, 543–69.

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

Kroupa, P., Tout, C.A. & Gilmore, G. [1993], 'The distribution of low-mass stars in the Galactic disc', *Mon. Not. R. Astron. Soc.* **262**, 545–87.

Kroupa, P., Aarseth, S.J. & Hurley, J. [2001], 'The formation of a bound star cluster: from the Orion Cluster to the Pleiades', *Mon. Not. R. Astron. Soc.* **321**, 699–712.

Makino, J. [1991], 'Optimal order and time-step criterion for Aarseth-type N-body integrators', *Astrophys. J.* **369**, 200–12.

Makino, J. & Aarseth, S.J. [1992], 'On a Hermite integrator with Ahmad–Cohen scheme for gravitational many-body problems', *Publ. Astron. Soc. Japan* **44**, 141–51.

Mardling, R.A. [2003], 'A new three-body formalism', in preparation.

Mardling, R.A. & Aarseth, S.J. [1999], 'Dynamics and stability of three-body systems', in *The Dynamics of Small Bodies in the Solar System*, ed. B.A. Steves & A. Roy (Kluwer), 385–92.

Mardling, R.A. & Aarseth, S.J. [2001], 'Tidal interactions in star cluster simulations', *Mon. Not. R. Astron. Soc.* **321**, 398–420.

Mikkola, S. [1985], 'A practical and regular formulation of the N-body equations', *Mon. Not. R. Astron. Soc.* **215**, 171–7.

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.

Mikkola, S. & Aarseth, S.J. [1998], 'An efficient integration method for binaries in N-body simulations', *New Astron.* **3**, 309–20.

Miyamoto, M. & Nagai, R. [1975], 'Three-dimensional models for the distribution of mass in galaxies', *Publ. Astron. Soc. Japan* **27**, 533–43.

Portegies Zwart, S.F., Hut, P., McMillan, S.L.W. & Verbunt, F. [1997], 'Star cluster ecology, II. Binary evolution with single-star encounters', *Astron. Astrophys.* **328**, 143–57.

Spurzem, R., Baumgardt, H. & Ibold, N. [2003], 'A parallel implementation of an N-body integrator on general and special-purpose computers', *Mon. Not. R. Astron. Soc.* in press.

Tout, C.A., Aarseth, S.J., Pols, O. & Eggleton, P. [1997], 'Rapid binary star evolution for N-body simulations and population synthesis', *Mon. Not. R. Astron. Soc.* **291**, 732–48.

16 Appendix

In this Appendix we provide some tables of useful information. Table 1 defines the parameters used in the general `common` block, together with representative values for a test calculation with 1000 single particles and 1000 primordial binaries. Note the provision of extra KS solutions in case of additional close encounters in the early stages.

Table 1: *FORTRAN parameters.*

$N_{\rm max}$	*NMAX*	Total particle number and c.m. bodies	4010
$K_{\rm max}$	*KMAX*	KS solutions	1010
$L_{\rm max}$	*LMAX*	Neighbour limit	100
$M_{\rm max}$	*MMAX*	Hierarchical binaries	10
$M_{\rm dis}$	*MLD*	Recently disrupted KS components	22
$M_{\rm reg}$	*MLR*	Recently regularized KS components	22
$M_{\rm high}$	*MLV*	High-velocity particles	10
$M_{\rm cloud}$	*MCL*	Interstellar clouds	10
$N_{\rm chain}$	*NCMAX*	Chain membership	10

Table 2 contains an example of standard input parameters and typical values for an $N = 1000$ test run (a larger value of *NNBMAX* may be used for primordial binaries). Both the book and *FORTRAN* notations are given for convenience.

Table 2: *Integration parameters.*

η_I	*ETAI*	Time-step parameter for irregular force	0.02
η_R	*ETAR*	Time-step parameter for regular force	0.03
S_0	*RS0*	Initial radius of the neighbour sphere	0.3
$n_{\rm max}$	*NNBMAX*	Maximum neighbour number	70.0
$\Delta t_{\rm adj}$	*DTADJ*	Time interval for energy check	2.0
$\Delta t_{\rm out}$	*DELTAT*	Time interval for main output	10.0
$Q_{\rm E}$	*QE*	Tolerance for energy check	1.0×10^{-5}
$R_{\rm V}$	*RBAR*	Virial cluster radius in pc	2.0
$M_{\rm S}$	*ZMBAR*	Mean stellar mass in solar units	0.8
$Q_{\rm vir}$	*Q*	Virial theorem ratio $(T/\|U - 2W\|)$	0.5
$\Delta t_{\rm cl}$	*DTMIN*	Time-step criterion for close encounters	4.0×10^{-5}
$R_{\rm cl}$	*RMIN*	Distance criterion for KS regularization	0.001
η_U	*ETAU*	Regularized time-step parameter	0.2
$h_{\rm hard}$	*ECLOSE*	Energy per unit mass for hard binary	1.0
$\gamma_{\rm min}$	*GMIN*	Limit for unperturbed KS motion	1.0×10^{-6}
$\gamma_{\rm max}$	*GMAX*	Termination criterion for soft binaries	0.01

The main options are listed below. For a complete list see routine `define.f`. To find where option # J is used, type '`grep KZ(J) *.f`' in the N-body directories (or just 'KZ' in the directory `Chain`).

Table 3: *Optional features.*

1	Manual common save on unit 1 at any time
2	Common save on unit 2 at output time or restart
3	Data bank on unit 3 with specified frequency
5	Standard initial conditions (=0: uniform; =1: Plummer)
6	Output of significant & KS binaries (=1, 2, 3 & 4)
7	Output of Lagrangian radii (several types)
8	Primordial binaries (extra input required)
10	Regularization diagnostics
11	Primordial triples (extra input required)
12	HR diagnostics of evolving stars (interval DTPLOT)
13	Interstellar clouds (extra input required)
14	External tidal force; open or globular clusters
15	Multiple regularization or hierarchical systems
16	Updating of regularization parameters $R_{\rm cl}$, $\Delta t_{\rm cl}$
17	Modification of η_I and η_R by tolerance $Q_{\rm E}$
18	Neighbour additions (> 0: high-velocity; > 1: all types)
19	Synthetic stellar evolution with mass loss (>=3)
20	Different types of initial mass functions
21	Extra output line (MODEL #, CPU, DMIN, AMIN, RMAX)
22	Initial conditions $m_i, \mathbf{r}_i, \dot{\mathbf{r}}_i$ on unit #10 (=2)
23	Removal of distant escapers (isolated or tidal)
26	Slow-down of KS and/or chain regularization (=1 or 2)
27	Tidal circularization (=1: sequential)
28	Magnetic braking or gravitational radiation
30	Chain regularization (> 1: special diagnostics)
32	Increase of output interval (limited by $t_{\rm cr}$)
33	Distribution of block-steps at output (1 or 2)
35	Integration time offset (standard = 100 time units)
36	Step reduction for hierarchies (not recommended!)
37	Fast time-step criterion (not recommended!)
38	Force polynomial corrections (=0: I > N; not rec.)
40	Increase neighbour numbers if <NNB> < NNBMAX/2

Table 4 defines significant counters, together with an actual example from run with 1800 single particles and 200 primordial binaries.

Table 4: *Characteristic counters.*

Name	Definition	Counts
NSTEPI	Irregular time-steps	8.0×10^7
NSTEPR	Regular time-steps	2.1×10^7
NBLOCK	Block steps	5.1×10^6
NKSTRY	Regularization attempts	2.4×10^6
NKSREG	KS regularizations	2.2×10^3
NKSHYP	Hyperbolic regularizations	800
NKSMOD	KS slow-down modifications	6.6×10^4
NKSPER	Unperturbed two-body orbits	4.6×10^{11}
NMERGE	Hierarchical mergers	359
NEWHI	Independent new hierarchies	25
NCHAIN	Chain regularizations	86
NSTEPU	Regularized time-steps	4.2×10^7
NSTEPC	Chain integration steps	1.1×10^5
NMDOT	Stellar evolution look-ups	5.0×10^4
NSN	Supernova events	5
NWD	White dwarfs	90
NCOLL	Stellar collisions	4
NBS	Blue stragglers	2
NSYNC	Circularized binaries	9
NSESC	Single escapers	1833
NBESC	Binary escapers	176
NMESC	Hierarchical escapers	2

Table 5 lists the control indicators used for decision-making.

Table 5: *Indicator for flow control.*

0	Standard value
1	New KS regularization
2	KS termination
3	Output and energy check
4	Three-body regularization
5	Four-body regularization
6	New hierarchical system
7	Termination of hierarchy
8	Chain regularization
9	Physical collision
-1	Exceptional cases