Explanations to install, compile and run ME(SSY)**2

Marc Freitag

July 17, 2008

1 Preamble

This file and the codes it describes should be available at the url
`http://www.ast.cam.ac.uk/research/repository/freitag/MODEST_MonteCarlo/MESSY_Download.html`.

1.1 References

ME(SSY)**2 stands for "Monte-carlo Experiments with Spherically SYmmetric Stellar SYstems"[1]. This name also conveys something of the programming style I used... Note that ME(SSY)**2 is just the "commercial name" but, traditionally, since I started developing this code in 1995 or 1996, the executable is called `evolamas` (something like "evolve cluster" in French) and the main source file is `EvolAmas.F`.

The physical and algorithmic principles underlying ME(SSY)**2 are described in:

- Freitag, M. & Benz, W. 2001, *A New Monte Carlo Code for Star Cluster Simulations: I. Relaxation*, A&A, 375, 711
  `http://uk.arxiv.org/abs/astro-ph/0102139`
  `http://dx.doi.org/10.1051/0004-6361:20010706`
- Freitag, M. & Benz, W. 2002, *A New Monte Carlo Code for Star Cluster Simulations: II. Central Black Hole and Stellar Collisions*, A&A, 394, 345
  `http://uk.arxiv.org/abs/astro-ph/0204292`
  `http://dx.doi.org/10.1051/0004-6361:20021142`

The code has been used in the following refereed papers: Freitag (2001); Freitag & Benz (2001, 2002); Freitag (2003); Freitag et al. (2006c,b,a). The code is based on the scheme pioneered by Michel Hénon in the 70s to follow the evolution of globular clusters (Hénon 1971a,b; Hénon 1973; Hénon 1975). For some general information on Hénon-type Monte-Carlo (MC) codes, see my slides for the "Cambody" summer school: `http://www.cambody.org/index.php?page=notes` (the slides on Fokker-Planck methods are useful to understand the treatment of 2-body relaxation). As a follow-on of this school, a book on collisional stellar dynamics is "in the making". My chapters on MC and Fokker-Planck approaches can be found at `http://www.ast.cam.ac.uk/~freitag/Cambody/Book/share/`. I have also written a couple of web pages on MC techniques for the MODEST[2] web site: `http://www.ast.cam.ac.uk/~freitag/MODEST_MonteCarlo/intro.html`. There you will find more references but it is a bit outdated as I haven't changed anything there since early 2004.

1.2 License and use of Numerical Recipes routines

ME(SSY)**2 should be free and open. It is distributed under an MIT license (see appendix A). Unfortunately it relies on the use of some routines taken from "The numerical recipes in Fortran" which you can only use if you have bought them, I think. See `http://www.nr.com/` and `http://www.numerical-recipes.com/com/storefront.html`. I should try to find open and free alternatives to these routines. For the time being, they are in files whose names and with `_NR.f` or `_NR.F`, *which are not distributed with ME(SSY)**2*. The routines used are:

[1] Name suggested by Pau Amaro-Seoane.
[2] "MOdeling Dense STellar systems", see `http://www.manybody.org/modest/`.

`covsrt dawson dfridr erfcc gammln gaussj hunt indexx lfit locate midinf midpnt midsql odeint polint qromb qromo qsimp rkck rkqs rtsafe selip shell spline trapzd zriddr`

The have all been adapted to the use of `double precision` floating-point number (instead of `real`). Furthermore, ME(SSY)**2 also uses a modified verion of `rtsafe`, `rtsafe_rel` which has the following interface:

```
double precision function rtsafe_rel(funcd,x1,x2,xacc_rel)
```

`xacc_rel` is the required *relative* accuracy of the root determination. I.e. you need to use a test of the following sort:

```
if (abs(dx).lt.(xacc_rel*abs(rtsafe_rel)+1.0d-20)) then
```

at the appropriate place ;-)

1.3 Very quick overview

ME(SSY)**2 evolve star cluster models. The code is based on the following set of core assumptions:

- Spherical symmetry (hence no rotation, flattening, triaxiality, etc).
- Dynamical equilibrium (no violent relaxation, etc).
- Diffusive relaxation. The 2-body relaxation is treated in the Chandrasekhar approximation by assuming that it amounts to the effects of a large number of uncorrelated, small-angle 2-body encounters, each of which is a Keplerian hyperbolic deflection.

As of November 2006, the physics included is:

- Self-gravity of the cluster.
- 2-body diffusive relaxation.
- Simple stellar evolution. The stars do not evolve on the MS. At the end of their MS lifetime, they turn into compact remnants.
- Stellar collisions. One can inter- or extrapolate from a large database of SPH simulations (Freitag & Benz 2005).
- Large-angle 2-body encounters ("kicks"). Those are supposed to be negligible in most circumstances but it was easy to include into the code.
- A central massive object (generally thought to be a (I)MBH). It is considered fixed at the centre. It interacts with the stellar system through:
  - Contribution to potential.
  - Tidal disruptions.
  - Direct plunges through horizon.
  - "Extreme mass-ratio inspiral" due to emission of gravitational waves. This process might not be treated accurately enough in the present implementation.
- Tidal truncation of the cluster (condition on apocentre distance).

Regrettably, at this point ME(SSY)**2 **does not** include

- Giant stars (or any detailed stellar evolution).
- Binaries.
- Detailed treatment of tidal effects such as variable tidal field, delayed evaporation (Fukushige & Heggie 2000; Baumgardt 2001), disk and bulge shocking.
- Effects of gas (contribution to potential, accretion onto stars or the MBH, effects of gas removal from young clusters).

1.4 Units

In most places (in particular for input and output), ME(SSY)**2 uses so called $N-$body units (Hénon 1971b; Heggie & Mathieu 1986). I define the unit system such that the constant of gravity is $G = 1$, the total *stellar* mass is initially $M_{\rm cl}(0) = 1$, and the total initial *stellar* gravitational energy (not accounting for the contribution of the MBH to the potential) is $-1/2$ (Freitag & Benz 2001; Freitag et al. 2006a). As a time unit, I use the "Fokker-Planck time" $T_{\rm FP}$ which is connected to the $N-$body time unit $T_{\rm NB}$ through $T_{\rm FP} = (N_*(0)/\ln\Lambda)T_{\rm NB}$ were $N_*(0)$ is the initial number of stars and $\ln\Lambda = \ln(\gamma N_*(0))$ the Coulomb logarithm (Binney & Tremaine 1987; Spitzer 1987). I prefer to use $T_{\rm FP}$ rather than $T_{\rm NB}$ because the former is a relaxation time while the latter is a dynamical time. Individual stellar masses and radii (important for stellar evolution and collisions) are in $\rm M_\odot$ and $\rm R_\odot$, respectively.

2 Installing ME(SSY)**2

2.1 Use of `xdr` format

For historical reasons[3] (FHR), I used some portable binary format called `xdr` for the large snapshot files containing the particle data, in particular the initial condition file(s). One has to download and install the `fxdr` fortran `xdr` library, to be found at `http://meteora.ucsd.edu/~pierce/fxdr_home_page.html`. This is generally straightforward to install. My make files assume the library `libfxdr.a` is put in `$HOME/lib` and the include file `fxdr.inc` in `$HOME/include`.

I can provide various codes to convert to/from `xdr` files. In practice, you will need some code to write the initial condition file(s) in `xdr` format as ME(SSY)**2 is not able to generate its own initial cluster model (this is a feature `;-)`). You will need to be able to read the `xdr` snapshot only if you want to know about the properties of individual particles. ME(SSY)**2 also outputs a lot of information in `ASCII` files to follow overall or statistical properties of the cluster, such as Lagrange radii or interesting events such as collisions.

2.2 Compiling the code

The code is written in more or less standard fortran-77. It is spread over a large number of files (153 and counting). The main file is `EvolAmas.F` and most other files have a name starting with `EvolAmas_`. The extension `.f` is for plain fortran files while `.F` files contain pre-processor directive allowing to customise the code at compile time. Most pre-processor variables are set in the make file `EvolAmas.mke`. The file `Machine_Dependent.F` contains all subroutines that might cause a problem when porting from one architecture to another or from a compiler to another. Their name end with `_MD`. In general, files whose names end with `_inc.f` are include files, i.e., they are "incorporated" into other files through the "`include`" command. Names ending with `_param.f` and `_common.f` contain definitions of parameters (aka constants) and common blocks that are included in several places.

I highly recommend to use GNU tools. In particular the GNU make and the the GNU fortran compiler `g77` or the more recent `gfortran`. I was able to compile the code on Solaris, various GNU/Linuxes PCs and Mac OSX. Long time ago, I was successful using the Solaris fortran compiler, the Portland group compiler (on PCs), and the Intel compiler (on PCs). But I now stick to gnu compiler and `EvolAmas.mke` probably only works for `g77` (if at all!). Note that the make file might not work with non-GNU versions of make and that it needs the Z shell `zsh` to be installed (its path is set via the variable `SHELL` at the beginning). Another reason why you need `zsh` is that it is used by the scripts `mke2list_preproc_defines.sh` and `CreateFortranPrintCodeForListPreprocDefines.sh` called by the makefile. These scripts are called to write some fortran routine which will print out the values of most (all?) precompiler variables in `CaracCode.asc` when ME(SSY)**2 is run.

If you are lucky, compiling ME(SSY)**2 is as easy as

```
%make -f EvolAmas.mke
```

where `%` indicates the prompt. This should create the executable `__evolamas__` which you can happily rename to, say, `evolamas`. However many aspects of what the code will do, in particular the included physics, have

[3]The goal was to avoid "endianness" problems when transferring files from, say, Solaris workstations to PCs. But a more widely used and better supported format such as `FITS` or compressed `ASCII` would have been preferable.

to be decided before compilation as they are controlled by pre-processor variables. In Table 1, I list the pre-processor variables that are set in `EvolAmas.mke` through the `-D` switch of the compiler. For instance, the variable `_ECRIT_ETOILES_` is set to 1 in through `-D_ECRIT_ETOILES_=1`. Not all pre-processor variables are set in `EvolAmas.mke`. Some are (re)set in various `.F` files using the pre-processor "`#define`" directive. As a general rule, only the pre-processor variables with a leading "*" in Table 1, should be tinkered with. Note that many of them determine what kind of information is output. See § 3.2 and Table 5 for more information about ME(SSY)**2's rich output.

Before compiling, you should also set the parameter `iDimSE` in `Param_Amas.f` to a value large enough to accommodate your number of particles. `iDimSE` is the size of arrays containing the particles' data; it can be larger than the number of particles you will use (this might or might not result in more memory usage than required).

3 Running ME(SSY)**2

3.1 Input files and parameters

This is where the fun begins! Over the years, ME(SSY)**2 has grown quite complex but I always tried to preserve compatibility with the data and parameter files of the previous versions. As a results, the input and output data are scattered in more files (and using more formats) that is really comfortable! Table 2 shows a list of input files, i.e., files containing initial conditions (the structure of the cluster at $t = 0$) or parameters to set the physics or affecting the numerics. Note that only a very few are compulsory for ME(SSY)**2 to run. The others can be used to override the default physics and numerics.

In Table 3, I list files that can be used for some (limited) real-time control of the run, such as requesting a "snapshot" (saving all particle data to disk) or an early termination. Most importantly, there is a Perl script, `gere_evolamas.pl` which is driver for the fortran executable `evolamas`.

3.1.1 The main parameter file (aka `input_EvolAmas`)

This is the most important file to set nearly all physical and numerical parameters. If one uses `gere_evolamas.pl` to drive the run, the script will look for a file named `input_EvolAmas` containing parameters and comments, and format it for `evolamas`. In this case one can include blank lines and anything starting with `#` is considered a comment. If `gere_evolamas.pl` is not used, the format is more strict (see below) and the preferred way to pass the parameters is by using the command:

```
%./evolamas -f my_parameter_file
```

where `my_parameter_file` is your parameter file (no kidding!). For simplicity, we will assume it is called `input_EvolAmas`. The stricter format for `input_EvolAmas` accepted by `evolamas` is something like that:

```
Param_name_1 # Some comments if you like
   Param_value_1
Param_name_2
   Param_value_2  # More comments if you please
Param_name_3
   Param_value_3
```

This will give the value `Param_value_1` to the variable `Param_name_1` and so on. Note that one or more white space must appear before the value of a parameter and none before its name. In principle `#` is not required before a comment as only the first "word" of each line will be read anyway! `evolamas` will complain and stop if it doesn't recognise the name of a parameter. If you are using `gere_evolamas.pl`, you can change the value of parameters on the command line:

```
%./gere_evolamas.pl --option Param_name_1=Param_value_1 --option Param_name_2=Param_value_2
```

If you use the switch `--Print` with `gere_evolamas.pl`, it will just output on stdout a cleaned-up version of the parameter file but not run `evolamas`. This is useful for systems where you cannot drive the code with such a script (such as a PC cluster using some queue submission software) but you still want `gere_evolamas.pl` to prepare a nice parameter file for you.

It should be noted that the code will try to start with the snapshot files (`MyRun%xxxxxxxxxx%AMAS.xdr`, etc) corresponding to the step number given by the parameter `i_ini` (value of "`xxxxxxxxxx`")[4]. Hence, if you want to (re)start from $t = 0$ set `i_ini` to 0. If you want to continue a simulation interrupted at some step n (with existing corresponding snapshot files), set it to n. `gere_evolamas.pl` will not allow you to run a simulation in a directory where there are already snapshot files for a step number larger than 0 unless you use the switch `--Force` to force restart from scratch or `--Continue` to continue from the last snapshot in which case you don't have to set `i_ini` manually.

Note that it is important that the number of digits in "`xxxxxxxxxx`" be exactly 10 (ten, dix, sepuluh). The code is too dumb to find the input file(s) if they do not follow this convention!

3.1.2 Initial condition files

The file `MyRun%0000000000%AMAS.xdr` is one of the very few which is compulsory it contains the initial cluster structure, particle by particle. This data is the mass M, radius R, (specific) kinetic energy E and modulus of (specific) angular momentum J for each particle. These data are stored in the arrays `M_SE`, `R_SE`, `T_SE` and `J_SE` in a common bloc (defined in `VarSE.f`)[5]. $N-$body units are to be used for all these data.

In most cases, you will also need a file `MyRun%0000000000%ETOILES.xdr` containing the basic stellar properties of the stars of each particle, its mass M_* (in $M_\odot$), type and "date of birth" (in yr, since $t = 0$). The mass information is redundant with that in `MyRun%0000000000%AMAS.xdr` since $M_* \propto M$ but when more detailled stellar evolution will be included, M_* will probably refer for the "zero-age" mass while M will track mass loss. The date of birth is generally when the star acquirred its present type but may be modified in collisions to account for "rejuvanation" (mixing in MS stars). It is used to determine the current (effective) evolutionary age. There are currently for stellar types: 1, 3, 4, 5 for MS, WD, NS and BH; no giant phase!. These data are stored in the arrays `Met_SE`, (stellar mass) `DNet_SE`, (birth date) `iTet_SE` (stellar type) in a common bloc (defined in `VarSE.f`). The numerical type of `iTet_SE` is `byte`. All other arrays are `real*8` (aka `double precision`).

Typical usable number of particles range from 10^5 to 10^7. $10^3 - 10^5$ can be used for test runs. It is important to insist on the fact that the number of stars is completely independent of the number of particles (unlike in other current Hénon-type MC codes). The number of stars is given by the parameter `rNbEtoiles` in `input_EvolAmas`.

The easiest way to get initial condition files is to ask me! The next easiest way is to use some of my programs to generate initial conditions:

- `Creer_Amas_FdeE.F` can create `AMAS.xdr` according to a variety of theoeritcal models: Plummer, King, Wooley & Dickens (not tested), Isochrone, Dehnen with $\gamma = 0, 1, 2$, and stellar polytropes (Binney & Tremaine 1987; Dehnen 1993). No central object can be included. Which model will be generated (as well as what type of parameters have to be entered) depend on the value of the pre-processor variable `_TYPE_` (see source file). When compiled for Plummer models, and called "`create_plummer`", the command-line syntaxe is

  ```
  %create_plummer -N 1000000 -Name MyPlummer [-iseed 763367]
  ```

  to create a file `MyPlummer.xdr` containing one million particles. The option `-iseed` can be used to impose a different seed for the random number generator. When compiled for King models, and called "`create_king`", the command-line syntaxe is

  ```
  %create_king -N 1000000 -Name MyKing -W 8 [-iseed 763367]
  ```

  were `-W` is used to give the value of the dimensionless central potential W_0. In this case a tiny file `MyPlummer_MAREE.xdr` is also created to indicate the value of the tidal radius.

- `Creer_Amas_EtaTN.F` can create "eta-models" (Dehnen 1993; Tremaine et al. 1994) with or without a central mass. This includes Hernquist and Jaffe models. The syntaxe is

[4] One step correspond to the modification of just one pair of particles. Therefore the step number can reach billions.

[5] `SE` comes from "Super-Étoile" the French for "super star" as it is how Hénon used to call particles to stress the fact that they may represent more than one star each.

```
%create_eta -N 1000000 -Name MyEta -eta 1.5 -mu 0.03 [-iseed 763367]
```

where `-eta` if for the parameter η (the density at small radii is $n \propto R^{\eta-3}$) and `-mu` is for the parameter μ, the mass of the central object (in units of the total stellar mass, so the total mass of the system is $1+\mu$). If $\mu > 0$, a file `MyPlummer_TN.xdr` is created to contain the value of the mass of the central object.

These models will have no mass spectrum (and no `ETOILES.xdr` file). To introduce a mass spectrum and create the corresponding `ETOILES.xdr` file, one can use `AppliquerFM3.F` (sorry for the name!) which allow to apply any initial mass function that can be represented as a piecewise power-law (see Appendix A2 of Freitag & Benz 2002). Explanations about usage are given in the source code. For a Kroupa IMF, you would do

```
%appliquerfm3 -m "0.01,0.08,0.5,150" -a "0.3,1.3,2.3" MyModel MyModel_WithMF
```

so `-m` gives the list of limit-masses, `-a` the list of exponents (2.35 is Salpeter), `MyModel` the base name of the input model without mass function and `MyModel_WithMF` the base name of the output model. At the minimum files `MyModel_WithMF.xdr` and `MyModel_WithMF_ETOILES.xdr` will be created. The radii might be rescaled a bit to enforce strict virialisation and keep with the definitioin of $N-$body units, therefore a file `MyModel_WithMF_MAREE.xdr` can be created to contain the value of the rescaled tidal truncation radius.

To start a run, create a new directory. Inside copy your initial condition files with the following change of names:

`MyModel.xdr` → `MyRun%0000000000%AMAS.xdr`

and

`MyModel_ETOILES.xdr` → `MyRun%0000000000%ETOILES.xdr`,

`MyModel_MAREE.xdr` → `MyRun%0000000000%MAREE.xdr`, etc

if it applies. Note that you need exactely 11 (eleven) "0" in the filenames. If you use `gere_evolamas.pl`, you also need to put the executable `evolamas` and `input_EvolAmas` into this directory.

3.2 Output files

ME(SSY)**2 can produce a large variety of output files. They are presented in Table 5. Many of them are for specialised application or testing purpose and only a few are of general use. FHR, the formats are `xdr`, to preserve the full numerical precision of some data which might be crucial for restart[6] and `ASCII` for readability for data which are most useful for monitoring the run and plotting results. Most files are incremental, with information appended to the file as the simulation proceeds. The important exception are "snapshot files" that contain (in principle) the whole information about all particles and can be used for restart or detailed analysis. In this case a different set of files is written each time a snapshot is dumped, with file name indicating the step number, for instance `MyRun%00001200000%AMAS.xdr` for the mass and orbit information of all particles at step 1200000.

The most important output file is the "master log file" `MyRun%%Log.asc`. It contains a wealth of information about the evolution of the simulation, from a numerical and physical point of view. The same information is also sent to stderr to entertain the user. The format is supposed more human- than computer-friendly and a script `Log2rdb.pl` can be used to turn this file into a flat table format[7] The information present in `MyRun%%Log.asc` is presented in Table 6. For the time, I recommend to use the variable `Tps_Ufokpl` which give it in Fokker-Planck units. In most other output file, no time information is given, only the step number (`iPas_Evol` or similar name) or, if a time is given, it might not be in the proper units! Therefore it is safer to use `MyRun%%Log.asc` to derive the `iPas_Evol` → `Tps_Ufokpl` relation and used it determine time from step number.

Most other `ASCII` files have a rather straightforward flat format. The contain data in columns and a header to explain what the data is. The header are the first lines, starting with `#`. The last of these lines generally looks like

[6] But again, saving 8-byte real data in `ASCII` with 15 decimal positions also preserves the full precision. Once compressed with, e.g., `gzip`, such files are nearly as compact as their `xdr` counterpart. This is the way of the future!

[7] FHR the output format is "
`rdb`" with a two line header and tabs used to separate columns. It can be turned into the usual `ASCII` format I use for output from fortran code with another script: `rdb2fort.sh`.

```
# 1: iPas_Evol 2: this_var 3: that_var 4: this_other_quantity
```

which means that the first column contains (the successive values of) `iPas_Evol`, the second contains `this_var`, etc. `MyRun%%RayLag.asc` is an important output file whose header has a slightly different structure. The last header line looks like

```
# 1: iPas_Evol 2: NbSE_liees 3: M_liee 4: Tps_amas || FracMasse :  5: R0001 (0.001000) \
  6: R0010 (0.010000)  7: R0020 (0.020000)
```

which indicates that the 5th column contains the radius of the sphere containing 0.1% of the total mass, the 6th column the radius for 1% and so on. These quantities are known as "Lagrange radii" and a sphere containing a given fraction of the (remaining) cluster mass is a "Lagrange sphere". Similarly, the file `MyRun%%Segr.asc` ahs the following header

```
# 1: iPas_Evol 2: Tps_amas || Masse stellaire moyenne :  3 : m00050 (M<=0.0050)   \
  4 : m00100 (M<=0.0100)   5 : m00500 (M<=0.0500)
```

indicating that the third column contain the average stellar mass (in $M_\odot$) for all particles inside the 0.5% Lagrange sphere, etc.

If the code detects a error situation it will try and write a lot of information (such as a last snapshot) in files with name starting with `_RIP_` (e.g., `_RIP_%00051278532%AMAS.xdr`) before it exits to help post-mortem investigations into the origin of the problem.

Table 1: Pre-processor variables in `EvolAmas.mke`. A leading * indicates a particularly important varaiable.

	Variable	Recommanded value	What it does
	_SURV_CROISS_TEMPS_	0	If >0, check that cluster time is actually increasing (!)
	_SURVEILLE_SQRT_	0	If >0, check if argument of sqrt is ≤ 0 (debugging)
	_TEST_MODIF_PROP_SE_ACT_	0	?
	_NO_DF_	0	If >0, do not try to use a "`call system('df')`" to test for free disk space
	_NO_FLUSH_	0	If >0, do not use a "`call flush`" to force write into file
*	_ECRIT_ETOILES_	1	If >0, write stellar data for each snapshot (in `*%ETOILES.xdr` files)
	_ECRIT_CB3C_	0	If >0, write data about 3-body binary heating (does not work)
*	_ECRIT_RAYLAG_	1	If >0, write Lagrange radii data (in `MyRun%%RayLag.asc`)
*	_ECRIT_ANILAG_	1	If >0, write anisotropy data (in `MyRun%%AniLag.asc`)
	_ECRIT_SEGR_	1	If >0, write average mass data (in `MyRun%%Segr.asc`)
	_ECRIT_RELAX_	0	If >0, write global relaxation data (in `MyRun%%GlobRelax.asc`)
*	_ECRIT_COLL_	1	If >0, write data about each collision (in `MyRun%%Coll.asc`)
	_ECRIT_DUPLIC_	0	If >0, write data about particle duplication (in `MyRun%%Duplic.asc`)
*	_ECRIT_EVAP_	1	If >0, write data about each evaporation (in `MyRun%%Evap.asc`)
	_ECRIT_LC_	1	If >0, write data about each "loss cone" event (in `MyRun%%LC.asc`)
	_ECRIT_CAPT_GW_	1	If >0, write data about each "extreme mass-ratio inspiral" event (in `MyRun%%GW.asc`)
*	_ECRIT_SUIVI_TYPES_STELL_	1	If >0, Follow distribution of the various stellar types (in `MyRun%%SuiviMS.asc`, etc)
	_ECRIT_SUIVI_SE_PART_	0	If >0, follow very closely some particles (in `MyRun%%SuiviSE.asc`
	_ECRIT_ARBRE_	0	If >0, write data about binary tree (in `*%ARBRE.xdr` files)
	_ECRIT_PROFILPOT_	0	If >0, write grids used by code to estimate ρ and σ (in `*%PG.xdr` files)
	_ECRIT_TH_	0	If >0, write parameters for the selection of particle pairs (in `*%TH.bin` files)
	_ECRIT_RAND_	0	If >0, write internal variables of random generator (in `*%RAND.asc` files)
	_ECRIT_GRILLE_	?	If >0, write grids to follow radial structure evolution (in `*%GRILLE.xdr` files)
	_ECRIT_PG_	0	If >0, write grid used by the code to compute density and velocity dispersion (in `*%PG.xdr` files)
	_ECRIT_TSCALE_	1	If >0, write some info about timescales (in `MyRun%%Tscale.asc`)
	_ECRIT_SPECMASSE_	?	If >0, write some info about mass spectrum (in `MyRun%%MSpec.asc`)
	_ECRIT_STEVOL_	?	If >0, write info about stellar evolution events (in `MyRun%%StEvol.asc`)
	_ECRIT_NATKICKS_	?	If >0, write info about NS/BH natal kicks (in `MyRun%%NatKicks.asc`)
	_ECRIT_LAGQUANT_	?	If >0, write lots of Lagrangian quantities (in `MyRun%%LagQuant.asc`)

continued on next page

continued from previous page

	Variable	Recommanded value	What it does
	_MONITOR_MBINS_	?	If >0, track evolution of particles in various mass bins (in `MyRun%%MbinRad.asc`)
	_MONITOR_STRONG_ENCOUNTERS_	0	If >0, keep track of "super encounters" (to simulate 2-body relaxation) with large (cumulative) deflection angles (in `MyRun%%StrongEncounters.asc`))
	_WRITE_COLL_EXCEPTIONS_	1	If >0, write info about collisions requiring special treatment (in `MyRun%%CollExcept.asc`)
	_FOLLOW_SUBPOP_	0	If >0, follow closely subpopulation of particles (in `MyRun%%SubPop.asc`
*	_PRESENCE_TN_	?	If >0, include central object (MBH generally)
*	_TRONC_MAREE_	?	If >0, include tidal truncation
	_LIMITE_EXT_	0	If >0, include reflecting wall (broken!)
	_EMPECHE_EVAP_RELAX_	1	If >0, do not allow a star to escape following a relaxation "super-encounter"
	_DEPL_ORBITAL_	1	If $=0$, do not move stars on their orbits
*	_RELAXATION_	1	If >0, include 2-body relaxation
*	_KICKS_	0	If >0, include large-angle scatterings
*	_COLLISIONS_	?	If >0, include collisions
*	_DECHIREMENTS_	1	If >0, include tidal disruptions by MBH
*	_DISPARITIONS_	1	If >0, include direct plunges into MBH
*	_CAPT_GW_	0	If >0, include inspiral into MBH by GW emission
*	_EVOL_STELL_	?	If >0, include stellar evolution
*	_NATKICKS_	1	If >0, include natal kicks for NSs and stellar BHs
	_CHAUFF_BIN_3C_	0	If >0, include heating by 3-body binaries (does not work!)
	_COLLISION_CHECK_ORBIT_OVERLAP_	0	If >0, do not allow collisions if orbits do not overlap in R
*	_TYPE_COLL_	10	Determine collision treatment. 1 for simplistic model (pure meger or total disruption); 2 for Davies' formulas in (Rauch 1999); 3 for point-mass gravitational scattering; 4 for toy-model parametrisation; 9 non-physical, for tests; 10 for SPH-inspired presciptions (Freitag & Benz 2005)
	_TRAITEMENT_COLL_MS_RMN_	2	Determine treatment of MS-compact collisions. 0 does nothing (collisions still counted, stars and orbits not modified); 1 for complete disruption of MS star; 2 for tidal disruption of MS star and partial accretion onto compact object
*	_COLL_REJUV_	2	Determine treatment of collisional rejuvenation of MS stars. 0 to keep absolute age of more massive star; 1 resets age to 0 in case of merger; 2 use central He mass to determine effective age
*	_HARDEN_SPH_COLL_TREATMENT_	3	Determine if and how the SPH-inspired treatment of collisions should be "hardened" to avoid problems (see Freitag et al. 2006c).
	_CONST_R_TID_DISRUP_	0	If >0, assume constant tidal disruption radius for all stars
*	_TYPE_TREL_GW_	0?	Determine how to estimate relaxation time to test for GW-inspiral into MBH. 0 uses (noisy) "encounter relaxation time"; 1 uses some orbit average; 2 uses fixed value
	_TYPE_FOR_MR_RELATION_	0	If >0, use this as the stellar type to determine M–R relation (1:MS, 3:WD, 4:NS, 5:BH)
	_FIGER_RMAR_	0	If >0, keep tidal truncation radius at initial value
	_AJUST_PASSIF_	0	If >0, a fraction of steps will be "passive adjustment" ones where particles are just moved on their orbits

continued on next page

continued from previous page

	Variable	Recommanded value	What it does
	_SURV_ADIAB_	0	If >0, make sure that time steps are shorter that a fraction of the local timescale for potential change
	_TYPE_LOG_COUL_	0	Type of Coulomb logarithm used. 0 for $\Lambda = \gamma N_*$; 1 for local value $\Lambda \propto \sigma^3 P_{\rm orb}(G\langle m_*\rangle)^{-1}$; 2 for $\Lambda = \gamma N_*(R) + M_{\rm BH}/\langle m_*\rangle$
	_EVOL_UNE_SE_	0	If >0, only one particle is modified in a pair (experimental! Energy not conserved!)
	STATIQUE	0	If >0, do not actually evolve anything (undo everything at end of each step; for debugging)
	_RELAX_NULLE_	0	If >0, undo relaxation at end of step (for debugging)
	_COLL_NULLES_	0	If >0, undo collision at end of step (for debugging)
	_DECHIR_NULS_	0	If >0, undo tidal disruption at end of step (for debugging)
	_FORCER_CROISSANCE_TN_	0	If >0, use adhoc prescription for growth of central object
	DUPLICATION	0	If >0, duplicate each particle when the particle number has become less or equal to half the initial number
*	_VERBOSITUDE_	5	Determine how verbose the code will be (on stdout)
	_VERB_PERTES_	0	Determine how verbose the code will be when a particle is "lost" to any process
	_VERB_STELLEVOL_	1	Determine how verbose the code will be about stellar evolution
	_VERB_EVAP_	0	Determine how verbose the code will be about evaporation
	_VERB_COLL_	0	Determine how verbose the code will be about collisions
	_VERB_INTERP_RESCOLL_	1	Determine how verbose the code will be about interpolation of SPH collisions results
	_SYM_M1_M2_	1	If >0, enforce mass-symmetry in the outcome of MS–MS collisions
	_RAY_LAG_SPECIAL_	0	If >0, use special choice for the default Lagrange fraction (obsolete)

File name	Compulsory?	Format	Role
input_EvolAmas	y	ASCII	Contains most numerical and physical parameters for the run
MyRun%00000000000%AMAS.xdr[1]	y	xdr	Initial cluster structure (masses, velocity, positions of particles)
MyRun%00000000000%ETOILES.xdr	n	xdr	Initial cluster structure (stellar masses, types [MS,WD,NS,BH] and ages)
MyRun%00000000000%TN.xdr	n	xdr	Initial masses of the "central black hole" and "central gas reservoir"
MyRun%00000000000%MAREE.xdr	n	xdr	Initial value of tidal truncation radius
Mbins.asc	n	ASCII	Specification of the mass bins to be tracked (Lagrange radii, etc.)
FracRayLag.asc	n	ASCII	Specification of the fractional mass for the overall Lagrange radii
FracSegrLag.asc	n	ASCII	Specification of the fractional mass for the Lagrange spheres to track the average stellar mass
FracSTSLag.asc	n	ASCII	Specification of the fractional mass for the Lagrange spheres used to track the various species
ParamStellEvol.asc	n	ASCII	Parameters for stellar evolution (Max ZAMS mass to form WD, NS, etc.)
MassRadiusMS.asc	n	ASCII	Main-Sequence Mass-Radius relation
Param_Croiss_Forc_TN.asc	n	ASCII	Parameters determining "forced" growth of central object
Liste_SE_a_suivre.asc	n	ASCII	List of IDs of particles to follow very closely
ListSubPop.asc	n	ASCII	List of IDs of particles to follow closely ("subpopulation")
Grille_ResColl_SPH.asc	y if collisions	ASCII	Grid of outcome of collsions of MS stars (from SPH simulations)

Table 2: Input files. [1] "MyRun" is the name of the run given in input_EvolAmas.

File name	Role
gere_evolamas.pl	Perl script to "drive" simulation. Makes it easier to pass parameters and to request special snapshots
_EvolAmas%%DemSauv.asc	Can be used to request snapshot save when some conditions are met if simulation is driven by gere_evolamas.pl
_LIRE_COND_SAUV_	Flag file. Forces gere_evolamas.pl to re-read conditions from _EvolAmas%%DemSauv.asc (erased once done)
_SAUV_DEMANDEE_	Flag file. Forces ME(SSY)**2 to save a snapshot as soon as possible (erased once done)
STOP	Flag file. Forces (clean) stop (erased at [re]start)

Table 3: Control files.

Table 4: Parameters in `input_EvolAmas`. A leading * indicates a particularly important parameter. Other parameters can generally be left at their default value. In the "Type" column, "r" stands for real, "c" for character and "i" for integer.

	Parameter	Type	Default	Explanation
	`NomSimul`	c*80	`'_EvolAmas'`	Run name. Used to form the name of input/output files
	`iVerbeux`	i	`1`	Determines how verbose the code will be (on the standard error)
*	`FacNbPasSauv`	r*8	`5.0d0`	A complete save (snapshot and all) is done every `FacNbPasSauv*NbSE` steps (`NbSE` is the number of particles)
*	`FacNbPasSauvPart`	r*8	`1.0d0`	Same as `FacNbPasSauv` for partial save
*	`FacNbPasInfo`	r*8	`1.0d0`	General info about run will be given every `FacNbPasInfo*NbSE` steps
	`FacNbPasRecArbre`	r*8	`2.0d0`	Binary tree rebuilt from scratch every `FacNbPasRecArbre*NbSE` steps
	`FacNbPasRecRelax`	r*8	`10.0d0`	Relaxation parameters recomputed every `FacNbPasRecRelax*NbSE` steps
	`FacNbPasRecTpsAmas`	r*8	`1.0d0`	Cluster (median) time recomputed every `FacNbPasRecTpsAmas*NbSE` steps
	`FacNbPasDetTrelExtremes`	r*8	`0.5d0`	Extreme values of relax times recomputed every `FacNbPasDetTrelExtremes*NbSE` steps
	`FacNbPasDetEvap`	r*8	`1.0d0`	Check for evaporated stars performed every `FacNbPasDetEvap*NbSE` steps
	`FacNbPasTestDuplic`	r*8	`0.5d0`	Test for particle duplication performed every `FacNbPasTestDuplic*NbSE` steps
	`FacNbPasTestDemSauv`	r*8	`0.1d0`	Test for file asking for snapshot (`_SAUV_DEMANDEE_`) performed every `FacNbPasTestDemSauv*NbSE` steps
	`FacNbPasRecalcTH`	r*8	`0.5d0`	The parameters for the function used to select particle pairs are recomputed every `FacNbPasRecalcTH*NbSE` steps
*	`i_ini`	i*8	`0`	Initial step value (Code will attempt to read corresponding snapshot)
*	`i_fin`	i*8	`2000000000`	Code terminates when step counter reaches this value
	`T_fin`	r*8	`1.0d5`	Code termintes when time (in some units!?) reaches this value
*	`Frac_Trelax`	r*8	`0.01d0`	Requested maximum value for time_step/relaxation_time
*	`Frac_Tcoll`	r*8	`0.01d0`	Requested maximum value for time_step/collision_time
	`Frac_Tkick`	r*8	`0.01d0`	Requested maximum value for time_step/large_angle_scattering_time
	`Frac_Tadiab`	r*8	`0.01d0`	Requested maximum value for time_step/potential_change_time
	`Frac_Taccr`	r*8	`0.01d0`	Requested maximum value for time_step/central_object_growth_time
	`Frac_Tevap`	r*8	`0.01d0`	Requested maximum value for time_step/evaporation_time
*	`Frac_Tevst`	r*8	`0.025d0`	Requested maximum value for time_step/stellar_evolution_time
	`Tadiab_ini`	r*8	`0.001d0`	Initial value of time scale for potential evolutioin (just needs to be short)
*	`FactTrelMax`	r*8	`1.0d3`	Maximum ratio of time steps (misnomer!)
*	`NbSECouchePot`	i	`25`	Requested number of particles per cell in density radial grid (misnomer!)
	`FacMinPG`	r*8	`0.5d0`	Grid rebuilt if number of particle in any cell drops below `FacMinPG*NbSECouchePot`
	`FacMaxPG`	r*8	`1.5d0`	Grid rebuilt if number of particle in any cell raises above `FacMaxPG*NbSECouchePot`
*	`MasseEtoileDef`	r*8	`1.0d0`	Default value for average stellar mass (overriden by info in `ETOILES.xdr` file if present)
	`TypeEtoileDef`	i	`1`	Default stellar type (overriden by info in `ETOILES.xdr` file if present; 1 for MS stars)
*	`rNbEtoiles`	r*8	`1.0d6`	Total number of stars (independent of `NbSE`!)

continued on next page

continued from previous page

	Parameter	Type	Default	Explanation
	iRand_Seed	i	133111	Seed for random number generator
	Frac_dt_max	r*8	1.0d0	Obsolete, was used to determine value of sub-timestep
*	TailleAmas_en_pc	r*8	3.0d0	N-body length units in parsecs
	NbSECoucheCB3c	i	1000	Not used. Number of grid elements used to determine 3-body binary heating. Never worked well.
	FacNbPasRecCB3c	r*8	0.5d0	Not used. 3-body binary heating recomputed every FacNbPasRecCB3c*NbSE steps
*	M_TN_ini_def	r*8	0.0d0	Default initial mass of central object, in fraction of total stellar mass (overriden by info in TN.xdr file if present)
*	NbSauv_Conserve	i	5	Only 1 snapshot every NbSauv_Conserve is kept. Others are erased as the simulation proceeds to save disk space
*	R_Mar_ini_def	r*8	1.0d30	Default initial value of tidal truncation radius in N-body units (overriden by info in MAREE.xdr file if present)
*	Gamma_relax	r*8	0.14d0	Proportionalite coefficient in Coulomb log. Lambda=Gamma_relax*N_star
	frac_accr_Coll	r*8	1.0d0	Fraction of mass liberated in collisions accreted by MBH
	frac_accr_Dechir	r*8	1.0d0	Fraction of mass liberated in tidal disruptions accreted by MBH
	frac_accr_EvolSt	r*8	1.0d0	Fraction of mass liberated by stellar evolution accreted by MBH
	effic_conv_lum	r*8	0.0d0	Efficiency for converting mass into light during accretion onto MBH
	Fac_dt_accr_M	r*8	0.01d0	The amount of mass in the central reservoir (disk) that can be accreted onto the MBH is determined every Fac_dt_accr_M*NbSE steps
	Fac_dt_accr_Npart	r*8	10.0d0	The amount of mass in the central reservoir (disk) that can be accreted onto the MBH is determined when the reservoir has accumulated more than Fac_dt_accr_Npart times the mass of an average particle
	Mu_mol_elec	r*8	1.13d0	Average molecular mass per electron of stellar gas
	fact_b0_kick	r*8	2.0d0	Encounters with impact parameter smaller than fact_b0_kick times the value for 90 degree deflection are treated as large-angle scatterings
	FracPas_AjustPassif	r*8	0.0d0	Fraction of passive steps (change of orbital position only)
	Rang_CC	i	10	Number of particles subject to special "central control". Probably obsolete
	FactR_CC	r*8	0.5d0	Factor for distance to centre used in "central control". Probably obsolete
	DeltaRang_Paire	i	1	Rank difference for two particles in an interacting pair
	StellMet_def	r*8	0.02d0	Default stellar metallicity (0.02 is solar)
	Fac_SldAvg_Tscale	r*8	5.0d0	NbSECouchePot*Fac_SldAvg_Tscale is the number of particles for sliding average to compute time scales in Calc_Tcarac
	Type_CtrObj_def	c*16	'TN'	Default type of central object (TN or MS)
	CollMassLossFracDef	r*8	0.0d0	Flat fractional mass loss in simplistic collisional treatment
	CollDminDestr_MS_vs_RMN	r*8	0.5d0	Distance below which the MS star is completely destroyed in encounters with compact stars (for _TRAITEMENT_COLL_MS_RMN_=1)

continued on next page

continued from previous page

Parameter	Type	Default	Explanation
CollAccrFrac_MS_on_WD	r*8	0.0d0	Fraction of mass of the MS star accreted onto WD if MS star tidally disrupted (for _TRAITEMENT_COLL_MS_RMN_=2)
CollAccrFrac_MS_on_NS	r*8	0.0d0	Fraction of mass of the MS star accreted onto NS if MS star tidally disrupted (for _TRAITEMENT_COLL_MS_RMN_=2)
CollAccrFrac_MS_on_BH	r*8	0.5d0	Fraction of mass of the MS star accreted onto BH if MS star tidally disrupted (for _TRAITEMENT_COLL_MS_RMN_=2)
Coef_Trlx_GW_Capt	r*8	1.0d0	Fudge factor on trelax when compared to GW-inspiral time for GW-capture
Tmax_GW_Capt_Gyr	r*8	1.0d30	Maximum inspiral time considered for GW captures, in Gyr
Trlx_GW_def_Gyr	r*8	1.0d0	Relaxation time when a constant value is used to look for captures (_TYPE_TREL_GW_=2; not recommended)
R_tid_disr_def_NB	r*8	1.0d-7	Tidal disruption radius all stars in N-body units when _CONST_R_TID_DISRUP_>0

Table 5: Output files. `xxxxxxxxxx`" is the number of the step. The + sign(s) indicate particularly useful output. A − sign indicate a file you will probably never need.

	File name	Incr	Format	Role
++	`MyRun%xxxxxxxxxx%AMAS.xdr`	n	`xdr`	Cluster structure (masses, velocity, positions of particles)
++	`MyRun%xxxxxxxxxx%ETOILES.xdr`	n	`xdr`	Cluster structure (stellar masses, types [MS,WD,NS,BH] and ages)
	`MyRun%xxxxxxxxxx%TN.xdr`	n	`xdr`	Masses of the "central black hole" and "central gas reservoir"
	`MyRun%xxxxxxxxxx%MAREE.xdr`	n	`xdr`	Value of tidal truncation radius
	`MyRun%xxxxxxxxxx%TEMPS.xdr`	n	`xdr`	Individual times of particles
	`MyRun%xxxxxxxxxx%CONS.xdr`	n	`xdr`	Data to allow accurate tracking of energy conservation in case of restart
	`MyRun%xxxxxxxxxx%GRILLE.xdr`	n	`xdr`	Radial grids to plot profiles (ρ, σ, β, etc)
−	`MyRun%xxxxxxxxxx%PG.xdr`	n	`xdr`	Radial grids used by code to estimate ρ and σ
−−	`MyRun%xxxxxxxxxx%ARBRE.xdr`	n	`xdr`	Binary tree used by code to store rank and potential info
−−	`MyRun%xxxxxxxxxx%TH.bin`	n	`bin`	Parameters for the selection of particle pairs (for debugging)
−	`MyRun%xxxxxxxxxx%RAND.asc`	n	`ASCII`	Internal variables of random generator (for restart continuing with same random sequence)
++	`MyRun%%Log.asc`	y	`ASCII`	Main log-file containing most of the useful information (also written to stdout)
++	`MyRun%%RayLag.asc`	y	`ASCII`	Lagrange radii evolution
+	`MyRun%%Segr.asc`	y	`ASCII`	Evolution of average stellar mass in Lagrange spheres
+	`MyRun%%AniLag.asc`	y	`ASCII`	Evolution of anisotropy (averaged in Lagrange spheres)
++?	`MyRun%%LagQuant.asc`	y	`ASCII`	Lagrangian quantities evolution (can replace the previous 3 files)
	`MyRun%%MSpec.asc`	y	`ASCII`	Some info to track evolution of the mass spectrum
+	`MyRun%%MbinRad.asc`	y	`ASCII`	Lagrange radii of particles in various bins of stellar mass
+	`MyRun%%SuiviMS.asc`	y	`ASCII`	Lagrange radii, and various quantities in Lagrange spheres for MS stars
+	`MyRun%%SuiviWD.asc`	y	`ASCII`	Same for white dwarfs
+	`MyRun%%SuiviNS.asc`	y	`ASCII`	Same for neutron stars
+	`MyRun%%SuiviBH.asc`	y	`ASCII`	Same for stellar black holes
	`MyRun%%Tscale.asc`	y	`ASCII`	Some (statistical) info about various time scales
	`MyRun%%CaptGW.asc`	y	`ASCII`	Info about "extreme mass-ratio inspirals" into central MBH (aka "GW captures")
	`MyRun%%Coll.asc`	y	`ASCII`	Info about collisions
	`MyRun%%CollExcept.asc`	y	`ASCII`	Info about collisions that required special treatment (extrapolation)
	`MyRun%%Evap.asc`	y	`ASCII`	Info about stars lost from the cluster through "evaporation"
	`MyRun%%LC.asc`	y	`ASCII`	Info about "loss cone" events (tidal disruptions, inspirals, etc)
	`MyRun%%NatKicks.asc`	y	`ASCII`	Info about NS and BH natal kicks
	`MyRun%%StEvol.asc`	y	`ASCII`	Info about stellar evolution (star/particle turning into remnant)
	`MyRun%%StrongEncounters.asc`	y	`ASCII`	Info about "super encounters" (to simulate 2-body relaxation) with large (cumulative) deflection angles
	`MyRun%%SuiviSE.asc`	y	`ASCII`	Info about some particles followed very closely
	`MyRun%%SubPop.asc`	y	`ASCII`	Info about subpopulation followed closely

continued on next page

continued from previous page

	File name	Incr	Format	Role
	MyRun%%GlobRelax.asc	y	ASCII	Some global relaxation quantities
	MyRun%%Duplic.asc	y	ASCII	Log file of particle duplications
	MyRun%%LogSauv.asc	y	ASCII	Log file of "externally" requested snapshots
	PID	n	ASCII	PID of run (written once)
	PARAMS.asc	n	ASCII	Parameters actually used by run (written once)
	CaracCode.asc	n	ASCII	Value of many preprocessor variables at compile time (written once)
	Units.asc	n	ASCII	Physical units used by run (written once)
	divers.asc	y	ASCII	Bits of info of dubious value (self-described)
	INFO	y	ASCII	Info about pid, host, start time & date (self-described)
	DONE	n	ASCII	To indicate that the run has finished (contains time and hostname)
–	_test_TMS.asc	n	ASCII	Relation mass–MS lifetime (for checking purposes)
–	_test_Remn.asc	n	ASCII	Relation ZAMS mass → remnant mass and type (for checking purposes)
–	_test_MRrelMS.asc	n	ASCII	Main-Sequence Mass-Radius relation (for checking purposes)
–	_test_ParamStellEvol.asc	n	ASCII	Parameters for stellar evolution (for checking purposes)

Table 6: Quantities written in master log file `MyRun%%Log.asc`.

Quantity	Explanation
	Physical data ("DONNEES PHYSIQUES")
`iPas_Evol`	Step number
`Tps_Ucode`	Cluster time in some mysterious, historical code units. It is the median time of all particles
`Tps_Ufokpl`	Time in Fokker-Planck units
`Tps_en_yr`	Time in yrs
`dTps_inf`	One sixth of particles have time smaller than `Tps_Ucode-dTps_inf`
`dTps_sup`	One sixth of particles have time larger than `Tps_Ucode+dTps_sup`
`Tps_moy`	Average time in code units
`Sigma_Tps`	Dispersion of particle times in code units
`NbSE_subsist`	Number of particles still in the cluster
`Net_subsist`	Number of stars still in the cluster
`M_amas`	Total stellar mass in the cluster in $N-$body units
`M_amas_en_Msol`	Total stellar mass in the cluster in $M_\odot$
`M_TN`	Mass of central object in $N-$body units
`M_rsrv`	Mass in the central gas reservoir in $N-$body units
`iType_CtrObj`	Type of central object (1 for MS, 5 for BH)
`Age_CtrObj_yr`	Evolutionary age of central object in yrs (if its is a VMS)
`dMejec_evap`	Mass lost by evaporation (total, $N-$body units)
`dMejec_coll`	Mass ejected from system form collisions
`dMejec_evst`	Mass ejected from system from stellar evolution
`dMejec_dechir`	Mass ejected from system form tidal disruptions
`dMaccr_coll`	Mass accreted by central object form collisions
`dMaccr_evst`	Mass accreted by central object form stellar evolution
`dMaccr_dechir`	Mass accreted by central object form tidal disruptions
`dMaccr_disp`	Mass accreted by central object form plunges through horizon
`dMaccr_captGW`	Mass accreted by central object from GW-inspirals
`Etot_amas`	Total energy of the system ($N-$body units)
`Ecin_amas`	Total kinetic energy of the stars ($N-$body units)
`Estell_grav`	Total gravitational energy of the stars ($N-$body units)
`dEtot_evap`	Change of total energy by evaporation (total, $N-$body units)
`dEtot_coll`	Change of total energy by collisions
`dEtot_evst`	Change of total energy by stellar evolution (mass loss)
`dEtot_dechir`	Change of total energy by tidal disruptions
`dEtot_disp`	Change of total energy by plunges through horizon
`dEtot_captGW`	Change of total energy by GW-inspirals
`dEtot_bin3c`	Change of total energy by 3-body binary heating (doe not work)
`dEtot_nk`	Change of total energy by natal kicks
`R_maree`	Tidal truncation radius ($N-$body units)
`Pot_ctr`	Central potential ($N-$body units)
`Rho_ctr`	Central density ($N-$body units)
`Sigma3D_ctr`	Central 3-D velocity dispersion ($N-$body units)
`Nb_relax`	Number of steps with relaxation ("super-encounters")
`Nb_coll`	Number of steps with collision
`Nb_kick`	Number of steps with large-angle scattering
`Nb_dechir`	Number of steps with tidal disruption
`Nb_disp`	Number of steps with plunge through horizon

continued on next page

continued from previous page

Quantity	Explanation
Technical data ("DONNEES TECHNIQUES")	
iPasEvol	Step number
Date	Computer date
Heure	Computer time
Tcpu	Number of CPU seconds used
Viriel	Virial ratio $-(E_{\rm kin}+E_{\rm tot})/E_{\rm tot}$ (should be close to 0)
Erreur_rel_E	Accumulated relative error in energy (should be close to machine precision)
Erreur_rel_M	Accumulated relative error in mass (should be close to machine precision)
Nb_moy_VN_pos_orb_par_pas	Number per step of rejections to find new orbital position (average since last write)
Nb_moy_parcrs_arbre_par_pas	Number per step of binary tree descents
Nb_moy_test_LC_par_pas	Number per step of test for loss-cone entry
Nb_RejetPaire	Number of pair rejections (since last write)
Nb_AnnulRelaxEvap	Number of "super-encounters" prevented because they would result in an ejection
Nb_AnnulRelaxCC	Number of "super-encounters" prevented because of the "central control"
Frac_Pas_Relax	Fractional number of steps with relaxation (i.e., super-encounters), generally 1
Dev_SuperRenc_moy	Average deflection angle in "super-encounters"
Dev2_SuperRenc_moy	RMS deflection angle in "super-encounters"
Frac_SuperRenc_Pi16	Fractional number of "super-encounters" with deflection angle $\theta_{\rm SE} \geq \pi/16$
Frac_SuperRenc_Pi8	idem, $\theta_{\rm SE} \geq \pi/8$
Frac_SuperRenc_Pi4	idem, $\theta_{\rm SE} \geq \pi/4$
Frac_SuperRenc_Pi2	idem, $\theta_{\rm SE} \geq \pi/2$
dR_rel_Renc_moy	Average relative difference in radius for encountering pairs ($\delta_R \equiv 2(R_2-R_1)/(R_1+R_2)$)
dR_rel_Renc_max	Maximum relative difference in radius for encountering pairs
Frac_Renc_dR_001	Fraction of encounters with $\delta_R \geq 0.01$
Frac_Renc_dR_01	idem, $\delta_R \geq 0.1$
Frac_Renc_dR_05	idem, $\delta_R \geq 0.5$
Frac_Renc_dR_1	idem, $\delta_R \geq 1$

References

Baumgardt, H. 2001, MNRAS, 325, 1323

Binney, J. & Tremaine, S. 1987, Galactic Dynamics (Princeton University Press)

Dehnen, W. 1993, MNRAS, 265, 250

Freitag, M. 2001, Classical and Quantum Gravity, 18, 4033

—. 2003, ApJ Lett., 583, L21

Freitag, M., Amaro-Seoane, P., & Kalogera, V. 2006a, ApJ, 649, 91

Freitag, M. & Benz, W. 2001, A&A, 375, 711

—. 2002, A&A, 394, 345

—. 2005, MNRAS, 358, 1133

Freitag, M., Gürkan, M. A., & Rasio, F. A. 2006b, MNRAS, 368, 141

Freitag, M., Rasio, F. A., & Baumgardt, H. 2006c, MNRAS, 368, 121

Fukushige, T. & Heggie, D. C. 2000, MNRAS, 318, 753

Heggie, D. C. & Mathieu, R. D. 1986, in The Use of Supercomputers in Stellar Dynamics, ed. P. Hut & S. L. W. McMillan (Springer-Verlag), 233

Hénon, M. 1973, in Dynamical structure and evolution of stellar systems, Lectures of the 3rd Advanced Course of the Swiss Society for Astronomy and Astrophysics (SSAA), ed. L. Martinet & M. Mayor, 183–260

Hénon, M. 1975, in IAU Symp. 69: Dynamics of Stellar Systems, ed. A. Hayli, 133–149

Hénon, M. H. 1971a, Ap&SS, 13, 284

—. 1971b, Ap&SS, 14, 151

Rauch, K. P. 1999, ApJ, 514, 725

Spitzer, L. 1987, Dynamical evolution of globular clusters (Princeton University Press)

Tremaine, S., Richstone, D. O., Byun, Y., Dressler, A., Faber, S. M., Grillmair, C., Kormendy, J., & Lauer, T. R. 1994, AJ, 107, 634

A License