1 MCLUSTER MANUAL

The tool McLuster is an open source programme that can be used to either set up initial conditions for N-body computations or, alternatively, to generate artificial star clusters for direct investigation. There are two different versions of the code, one basic version for generating all kinds of unevolved clusters (in the following called `mcluster`) and one for setting up evolved stellar populations at a given age. The former is completely contained in the C file `main.c`. The latter (here dubbed as `mcluster_sse`) is more complex and requires additional FORTRAN routines, namely the Single-Star Evolution (SSE) routines by Hurley, Pols & Tout (2000) which are provided with the McLuster code. For a quick introduction read the README file which is also provided with the code. For technical details on how to generate initial conditions for star cluster in general we would like to refer to Kroupa (2008) and referenced literature within.

1.1 Compilation

After extracting the archive which can be obtained from the given web address[1], the basic version `mcluster` can be compiled on a UNIX system from the command line with

```
> cc -lm -o mcluster main.c
```

where `cc` may be replaced by the C-compiler available on your computer. It can also be compiled by using the Makefile, i.e.

```
> make mcluster
```

For the more complex version `mcluster_sse` the Makefile has to be used. Type

```
> make mcluster_sse
```

after which the programme should compile, generating an executable named `mcluster_sse`.

- Note that, when using the Makefile, you may have to change the C- and/or FORTRAN-compiler entry as well as the path of your compiler.
- In case you want to apply any change to the code make sure that you first delete all object files by typing

```
> make clean
```

before re-compiling the code.

1.2 Input

There are two ways of choosing the desired cluster parameters. One is to set the parameters manually within `main.c`, within the upper part of the code right at the beginning of the `main` routine where all variables are declared. Note that, after changing the value of a variable, you have to compile the code again. The more convenient way is therefore to pass arguments to the code at the time of execution via the command line (see Tab. 1 for an overview of available options). Type

```
> mcluster -h
```

or

```
> mcluster_sse -h
```

respectively, to get a quick help on the available parameters and their usage.

- In case you have not specified a certain parameter, the default value as set within the `main` routine is used.
- Not all parameters can be set via the command line, some have to be changed within the code.
- All command line arguments are the same for `mcluster` and `mcluster_sse`, except for the age parameter (`-e`) which is mentioned in more detail below.

1.3 Density profile

McLuster can generate star clusters with various radial density profiles (option `-P`):

(i) The simplest option is the analytical Plummer (1911) profile (option `-P0`). For this profile only the half-mass radius has to be specified additionally (option `-R`, in parsec). The Plummer profile is in principle infinitely extended but gets automatically truncated at the theoretical tidal radius of the cluster in case a tidal field has been specified (see below).

(ii) A more sophisticated set of models is given by the distribution function of King (1966). For this profile (option `-P1`) the half-mass radius and the value of W_0 (option `-W` ranging from 2.0–12.0) has to be specified. The underlying routine for generating the density distribution from the distribution function is based on KING0 by Douglas C. Heggie (e.g. Heggie & Aarseth 1992). Note that the final density distribution gets scaled to exactly match the desired half-mass radius; the King radius at which the density becomes zero does not necessarily match the theoretical tidal radius in this case.

(iii) Šubr, Kroupa, & Baumgardt (2008) give a density profile which can be chosen to be mass segregated. For this density profile (option `-P2`) the half-mass radius and an additional mass-segregation parameter (option `-S`, ranging from 0.0–0.99, but $S < 0.5$ for reasonable models in virial equilibrium) have to be chosen. For `-S0.0` the Šubr profile is equal to a Plummer profile. McLuster uses a slightly modified version of the Plumix routine by Ladislav Šubr to set up this kind of profile.

(iv) Young clusters in the Large Magellanic Cloud were found to follow a two-dimensional density profile consisting of a core and a power-law tail without visible tidal truncation. Elson, Fall & Freeman (1987) give a simple analytical formula for such profiles which can be deprojected with McLuster and used to set up 3D star cluster models (option `-P3`). Since those models are in principle infinitely extended, the so-called EFF models do not get scaled to a certain half-mass radius, but rather require specification of a cut-off radius to which the profile should extend (option

[1] `www.astro.uni-bonn.de/~akuepper/mcluster/mcluster.html` or `www.astro.uni-bonn.de/~webaiub/english/downloads.php`

[htdp]

Table 1. Overview of available command line options in McLuster. For details on the available choices see the corresponding paragraphs.

Option	Range	Meaning
-N	$0 < N < Nmax$	Number of cluster stars
-M	$M > 0$	Mass of the cluster (specify either N or M)
-P	0/1/2/3/-1	Density profile (Plummer/King/Šubr/EFF/homogeneous sphere)
-W	1.0–12.0	W_0 parameter for the King model ($P = 1$)
-R	$R > 0$	Half-mass radius in parsec (ignored for $P = 3$)
-r	$0 < r < c$	Scale radius of the EFF template in parsec ($P = 3$)
-c	$c > r$	Cut-off radius of the EFF template in parsec ($P = 3$)
-g	$g > 1.5$	Power-law slope of the EFF template ($P = 3$)
-S	0.0–1.0	Degree of mass segregation (0.0 means no segregation; $S < 1.0$ for $P = 2$!)
-D	1.6–3.0	Fractal dimension (3.0 means no fractality)
-T	$T > 0$	Nbody4/6 computation time in N-body units
-Q	$Q \geqslant 0$	Virial ratio ($Q = 0.5$ for virial equilibrium)
-C	0/1/3	Output type (Nbody6/Nbody4/ASCII table)
-A	$A > 0$	Nbody4/6 adjustment time in N-body units
-O	$O > 0$	Nbody4/6 output time in N-body units
-G	0/1	Use GPU with Nbody6 (no/yes)
-o	—	Output name of the cluster model
-f	0/1/2	IMF (no IMF/Kroupa IMF/user defined)
-a	$a > 0$	IMF slope for a user defined IMF, may be used multiple times, from low to high mass
-m	$m > 0$	IMF mass limit ($M_\odot$) for a user defined IMF, may be used multiple times, from low to high mass
-B	0–$N/2$	Number of binary systems
-b	0.0–1.0	Binary fraction (specify either B or b)
-p	0/1	Binary pairing (random/ordered for $M > 5\,M_\odot$)
-s	$s \geqslant 0$	Seed for randomization, set 0 for randomization by local time
-t	0/1/2/3	Tidal field (no tidal field/near-field approximation/point-mass potential/Milky-Way potential)
-e	$e \geqslant 0$	Epoch for stellar evolution in Myr (only available in `mcluster_sse`)
-Z	0.0001-0.03	Metallicity ($Z = 0.02$ for solar metallicity)
-X	$X \geqslant 0$	Galactocentric radius vector in parsec (use 3 times for x-, y- and z-coordinate)
-V	$V \geqslant 0$	Cluster velocity vector in km/s (use 3 times for x-, y- and z-coordinate)
-u	0/1	Output units (N-body units/astrophysical units)
-h/-?	—	Display help

`-c`, in parsec). The central density then gets calculated automatically using the specified cluster mass (see below). In addition, the radius of the two-dimensional core (option `-r`, in parsec) and the slope of the power-law part of the profile (option `-g`) have to be chosen.

(v) A further possibility to set up the density distribution is given by option `-P-1`. In this case the final cluster has no density gradient, but consists of stars which are homogeneously distributed within a sphere. The size of the sphere is specified by choosing the half-mass radius. This option is especially useful in case of fractal initial conditions (see below).

• The exact matching of the actual half-mass radius to the specified value may be switched off within the `main` routine, but is not recommended. Set `match = 0` and compile the code again.

1.4 Tidal field

As mentioned above, the choice of the tidal field and of the cluster orbit may influence the extent of the density profile. McLuster offers different kinds of tidal fields which can be specified with the option `-t`. In addition it may be necessary to specify the galactocentric radius vector and the orbital velocity vector. This can be done by using the option `-X` (in parsec) and `-V` (in km/s), respectively, three times for the x-, y- and z-component (within a Cartesian coordinate system where the galactic disk would lie in the x-y-plane, if applicable). As an example, `-X8500.0 -X0.0 -X0.0 -V0.0 -V220.0 -V0.0` would give the motion of the Local Standard of Rest. This option is especially useful when generating input for N-body computations, since the input files are automatically adjusted accordingly.

(i) For a cluster in isolation choose `-t0`. No truncation is applied to profiles like the Plummer profile in this case.

(ii) A linearized tidal field, as described in Fukushige & Heggie (2000) can be chosen with `-t1`. If you have selected to generate input files for Nbody6 (see below) then the values for the Local Standard of Rest are used. In all other cases the galactocentric radius has to be specified additionally. Therefore use the option `-X` and set all but one component to zero, e.g. `-X6000.0 -X0.0 -X0.0` for an orbit at 6 kpc. No orbital velocity vector has to be specified as the linearized tidal field mimics a circular orbit.

(iii) If you choose `-t2` then you get a point-mass galaxy for which you can specify any galactocentric radius and orbital velocity. The mass of the galaxy is set within the header of the `main` routine. By default, `M1pointmass` is set such that you get an orbital velocity of 220 km/s at a galactocentric radius of 8.5 kpc.

(iv) For a more realistic, Milky-Way potential you can choose option `-t3`. This potential consists of a central point mass/bulge, modelled as a Hernquist potential (Hernquist 1990), a Miyamoto disk (Miyamoto & Nagai 1975) and a log-

arithmic (dark-matter) halo, with values as given in Allen & Santillan (1991). If you set up initial condition for NBODY6 then the logarithmic halo will be adjusted such that the circular velocity, `VCIRC`, at some radius, `RCIRC`, has a specific value. As default this is 220 km/s and 8.5 kpc, respectively, which may be changed in the header of the `main` routine. The other parameters of this potential may also be changed there.

1.5 Cluster mass and stellar mass function

You can either fix the total number of stars in your cluster (option `-N`) or you can set a desired total mass (option `-M`, in solar units). In the latter case, MCLUSTER draws stars from the selected mass function until the desired mass is exceeded. The mass function of stars in the cluster can be defined to be one of the following three kinds (option `-f`).

(i) All stars can have the same mass (option `-f0`). The mass of each star is by default assumed to be $1\,\mathrm{M}_\odot$, which may be changed within the `main` routine (parameter `single_mass`).

(ii) The canonical Kroupa (2001) initial mass function can be used with `-f1`. This IMF has a slope of $\alpha_1 = 1.3$ for stellar masses $m = 0.08 - 0.5\,\mathrm{M}_\odot$, and the Salpeter slope $\alpha_2 = 2.3$ for $m > 0.5\,\mathrm{M}_\odot$. The lower and upper IMF limit, `mlow` and `mup`, are by default $0.08\,\mathrm{M}_\odot$ and $100\,\mathrm{M}_\odot$, respectively, but these values may be changed in the `main` routine.

(iii) More sophisticated, multi-power-law IMFs can be set up with option `-f2`. Therefore, MCLUSTER uses the routine MUFU by Ladislav Šubr. This routine allows to define several mass limits and corresponding mass-function slopes between these limits. The limits and the slopes can be passed to MCLUSTER with the options `-m` and `-a`, respectively. These options have to be used multiple times, where one more limit has to be passed to MCLUSTER than number of slopes. For example, the Kroupa IMF with a steeper slope of $\alpha_3 = 2.7$ for stars more massive than $5\,\mathrm{M}_\odot$ up to a maximum mass of $80\,\mathrm{M}_\odot$ would be `-f2 -m0.08 -a1.3 -m0.5 -a2.3 -m5.0 -a2.7 -m80.0`.

• The cluster mass to maximum stellar mass relation found by Weidner & Kroupa (2006) can be used to automatically cut off the IMF at the corresponding upper mass limit. This routine is switched off by default but may be activated by setting the `weidner` parameter in the `main` routine to 1.

• In MCLUSTER there is a maximally allowed stellar mass limit defined through the parameter `upper_IMF_limit`. This parameter is set to $100\,\mathrm{M}_\odot$ since NBODY6, i.e. the stellar evolution routine SSE within NBODY6, does not allow higher masses. In case you need higher stellar masses anyway, set this parameter within the `main` routine to the desired value.

1.6 Mass segregation

With MCLUSTER it is possible to apply any degree of primordial mass segregation to all available density profiles, not only to the Šubr profile as already mentioned above. Therefore, the method as described in Baumgardt, De Marchi & Kroupa (2008a) is used. The advantage of this method is that the chosen density profile is not changed with increasing degree of mass segregation as is the case for the Šubr profile. In short, it works as follows: for a cluster of N stars MCLUSTER first draws the stellar masses from the selected IMF (see above) and then creates $N' = N\langle m\rangle/m_{low}$ orbits, where $\langle m\rangle$ is the mean stellar mass and m_{low} the lowest stellar mass in the cluster. These orbits get ordered by their specific energy, from low energy to high energy orbits. Then the stellar masses are also ordered and the cumulative mass function, $M_{cum}(i) = \sum_{j=1}^{i} M(j)$ is evaluated from this mass array. After dividing $M_{cum}(i)$ by the total cluster mass, the function is normalized such that it runs from 0 to 1. Finally, for any star an orbit from the list of energy-ordered orbits is chosen from the orbits between $N'M_{cum}(i-1)$ and $N'M_{cum}(i)$.

If the masses in the mass array are perfectly ordered from highest to lowest then this will yield a completely mass segregated cluster. This is, the highest mass star is on the lowest energy orbit, and the lowest mass star is on the highest energy orbit. Intermediate degrees of mass segregation can be achieved by non-perfect ordering. In MCLUSTER this is realized as follows: first, all N stellar masses are ordered from highest to lowest. Then, beginning with the highest mass, the masses are written to a new array, where the i-th massive star is written to the j-th empty slot counting from 0 to $N - i$. j is generated using

$$j = (N - i)\left(1 - X^{1-S}\right), \qquad (1)$$

where X is a random number between 0 and 1, and S is the mass segregation parameter. When S is zero, j can have all values from 0 to $N-i$ and we end up with a random distribution. But when S is 1, then j is always zero and we reproduce the perfectly ordered array we started with. This is, because every star i, beginning with the most massive, gets written to the next empty slot which is slot i. By choosing S values between 0 and 1 we can get intermediate degrees of partial mass segregation (option `-S`). Unlike for the Šubr profile, S can explicitly be chosen to be 1.0. Moreover, for all values of S we get clusters in virial equilibrium (if not explicitly specified differently) with the desired (mass) density profile.

1.7 Fractality

Star clusters are not born in perfect symmetry. They are rather formed in collapsing, fractal molecular clouds (see e.g. ???). With MCLUSTER you can set up two kinds of fractal initial conditions. First, you can set up a fractal distribution of stars within a sphere of constant average density, similar as described in Goodwin & Whitworth (2004). Secondly, you can add fractal substructure to any of the above given density profiles.

(i) When you choose a homogeneous density profile (option `-P-1`) the cluster stars get distributed within a sphere as follows. The first star (a so-called parent) is placed in the middle of a box of size 2 (arbitrary units), then this box is split into 8 sub-boxes of half the initial box size. In the centre of each sub-box a further star is placed (a so-called child), whereupon each sub-box is split up into 8 smaller pieces, such that each child now becomes a parent on its own. By applying a small random offset to each star from its sub-box centre, we make sure that the final cluster does not look too grid-like. This is repeated until we have generated $128.0 \times 8^{\log(N)/\log(8)}$ stars or until the total number of

stars would exceed this number with the next generation of children. From these stars we randomly draw N stars with radial distances of less than unity from the centre of the initial box. We end up with a homogeneous sphere of stars.

Now, if not every sub-box gets a new star, and only those sub-boxes get sub-divided which have a star in their centre, then the final distribution of stars becomes fractal. The probability that a sub-box gets a star can be expressed as $2^{(D-3)}$, where D is the fractal dimension (option `-D`). If D is chosen to be 3.0 then we get no fractality since the probability is unity, i.e. every parent gets 8 children. If it is, e.g., 2.0 then only every second sub-box gets a star, or, on average, every parent has 4 children.

Corresponding stellar velocities are drawn from a Gaussian distribution, and re-scaled such that all children of one parent are in virial equilibrium and the total mean velocity in one sub-group is unity (arbitrary units). In addition, each child gets the velocity of its parent. In a later step, McLuster re-scales the phase-space coordinates of the stars such that the cluster is in virial equilibrium (if not specified differently). In this way, we get a fractal structure consisting of coherently moving, gravitationally bound substructures.

(ii) Alternatively, we can choose to set up fractal clusters which follow a given density profile like, e.g., the Plummer profile or the King profile, but which show fractal substructure. This is realized by first generating a sphere of stars of radius unity with the above procedure. But now this distribution of stars gets folded with the chosen density profile. Therefore the radius of each star first gets re-scaled by its absolute value to the power of three. In this way, we get N stars with radii distributed homogeneously between 0 and 1, but which show sub-structure in 6D phase space. These radii are used as seeds to compute a corresponding radius within the specified density profile. In a last step, the space coordinates of each star get scaled by this newly generated radius. In this way the fractal distribution is conserved but folded with the specific density profile. Moreover, the velocity of each star is scaled to the expected velocity of a star at the given radius within the specified density profile.

• In order to achieve a minimum of spherical symmetry, you can tell McLuster to give 8 children to the first "urparent". In this way, you avoid too large asymmetries. This will lead to less differing results between initial conditions generated with different random seeds, but does on the other hand not yield perfectly fractal clusters. This tweak can be switched off within the header of the `main` routine. Set the `symmetry` parameter to 0 and re-compile.

• Once in a while McLuster gets stuck in the fractality sub-routine. In this case a restart with a different random seed should help.

1.8 Binaries

After the stellar masses got drawn from an initial mass function, you can choose to let McLuster set up binary systems. You can specify either the desired number of binary systems (option `-B`, values from 0 to $N/2$) or you can specify a fraction of stars which should be in binary systems (option `-b`, values from 0 to 1). Thus, from all N stars, $N \times b/2$ or B binaries are formed, respectively. The binaries are then replaced by a centre-of-mass (CoM) particle for the rest of the procedure. Only in the very end, after the density profile has been established and the velocities of the cluster members have been scaled appropriately, the CoM particles get replaced by their two constituent stars. The internal properties of the binaries can be generated according to the following semi-major axis distributions which can be selected in the header of the `main` routine (parameter `adis`).

(i) A flat distribution of semi-major axes can be specified with `adis = 0`. In addition, you have to choose a minimum and a maximum semi-major axis (parameters `amin` and `amax`).

(ii) If `adis` is set to 1 (default) then the semi-major axis of each binary is computed from a period which was drawn from the Kroupa (1995a) period distribution.

(iii) If you want the semi-major axes to be generated from the Duquennoy & Mayor (1991) period distribution then you have to set `adis = 2`.

• The pairing of primary and secondary components of the binaries can be chosen to be either random or ordered. In the latter case (option `-p1`) the stellar masses above a certain threshold (parameter `msort` in the `main` routine) get ordered from most to least massive. The rest of the stars are put in random order onto the list below the last star with mass above `msort`. The pairing of binaries now starts with the most massive star which gets paired together with the second-most massive, then the third with the forth, and so on down the list. The default value of `msort` is $5\,\mathrm{M}_\odot$, in rough agreement with recent findings (e.g. Kobulnicky & Fryer 2007).

• Eccentricities, e, are drawn from a thermal eccentricity distribution, i.e. $f(e) = 2e$ (e.g. Duquennoy & Mayor 1991).

• To correct for the fact that short-period binaries in the Milky Way do not show high eccentricities (Mathieu 1994), Kroupa (1995b) introduces an analytical correction for such systems, which is attributed to, so-called, pre-main sequence eigenevolution between the constituent stars. This correction can be switched off by setting the parameter `eigen = 0` in the `main` routine.

1.9 Stellar evolution

McLuster contains the SSE routines by Hurley, Pols & Tout (2000) which are also used in, e.g., Nbody6 for stellar evolution. If you just want to generate star clusters consisting of zero-age main sequence (ZAMS) stars then you only need the basic version `mcluster`. But if you want to set up a cluster with an evolved stellar population then you have to use `mcluster_sse` which makes use of those SSE routines. In this case you have to specify an age for the cluster population (option `-e`, in Myr). The evolution of the stars is done in the very beginning of the programme. When the stars are drawn from the IMF they get immediately evolved to the desired age. The masses of the evolved stars are then summed up and additional stars are generated until the desired cluster mass is exceeded or the desired number of stars is reached. The stellar parameters derived from SSE for each star are stored in an additional file, which also has to be passed to Nbody6 (see below).

• The internal parameters of SSE can be changed within the header of the `main` routine (not recommended).

• In case a star becomes a neutron star or a black hole SSE assigns a kick velocity to the remnant (if not specified differently). The kick velocity can be used to remove the remnant from the cluster. Therefore the present-day escape velocity of the cluster at its half-mass radius is calculated and if the kick velocity exceeds this velocity it gets removed. If you want to keep all compact remnants set the parameter `remnant = 0` in the `main` routine.

• The metallicity, Z, can be set with option `-Z`. Alternatively, you can specify the metallicity as [Fe/H] within the `main` routine, the corresponding Z value is computed using the relation given in Bertelli et al. (1994). Make sure that in this case you set $Z = 0$ beforehand.

• In case you are generating binaries and have selected ordered pairing for stars above a certain mass, `msort` (see above), then the ZAMS mass is used to decide whether a star is paired randomly to another star or not.

• The components of binaries are independently evolved as single stars with SSE. For a more advanced treatment of stars in binaries, the Binary-Star Evolution (BSE) routines by Hurley, Pols & Tout (2002) are also included in McLuster. Therefore, at the very end of the cluster generation procedure, when the binary CoM particles are replaced by their constituents and the orbital elements are generated, the masses of the components are reset to their ZAMS mass. Then the two stellar masses, a semi-major axis and an eccentricity are passed to BSE which finally returns corresponding evolved values. This feature is switched off by default but may be activated by setting the parameter `BSE` to 1 in the `main` routine.

1.10 Output

Up to now McLuster can generate input for Nbody6 (option `-C0`, Aarseth 2003) and Nbody4 (option `-C1`), or it can write an ASCII table of stars and their properties (option `-C3`).

(i) In the first and second case, there will be two output files which can be named with option `-o`. For example, `-o mycluster` will yield the files `mycluster.input`, containing all the input parameters for the run, and `mycluster.fort.10`, containing the masses, positions and velocities. Note that the latter has to be renamed to `fort.10` at the time of execution in order to be recognized by Nbody4/6. When using `mcluster_sse` there will be another file named `mycluster.fort.12`. This file also has to be renamed within the directory of the run to `fort.12`. The name `mycluster` is just added to the file names for convenience. Thus, a directory for a run should contain:

(a) `mycluster.input`,
(b) `fort.10`,
(c) `fort.12`.

The run is then started with the usual command, i.e.,

```
> /.../nbody6 < mycluster.input
```

where `/.../` should be replaced by the path to your Nbody6 installation.

(ii) In the case of `-C3` a file named `mycluster.txt` will be created containing mass, positions (x, y, z) and velocities (v_x, v_y, v_z). If you are using `mcluster_sse` then this file will also contain for each star the ZAMS mass[2], the stellar type[3], the epoch of the star[4], its spin[5], its radius[6], its luminosity[7], its age in Myr, its metallicity (Z), its absolute V magnitude, its apparent V magnitude[8], $B-V$, its effective temperature (K), an random error for the V magnitude, and a random error for $B-V$. The last six are generated assuming observation with an 8m-class telescope from a distance `Rgal` (see Küpper, Mieske & Kroupa 2011).

• In addition you have to specify whether you want the output to be in N-body units (see e.g. Heggie & Hut 2003, option `-u0`) or astrophysical units (option `-u1`). For Nbody6 and Nbody4 this should always be N-body units.

• With the ASCII table output you can easily draw a colour-magnitude diagram. Use columns 18(+21) versus 17(+20) for a diagram showing $B-V$ versus apparent V magnitude (+random errors).

• McLuster automatically computes a radial density profile and a cumulative radial density profile. Both are by default printed to the screen. This may be switched off within the `main` routine (parameters `create_radial_profile = 0` or `create_cumulative_profile = 0`, respectively).

1.11 Miscellaneous

• The virial ratio, $Q = -E_{kin}/E_{pot}$, where E_{kin} is the total kinetic energy of the single cluster stars and E_{pot} their potential energy, can be set with the parameter `-Q`. Note that this only affects the input file for N-body computations but not the stellar velocities in the table of stars (there the virial ratio will always be 0.5, i.e. virial equilibrium). The velocities get scaled within Nbody4/6 according to your choice of Q.

• The random seed can be set with the parameter `-s` which can be any positive integer. In the case of `-s0` McLuster will take the local time as random seed.

• There is an upper limit of temporary stars or orbits within McLuster. This number, `Nmax`, is set to 1.500.000 by default, i.e. McLuster allocates memory accordingly. When applying mass segregation or fractality to a cluster, many more temporary stars/orbits have to be generated than finally needed. Especially if a cluster shall be mass segregated and fractal at the same time this number may easily be exceeded. In this case you should increase it in the `main` routine.

• A few more parameters and command line arguments are available (see option `-h` and the header of the `main` routine) which mostly affect flags for N-body computations.

[2] from SSE, in $M_\odot$
[3] from SSE, see Hurley, Pols & Tout 2000
[4] from SSE, in Myr
[5] from SSE, in km/s
[6] from SSE, in solar units
[7] from SSE, in solar units
[8] assuming a distance `Rgal` from the observer which can be changed in the `main` routine

1.12 Examples

If no command line arguments are passed to McLuster it will use the default parameter values which are specified and which can be changed within the `main` routine. By typing

```
> mcluster
```

a file `test.txt` is created. This default cluster has $1000\,M_\odot$ ($\sim$ 1800 stars), a half-mass radius of 0.8 pc, a Plummer density distribution, is in a Milky-Way tidal field with LSR values, uses the Kroupa IMF and has no binaries. The entries in the data table are in astrophysical units. With

```
> mcluster -C0 -u0
```

the same cluster is written to the files `test.input` and `test.fort.10` but in N-body units. This can be passed to Nbody4/6 as stated above. The clusters used in this work were created using, e.g.,

```
> mcluster -M100000.0 -P3 -r0.1 -c20.0 -g2.0 -S1.0
-C0 -G1 -o R136 -f1 -b0.2 -p1 -s2 -Z0.01 -u0
```

for the fully mass segregated N-body model. The arguments stand for: a total mass of $100.000\,M_\odot$ (`-M`), the EFF density profile (`-P`) with a 2D core radius of 0.1 pc (`-r`), a cut-off radius of 20 pc (`-c`) and a 2D power-law slope of -2 (`-g`). It is completely mass segregated (`-S`), the output is for Nbody6 (`-C`), and we use a GPU (`-G`). The output is named R136 (`-o`), we use a Kroupa IMF (`-f`), 20% binaries (`-b`) and ordered pairing for massive stars (`-p`). The random seed of our model is 2 (`-s`) and the metallicity is 0.01 (`-Z`). The output is in N-body units (`-u`) since we want to pass it to Nbody6.

ACKNOWLEDGMENTS

The authors are grateful to Sverre Aarseth for making his Nbody codes accessible to the public. Moreover, they would like to thank Douglas Heggie, Ladislav Šubr and Jarrod Hurley for allowing the usage of their codes within McLuster. The code makes use of the Numerical Recipes (Press, Flannery & Teukolsky 1986).

REFERENCES

Aarseth S. J., 2003, Gravitational N-Body Simulations, Cambridge, UK, Cambridge University Press
Allen C., Santillan C., 1991, RMxAA, 22, 255
Baumgardt H., De Marchi G., Kroupa P., 2008a, ApJ, 685, 247
Bertelli G., Bressan A., Chiosi C., Fagotto F., Nasi E., 1994, A&AS, 106, 275
Duquennoy A., Mayor M., 1991, A&A, 248, 485
Elson R. A. W., Fall S. M., Freeman K. C., 1987, ApJ, 323, 54
Fukushige T., Heggie D. C., 2000, MNRAS, 318, 753
Goodwin S. P., Whitworth A. P., 2004, A&A, 413, 929
Heggie D. C., Aarseth S. J., 1992, MNRAS, 257, 513
Heggie D., Hut P., 2003, The Gravitational Million-Body Problem, Cambridge, UK, Cambridge University Press
Hernquist L., 1990, ApJ, 356, 359
Hurley J. R., Pols O. R., Tout C. A., 2000, MNRAS, 315, 543
Hurley J. R., Pols O. R., Tout C. A., 2002, MNRAS, 329, 897
King I. R., 1966, AJ, 71, 64
Kobulnicky H. A., Fryer C. L., 2007, ApJ, 670, 747
Kroupa P., 1995a, MNRAS, 277, 1491
Kroupa P., 1995b, MNRAS, 277, 1507
Kroupa P., 2001, MNRAS, 322, 231
Kroupa P., 2008, LNP, 760, 181
Küpper A. H. W., Mieske S., Kroupa P., 2011, MNRAS, in press, arXiv:1012.3163
Mathieu R. D., 1994, ARA&A, 32, 465
Miyamoto M., Nagai R., 1975, PASJ, 27, 533
Plummer H. C., 1911, MNRAS, 71, 460
Press W. H., Flannery B. P., Teukolsky S. A., 1986, Numerical recipes. The art of scientific computing, Cambridge, UK, Cambridge University Press
Šubr L., Kroupa P., Baumgardt H., 2008, MNRAS, 385, 1673
Weidner C., Kroupa P., 2006, MNRAS, 365, 1333