User Guide for mkgalaxy

version of November 23, 2007
Paul McMillan & Walter Dehnen

Summary. This documentation describes the how to use the script **mkgalaxy** and the falcON programs **mkhalo** and **mkWD99disc** to build a halo-bulge-disc galaxy model. For a full explanation of the ideas behind the programs, see McMillan & Dehnen (2007) and references therein.

0 Using mkgalaxy

The script **mkgalaxy** has been provided as an (hopefully) easy-to-use way to generate galaxy initial conditions according to McMillan & Dehnen (2007). In order to invoke it, type at the command line

mkgalaxy `name=`*name* [parameter list]

where "[parameter list]" refers to an optional list of parameters given in the same way as NEMO keywords, i.e. '*parameter_name=value*' for instance 'Nh=1000000'. A complete list of all parameters recognised as well as their default settings is given in table 1, while the remaining sections explain their detailed meaning. Upon successful completion, the script should have generated the following files:

name.err	error output (with a list of actual parameters, all debugging info and error messages)
name.S	NEMO snapshot with halo & bulge adjusted to full disc
name.grow	log output from the **gyrfalcON** simulation (see section 2)
name.snp	NEMO snapshot with full galaxy model

By using the default settings, a galaxy consisting of a truncated Dehnen & McLaughlin (2005) halo, a Hernquist (1990) bulge, and an exponential disc with, respectively, 1200000, 40000, and 200000 bodies will be created. In order to deviate from this default, users may either override the defaults by command-line parameters, or edit (a copy of) the script before running it, or both.

If the parameter `spheroid` is given, it must refer to a NEMO snapshot file containing the halo & bulge component adjusted to the full disc, i.e. a file like *name*.S generated by **mkgalaxy**. In this case, **mkgalaxy** will skip the generation of *name*.S (described in sections 1 and 2 below) and instead use the file referred to by `spheroid`. **N.B.** this works correctly *only* if the values of the parameters marked by a '$*$' in table 1 are identical to those used in the generation of the file referred to by `spheroid`. **mkgalaxy** has no way to check that this is actually the case (except for checking that the total number of bodies matches), and therefore extra care is required by the user. **You have been warned!**

The script **mkgalaxy** as well as the falcON programs it uses are not heavily tested, in particular for other than the default parameters. Therefore, errors ("bugs") may still be present. Please report any anomalies to Walter ¡wd11@astro.le.ac.uk¿ (ideally, send the error output in *name*.err generated with a value for `debug`, i.e. 10).

The following sections describe in some detail the process encoded in **mkgalaxy** and its usage of the falcON programs **mkhalo** and **mkWD99disc**.

1 Building the initial spheroid models

In the first step, we generate spherical initial conditions for halo and (if desired) bulge in the presence of the monopole part of the disc potential. Note that we first sample only half of the intended number of bodies and double them later (described in section 2).

The bulge and halo models are both built with the program **mkhalo**, which recognises parameters characterising the spheroid properties and describing any additional (spherical) gravitational potential (e.g. the monopole of the disc).

parameter	*	default	meaning
disc parameters			
Md	*	1	disc mass
Nd		200000	number of disc bodies
Rd	*	1	disc scale radius R_d
Zd	*	0.1	disc scale height z_d
Rsig		0	if $\neq 0$: scale radius R_σ for σ_R
Q		1.2	Toomre's Q: constant if $R_\sigma = 0$, otherwise $Q(R_\sigma) = \mathtt{Q}$
Nbpo		50	number of disc bodies sampled per orbit
ni		4	number of iterations in disc sampling
epsd		0.01	gravitational softening length ϵ_i for disc bodies
halo parameters			
Mh	*	24	halo mass
Nh	*	1200000	number of halo bodies
innerh	*	1	halo inner logarithmic density slope γ_0
outerh	*	3	halo outer logarithmic density slope γ_∞
etah	*	1	halo transition exponent η
Rcoreh	*	0	halo core radius r_c
Rh	*	6	halo scale length r_s
Rth	*	60	halo truncation radius r_t; Rth = 0 is interpreted as $r_t = \infty$
betah		0	halo anisotropy parameter β_0
r_ah		0	halo Ossipkov-Merrit anisotropy radius r_a; r_ah = 0 is interpreted as $r_a = \infty$
epsh		0.02	gravitational softening length ϵ_i for halo bodies
bulge parameters			
Mb	*	0.2	bulge mass
Nb	*	Nd(Mb/Md)	number of bulge bodies
innerb	*	1	bulge inner logarithmic density slope γ_0
outerb	*	4	bulge outer logarithmic density slope γ_∞
etab	*	1	bulge transition exponent η
Rcoreb	*	0	bulge core radius r_c
Rb	*	0.2	bulge scale length r_s
Rtb	*	0	bulge truncation radius r_t; Rtb = 0 is interpreted as $r_t = \infty$
betab		0	bulge anisotropy parameter β_0
r_ab		0	bulge Ossipkov-Merrit anisotropy radius r_a; r_ab = 0 is interpreted as $r_a = \infty$
epsb		epsd	gravitational softening length ϵ_i for bulge bodies
parameters controlling code			
kmax		3	maximum timestep $\tau_{\max} = 2^{-\mathtt{kmax}}$
kmin		7	minimum timestep $\tau_{\min} = 2^{-\mathtt{kmin}}$
fac		0.01	time step control: $\tau_i < \mathtt{fac}/|\boldsymbol{\nabla}\Phi|$
fph		0.04	time step control: $\tau_i < \mathtt{fph}/|\Phi|$
tgrow		40	disc growth time
seed		1	seed for random number generators
nmax		12	$n_{\max}$ in potential expansion
lmax		8	$l_{\max}$ in potential expansion
careful		t	t(rue) or f(alse): allow for non-monotonic halo/bulge distribution function?
debug		2	debugging level used when running falcON programs
spheroid			*optional*: file with halo & bulge adjusted to full disc

Table 1: List of keyword recognised by **mkgalaxy**; for detailed explanations see text.

1.1 Spheroid properties

The spheroid density profile is described by the parameters $M = \texttt{M}$ (total mass), $\gamma_0 = \texttt{inner}$ (inner logarithmic density slope), $\gamma_\infty = \texttt{outer}$ (outer logarithmic density slope), $\eta = \texttt{eta}$ (transition strength between inner and outer power-law), $r_c = \texttt{r\_c}$ (core radius), $r_s = \texttt{r\_s}$ (scale radius), and $r_t = |\texttt{r\_t}|$ (truncation radius). The latter three should satisfy the relation $0 \le r_c < r_s < r_t \le \infty$ (in practice $\texttt{r\_t} = 0$ is interpreted as $r_t = \infty$). The functional form of the density is

$$\rho_s(r) = \frac{C\,\mathrm{T}(r/r_t)}{x^{\gamma_0}(x^\eta + 1)^{(\gamma_\infty - \gamma_0)/\eta}} \qquad \text{with} \qquad x = \frac{\sqrt{r^2 + r_c^2}}{r_s} \tag{1}$$

where the truncation function is taken to be

$$\mathrm{T}(z) = \begin{cases} \mathrm{sech}(z) = \dfrac{2}{\exp(z) + 1/\exp(z)} & \text{for } \texttt{r\_t} > 0, \\ \dfrac{2}{\mathrm{sech}(z) + 1/\mathrm{sech}(z)} & \text{for } \texttt{r\_t} < 0. \end{cases} \tag{2}$$

Both forms decay exponentially at large radii, but at small radii they differ: $\mathrm{sech}(z) \approx 1 - \frac{1}{2}z^2$ for $z \ll 1$, while the second form $\approx 1 - \frac{1}{8}z^4$, i.e. at radii $r < r_t$ the density is less affected. The constant C in (2) is determined such that the total mass equals M. The distribution function is of Cuddeford (1991) type

$$f(E, L) = L^{-2\beta_0}\, g\left(-E - \frac{L^2}{r_a^2}\right), \tag{3}$$

where $\beta_0 = \texttt{b}$ (central anisotropy) and $r_a = \texttt{r\_a}$ (anisotropy radius) are free parameters, to within the bounds of what is physically possible (i.e. does not require $g < 0$), and limited by the implementation to $\beta_0 > -0.5$ (the code interpretes $\texttt{r\_a} = 0$ as $r_a = \infty$). The exact limits depend on the density distribution and the underlying external potential, but the code will complain if you stray outside. This choice of distribution function produces a velocity anisotropy

$$\beta \equiv 1 - \frac{\sigma_\theta^2}{\sigma_r^2} = \frac{r^2 + \beta_0\, r_a^2}{r^2 + r_a^2}. \tag{4}$$

The number of bodies in the spheroid is N=`nbody`. If you want to specify the softening length for the bodies of each component individually (as opposed to having the same softening length for all bodies), you can use the parameter `eps` to give all spheroid bodies the individual softening length $\epsilon_i = \texttt{eps}$.

Finally, there is a danger that the distribution function may be non-monotonic in some cases. This does not necessarily mean that the halo produced is unstable. However, it can cause trouble with the routines which determine the body velocities. To work around this **`mkhalo`** has the parameter `careful`, which, if set to be true (‘`careful=t`’) allows for this possibility and creates a halo even in case of a non-monotonic distribution function, but at an increased computational cost.

1.2 Potential of other components

The potentials of the other components of the galaxy are given as NEMO external potentials. For this the external potentials **`Halo`**, **`Monopole`**, **`DiscPot`** and (to be used later) **`PotExp`** are provided by falcON. From the script **`mkgalaxy`**, we have[1]:

```
accname=Halo+Monopole
accpars="0,$Rb,$Mb,$innerb,$outerb,$etab,$Rtb,$Rcoreb;1,10"
accfile=";$name.prm"
```

[1]Here and below, when we cite from the shell script **`mkgalaxy`**, we give shell variables such as `$innerb` and `$name` instead of their values, which are, of course, passed to the programs, such as **`mkhalo`**.

The keyword accname specifies the name of the external potential routines (the program searches for it in a certain path, but you can specify that path explicitly with the keyword accpath), in this case **Halo** (for the potential of the bulge component) and **Monopole** (for the spherically averaged potential of the disc component).

The keyword accpars provides the (comma-separated) parameter lists (separated by a semicolon[2]) for the two external potentials. **Halo** provides the potential generated by a density model of the form (2) and the parameters are (in order): Ω (ignored), r_s, M, γ_0, γ_∞, η, r_t, r_c. (In the above example, we are telling **mkhalo** about the potential of the bulge, so the parameters given are those of the bulge, which is reflected by the names ending in 'b': $innerb, $outerb etc..) For **Monopole** the parameters control the growth of the full potential (as described via keyword accfile, see below) from its monopole. The two numbers given are the start and end time of the growth of the full potential. This is important when growing the full potential from its monopole in simulation, but for finding the initial spheroid density profiles we just need to use the monopole, so only need to ensure that the growth start time is after $t = 0$.

The keyword accfile provides the names (separated by semicolon) of files which **Halo** or **Monopole** need to find a specific potential. **Halo** does not need any, so there is no file name before the semicolon. **Monopole** needs to know the contents of the file (referred to by) $name.prm, which tells **Monopole** the full potential to provide the monopole of. In that file, we must put the full disc potential, described in much the same way as here. **mkgalaxy** generates a file $name.prm with the contents

```
accname=DiscPot
accpars=0,$Sd_dp,$Rd,$Zd_dp,0,0
```

i.e. the disc potential is given by **DiscPot** with input parameters 0,$Sd_dp,$Rd,$Zd_dp,0,0. The external potential **DiscPot** provides the gravitational potential generated by a disc with spatial density

$$\rho(R,z) = \Sigma_0 \, \exp\left(-\frac{R_0}{R} - \frac{R}{R_d} + \varepsilon\cos\frac{R}{R_d}\right) \times \begin{cases} \frac{1}{2z_d}\exp\left(-\frac{|z|}{z_d}\right) & \text{for } z_d > 0 \\ \delta(z) & \text{for } z_d = 0 \\ \frac{1}{4|z_d|}\text{sech}^2\frac{z}{2|z_d|} & \text{for } z_d < 0 \end{cases} . \tag{5}$$

It employs the **GalPot** package, which implements the Milky-Way mass models of Dehnen & Binney (1998) and has been borrowed for this use. The input parameters provided via accpars are (in that order) Ω (ignored), Σ_0 (central surface density), R_d (scale length), z_d (scale height), R_0 (central hole radius, not used here), and ϵ (cosine modulation, also not used here).

N.B. DiscPot uses the same system of units as **GalPot**, which for historic reasons differs from that usually employed. As a consequence, we must multiply any mass dimension by 222293.02, see also the example shell script.

2 Growing the full disc potential

In the second step, we adjust the halo (and bulge) initial model to the presence of the full disc potential, rather than only its monopole. This is done by running a constrained N-body simulation using **gyrfalcON**, with the body distribution as our initial conditions and with an external potential that slowly changes from the monopole of the disc to its full potential.

While we are doing this, there is a real danger of the body distribution drifting away from the origin, which can cause serious problems in generating the disc initial conditions later. Therefore, it is essential to symmetrise the body distribution of the halo and bulge *before* and *throughout* the simulation. This means that for every body with phase-space coordinates $\boldsymbol{w} \equiv (\boldsymbol{x}, \boldsymbol{v})$, we have another one at $-\boldsymbol{w}$.

[2] Any keyword argument containing a semicolon must be enclosed in quotes, for otherwise the shell will be confused (independently of whether run from within a script or from the command line).

To ensure this for the initial model, we use the program **symmetrize** with the parameter `use=1`. This arranges the bodies in pairs such that $\boldsymbol{w}_{2i+1} = -\boldsymbol{w}_{2i}$, and does so by doubling the total number of bodies (this is why we originally generated only half as many bodies for halo and bulge). After this is done we run our simulation in **gyrfalcON**, using the manipulator **symmetrize_pairs**, which ensures after every block-step that $\boldsymbol{w}_{2i+1} = -\boldsymbol{w}_{2i}$. This is done by setting the **gyrfalcON** parameter `manipname=symmetrize_pairs`.

The growth of the full disc potential from its monopole is handled by the routines in **Monopole**. Since the **DiscPot** disc model, and the file that tells **Monopole** where to find the disc model were set up in step 1, this is relatively straightforward. The only new things are the timings for the start and end point of the disc growth. We set `accpars=0,$tgrow`. We have found it appropriate to take `$tgrow` $= 40$ in the case where we have disc mass $= 1$, disc scale length $= 1$, and $G = 1$. We run the simulation until $t = 60$ to allow the N-body model to settle fully into the new potential. This choice has worked perfectly well for us, but has not been rigorously tested to ensure that it is ideal. We include it only as a suggestion. Models with a different choice of units will require a different value for `$tgrow`.

Other than this, the simulation is an ordinary **gyrfalcON** simulation, and we refer you to the documentation on that in the user guide. One thing to note is that if you have given each component it's own softening length, you must set the parameter `eps` < 0, otherwise set it to the (global) softening length you desire. This step takes by far the longest time in the whole process. One can save a great deal of computer time by using pre-made haloes and bulges with disc models that have the same density profiles as in previous simulations, but different kinematic properties, i.e. skipping the steps described in sections 1 and 2.

3 Populating the disc

The final step is to populate the disc component, using the program **mkWD99disc**. To do this, the program needs to know the parameters of the disc model and the potential of any additional components (halo and bulge).

3.1 Disc parameters

The parameters of the disc model are $N_d =$ `nbody` (number of bodies), $R_d =$ `R_d` (disc scale length), $\Sigma_0 =$ `Sig_0` (central surface density), $z_d =$ `z_d` (disc scale height), $Q_0 =$ `Q` (normalisation for Toomre's Q), and $R_\sigma =$ `R_sig` (velocity-dispersion scale length). The disc has a target density profile

$$\rho(R,z) = \frac{1}{2\,z_0}\Sigma_0 \exp\left(-\frac{R}{r_d}\right)\mathrm{sech}^2\left(\frac{z}{z_0}\right), \tag{6}$$

equivalent to (5) for $z_d = -z_0/2$, $R_0 = 0$, and $\epsilon = 0$, and corresponding to a disc mass $M_d = 2\pi R_d^2 \Sigma_0$.

The target radial velocity dispersion in the plane, $\sigma_R(R, z = 0)$, is determined by the choice of the parameters Q_0, and (optionally) R_σ. If $R_\sigma = 0$ (default), the target velocity dispersion is such that $Q(R) = Q_0$ at all radii, where (Toomre, 1964)

$$Q(R) = \frac{\sigma_R(R)\,\kappa(R)}{3.36\,G\,\Sigma(R)}. \tag{7}$$

If $R_\sigma \neq 0$, then $\sigma_R \propto \exp(-R/R_\sigma)$, with the normalisation constant given by the condition that $Q(R_\sigma) = Q_0$.

The disc's thickness (z_0), and its isothermal profile define the vertical velocity dispersion such that $\sigma_z^2 = \pi G \Sigma(R) z_0$, independent of z, where $\Sigma(R)$ is the disc surface density.

The method for choosing the positions and velocities of bodies follows Dehnen (1999) and is also detailed in McMillan & Dehnen (2007). It, in effect, samples orbits in energy and angular momentum. The average number of bodies per orbit is given via the parameter `nbpero`. This produces a disc which has $\Sigma(R)$ and $\sigma_R(R)$ similar to those targeted, but not identical, with the difference being greater for warmer disc models. The code uses an iterative approach to tend towards the properties desired. The parameter `ni` gives the number of iterations (at least 1).

The parameter `eps` can be used to provide the same individual softening length $\epsilon_i =$ `eps` to all disc bodies.

3.2 Halo and bulge potential

As mentioned, **mkWD99disc** requires a description of the potential of halo and bulge. We use the routines in **PotExp** to find the smoothed, azimuthally averaged potential of these components. The **PotExp** external potential employs the potential expansion proposed by Zhao (1996), a generalisation of those due to Hernquist & Ostriker (1992) and Clutton-Brock (1973), and is given by

$$\Phi(\boldsymbol{x}) = \sum_{n=0}^{n_{\max}} \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} C_{nlm}\, \Phi_{nlm}(\boldsymbol{x}) \qquad \text{with} \qquad \Phi_{nlm}(\boldsymbol{x}) = \Psi_{nl}(r)\, Y_{lm}(\theta,\phi). \tag{8}$$

The lowest order radial basis function is (Zhao, 1996)

$$\Psi_{00} = -(r^{1/\alpha} + r_0^{1/\alpha})^{-\alpha}, \tag{9}$$

which for $\alpha = 1$ gives the potential of a Hernquist (1990) sphere, while $\alpha = 1/2$ gives a Plummer sphere (the cases considered by Hernquist & Ostriker 1992 and Clutton-Brock 1973, respectively). The coefficients C_{nlm} are determined by exploiting the *bi-orthogonality* relation ($4\pi\rho_{nlm} = \boldsymbol{\nabla}^2\Phi_{nlm}$)

$$\int \mathrm{d}^3\boldsymbol{x}\, \Phi_{nlm}(\boldsymbol{x})\, \rho_{n'l'm'}(\boldsymbol{x}) = -\delta_{nn'}\, \delta_{ll'}\, \delta_{mm'}$$

as

$$C_{nlm} = -\sum_i m_i\, \Phi_{nlm}(\boldsymbol{x}_i).$$

with the masses m_i and positions $\boldsymbol{x}_i$ taken from a data file (which must be in NEMO snapshot format) provided via the `accfile` keyword. The parameters provided via `accpars` are Ω (ignored), $\alpha \geq 1/2$, r_0, $n_{\max}$, $l_{\max}$, `symm`, and G. The last is Newton's constant of gravity, while `symm` specifies assumptions made about the underlying symmetry. For `symm`$= 0$, no assumptions are made; `symm`$= 1$ implies reflection symmetry w.r.t. the origin; `symm`$= 2$ means triaxial symmetry, i.e. reflection symmetry w.r.t. the x, y and z axes; `symm`$= 3$ means rotational symmetry w.r.t. the z axis (**N.B.** which is used here); finally `symm`$= 4$ refers to spherical symmetry. The symmetry constraint is implemented by imposing the corresponding constraint on the C_{nlm}.

Since the lowest order basis function corresponds to a model with inner logarithmic density slope of $2-1/\alpha$, a reasonable choice for α is $1/(2-\gamma)$, where γ is the slope of the distribution modelled, i.e. in the shell script **mkgalaxy** the values referred to by `$innerb` and `$innerh` for bulge and halo, respectively. While we could in principle use just one potential expansion for bulge and halo, we prefer using separate **PotExp** potentials for bulge and halo, adapted in α and r_0 (set equal to r_s of the respective component).

N.B. the program **mkWD99disc** will bail out if it finds that the total (disc plus external) gravitational force is directed outwards at any radius. This problem may well occur for various reasons. First, if the spheroid body distribution has wandered away from the origin (which we have prevented by enforcing point symmetry). Second, if the potential expansion is noisy (with low particles numbers and too many coefficients). In this latter case, it may be helpful to use smaller $n_{\max}$ and/or different α (via 'try & error').

4 Ready to use

The above steps should produce a galaxy as a NEMO snapshot, which can be manipulated and examined with many of the tools provided with NEMO. If you wish to convert it to other formats, then NEMO provides various tools, including the falcON programs **s2g** and **s2a** for conversion to, respectively, `gadget` and ASCII format, which can presumably be converted into any format you wish. These models have already been used for studies of mergers (McMillan, Athanassoula & Dehnen, 2007) and bars (Athanassoula, 2007).

References

Athanassoula E., 2007, MNRAS, 377, 1569
Clutton-Brock M., 1973, Ap&SS, 23, 55
Cuddeford P., 1991, MNRAS, 253, 414
Dehnen W., 1999, AJ, 118, 1201
Dehnen W., Binney J. J., 1998, MNRAS, 294, 429
Dehnen W., McLaughlin D. E., MNRAS, 363, 1057
Hernquist L., 1990, ApJ, 356, 359
Hernquist L., Ostriker J. P., 1992, ApJ, 386, 375
McMillan, P. J., Athanassoula E., Dehnen, W., 2007, MNRAS, 376, 1261
McMillan, P. J., Dehnen, W., 2007, MNRAS, 378, 541
Toomre A., 1964, ApJ, 139, 1217
Zhao H. S., 1996, MNRAS, 278, 488