NBODY6df

User Manual

James Petts[1]

University of Surrey

February 17, 2017

[1]j.petts@surrey.ac.uk jamesapetts@gmail.com

1 Introduction

NBODY6df is an extension of Sverre Aarseth's GPU + CPU parallel direct summation code, NBODY6 [1,8]. The module implements a thoroughly tested semi-analytic model of dynamical friction, allowing the user to include the drag force induced on a massive body when orbiting a spherical, isotropic galaxy model [3, 9, 10]. Dynamical friction is important for globular cluster evolution when the dynamical friction timescale is less than a Hubble time (e.g. in dwarf galaxies), and important for young star clusters when the timescale is comparable to the integration time of interest (.e.g in galactic centres).

1.1 Assumed knowledge of the reader

This document serves as a user manual for practical application of the NBODY6df code. For details about the mathematical theory and physics implemented, as well as vigorous comparison to full N-body simulations, see [9, 10]. Note that this manual assumes prior ability to compile/run the "standard" NBODY6 code. The user will need to change some flags or paths in the Makefile, depending on your system architecture/CUDA versions, etc, in order to compile. For information on general NBODY6 usage I refer the reader to Sverre Aarseth's website, containing documentation on NBODY6[1].

1.2 License

This code is open source under the MIT license to use/modify for your research or for any other purpose. If you use NBODY6df for a scientific publication we politely request you cite Petts et al 2015 & 2016 [9, 10].

1.3 Disclaimer

We have extensively tested NBODY6df and used it in several projects at the University of Surrey. We have ironed out all known edge cases/bugs, however, as it is impossible to test every astrophysical problem the code may be used for, there may be cases where the code doesn't function as intended. If you encounter such problems please contact me[2].

2 Implementation

Every DTADJ N-body times, adjust calls the subroutine dynfvars, where the cluster membership array BOUND(NMAX) and dynamical friction coefficient DYNFCOEF, are updated. This update is performed during the adjust

[1] https://www.ast.cam.ac.uk/ sverre/web/pages/nbody.htm

[2] j.petts@surrey.ac.uk jamesapetts@gmail.com

step for efficiency, as `dynfvars` requires cluster properties that are already calculated during the adjust step.

The dynamical friction coefficient is assumed to be constant until the next `adjust` call. This is a valid assumption so long as:

$$\texttt{DTADJ} \lesssim P/100, \tag{1}$$

where P is the orbital period.

During the regular integration step, external forces are applied to the stars by subroutine `xtrnlf`. It is here that the dynamical friction force is included, alongside the tides. The acceleration and the jerk are added to the regular forces of the "bound" particles.

Unlike the tidal forces which perform *relatively* little work on the cluster, dynamical friction can remove orbital energy equal to $\sim 10 - 100\times$ the internal energy of the system over the clusters lifetime. The energy removed from the cluster is calculated analytically, integrated to second order – as accurately as the tides – and added to `ETIDE`. However, due to the magnitude of energy removed this can lead to a relative energy drift of order $\sim 10^{-3}$ over a few thousand N-body times. It is unlikely this can be improved without an overhaul of the `NBODY6` integration scheme. I recommend testing that your new implementations conserve energy with dynamical friction switched off, before running with dynamical friction.

Note that the stars denoted as "bound" throughout this text are simply within the tidal radius of the cluster. This distinction is made as even if a star is "energetically unbound", it still contributes to the instantaneous dynamical friction force until it leaves the cluster [5, 9].

2.1 List of routines and `DYNFRI` common block variables

Table 1 shows a list of new routines, and routines modified in order to implement dynamical friction. If you wish to implement dynamical friction into to your custom version of `NBODY6`, Table 1 should make it rather straight forward. The modification of existing routines is mostly minimal, and the required changes can be found by searching for the tag "Jpetts". Table 2 shows a list of the variables in the new common block, `DYNFRI`.

2.2 Galaxy models

In the current version of `NBODY6df`, the models supported are Dehnen models [4] with asymptotic inner slope, $\gamma = 0, 0.5, 1.0, 1.5, 1.75$. For models with $\gamma \geq 0.5$, a Maxwellian approximation is made for the velocity distribution, which means `DYNFCOEF` is entirely analytic. For $\gamma = 0$ the Maxwellian approximation breaks down [10], and the number of slow moving stars must be computed numerically from the distribution function. For models with

Routine	Description
New	
`dynfapo`	Calculates the apocentre of the cluster orbit (by numerically solving Eq. 3.14 of [2]).
`dynfvars`	Calculates the dynamical friction coefficient, tidal radius and cluster membership.
`fdynfri`	Calculates the acceleration and the jerk on a star due to the dynamical friction force experienced by the star cluster.
`fv`	Calculates $f(v_*)$ from the isotropic distribution function, $f(E)$, at fixed radius. Only used for $\gamma = 0$ Dehnen models at the current time (see section §2.2).
Modified	
`adjust`	Added call to `dynfvars`.
`core`	Added a calculation of the average velocity of the stars in the core (saved to `VCORE`).
`common6.h`	Added common block `DYNFRI` containing variables relating to dynamical friction.
`gpucor`	Added ID of star to argument of `xtrnlf`.
`ksreg`	Added a re-arrangement of elements of the `BOUND` array upon KS-regularisation.
`ksterm`	Added a re-arrangement of elements of the `BOUND` array. upon KS-termination.
`lagr`	Now uses `MASSCL` calculated by `dynfvars` when calculating the half-mass radius. Therefore `RSCALE` is now the half mass radius of all stars within the tidal radius (i.e. not just the bound stars).
`mydump`	Now reads and writes the common block `DYNFRI` to restart files.
`nbint`	Added ID of star to argument of `xtrnlf`. (Zero passed as only include friction on the regular step).
`nbintp`	Added ID of star to argument of `xtrnlf`. (Zero passed as only include friction on the regular step).
`output`	Now writes `NBOUND` and `BOUND(NMAX)` to file.
`xtrlnd`	Added ID of star to argument of `xtrnlf`.
`xtrnlf`	Added dynamical friction term to list of external forces (calls `fdynfri`).
`zero`	Added initialisation of variables in the common block `DYNFRI`.

Table 1: Table displaying the routines added to NBODY6df and the routines that were modified.

Variable	Type	Description
COULOG	real*8	The Coulomb logarithm, $\log(\Lambda^2 + 1)$.
ONEPI	real*8	π, calculated at initialisation in zero.f.
RGDENSMAG	real*8	Galactocentric distance of the density centre of the cluster.
VCORE(3)	real*8	average velocity of stars in the core.
MASSCL	real*8	Mass of bound material.
BOUND(NMAX)	integer*4	array of bound stars.
NBOUND	integer*4	Number of bound stars.
DYNFCOEF	real*8	Dynamical friction coefficient.
DFOFF	logical	A switch that tells dynamical friction to turn off in the tidal stalling regime [10].

Table 2: Table displaying a list of the variables inside the new common block, DYNFRI

$\gamma \geq 0.5$, the user has the option of including a central supermassive black hole to the potential.

One may include a Miyamoto-Nagai disk [7] in their galaxy by default in NBODY6, however, by default the cluster will experience no drag from the disk, and one would underestimate the dynamical friction. If one would like to include the drag from a disk component, I urge the ambitious reader to test the correspondence with full N-body simulations, as one would have to construct an additional dynamical friction model suitable for friction induced by a flattened system. One may also need to consider the effect the disk has on the halo's $f(v_*)$.

If one wishes to use a galaxy model other than the Dehnen profile, one would first have to implement it into NBODY6. The user must then make sure to update the definitions of the density profile, $\rho(r)$, maximum impact parameter, $b_{\max}$, and the tidal radius, $r_{\rm t}$, in dynfvars. The user must also change the acceleration contribution from the potential in fdynfri.

Finally the user must replace the velocity distribution function, $f(v_*)$, and the potential, $\Phi(r)$, in fv.f and dynfapo, respectively. If the distribution function is non-analytic, one would need to numerically integrate the Eddington equation to obtain $f(v_*)$. I recommend fitting a similar density profile to your galaxy that has an analytic distribution function instead.

3 Running a simulation

The dynamical friction implementation requires integration of the cluster orbit, as such it requires the input file option KZ(14) = 3. It should be noted that one should not use KZ(31) $\geq$ 1 (adjustment of the guide centre), as this conflicts with the dynamical friction implementation. One may use

any escaper removal criteria they wish (i.e. `KZ(23)` = 0..3). If one chooses to keep all the particles in the simulation, particles that have escaped the cluster will, of course, feel no frictional force. If a star re-enters the tidal radius of the cluster, it will once again experience a frictional force.

If the orbital period changes significantly over the integration timescale, one can pause the simulation and restart it with a smaller `DTADJ` to ensure accuracy.

3.1 Example galaxy/orbit conditions

When one runs a simulation with `KZ(14) = 3`, `NBODY6(df)` requests an additional two lines in the input file, that specify the galaxy properties and initial orbit. Details on these two lines can be found in `define`.

Example initial conditions for a cluster initially on a circular orbit at 300 pc in a 3×10^8 $M_\odot$ cuspy dwarf spheroidal with $\gamma = 1$ and a scale radius of 1 kpc:

```
0.0 0.0 0.0 0.0 0.0 0.0 300000000.0 1.0 1.0
0.300 0.0 0.0 0.0 15.134171120446554 0.0
```

Where the last 3 terms on the top line are the halo mass, scale radius and asymptotic inner slope, γ, and the bottom line describes the initial orbit: x, y, z, v_x, v_y, v_z. Note that the scale radius of the Dehnen model is specified in kpc, whereas the initial position of the cluster is specified in pc. To include a black hole, simply change the first field of the first line, `GMG`, to the black hole mass. Note this can't be used with $\gamma = 0$ by default. If one wants to use a potential other than a Dehnen model, please consult section §2.2.

3.2 Output and analysis

`NBODY6(df)` saves information about the particles and cluster orbit to a binary file called `OUT3`. Included with the code is a script named "`out3_to_hdf5_nbody6df.py`", which converts this binary file to a hdf5 file containing important information about each snapshot. The hdf5 file format is easily readable in Python with the publicly available "`h5py`" library[3].

At run-time, the cluster drifts off the initial guide centre due to dynamical friction. This is accounted for when converting the binary data to hdf5 format. The script corrects the positions and velocities of the stars so that they are with respect to the infalling density centre of the cluster. Note that the reference frame changes with time - even for a circular orbit. To run the script, make it executable (you will also need to change the path to your python installation in the script's header), and run from the command line:

[3] http://www.h5py.org/

Directory	Variable	Description
stars	mass	Array of stellar masses ($M_\odot$).
	pos	Array of the stellar positions (pc).
	vel	Array of the stellar velocities (kms$^-$1).
	id	Array of unique stellar identifiers.
	kstar	Stellar type, -1 to 14 [6]. (If stellar evolution is on).
	lum	Stellar luminosity (log $L_\odot$) (If stellar evolution is on).
	teff	Effective surface temperature (log K) (If stellar evolution is on).
cluster	age	Physical simulation time (Myr).
	nbin	Number of regularised binaries.
	nbound	Number of stars within the tidal radius.
	ncore	Number of stars in the core.
	rcore	The core radius.
	rg	Position of the cluster relative to the galactic centre (pc).
	vg	Velocity of the cluster centre relative to the galactic centre (kms$^-$1).
	rhm	The half mass radius of all stars within the tidal radius (pc).

Table 3: Table describing the output variables of each snapshot.

```
./out3\_to\_hdf5\_nbody6df.py
```

when running without stellar evolution, or:

```
./out3\_to\_hdf5\_nbody6df.py -s
```

when running with stellar evolution switched on. If you have multiple runs (e.g. from restarting the code), you can combine all the `OUT3` files into a single hdf5 file with:

```
./out3\_to\_hdf5\_nbody6df.py -f ‘‘run1/OUT3 run2/OUT3’’
```

In the resulting hdf5 file, each snapshot is saved as a directory, with a sub-directory of “cluster” and “stars”. The “cluster” directory contains information on the cluster orbit and the global cluster properties, whereas the “stars” directory contains information on each star. Note that the order of stars may change between each snapshot, if one wants to refer to an individual star they must look for the star’s unique ID. E.g. if `id[42]=1`, then `pos[42]` is the position of the star with an ID of 1 in that snapshot. Table 3 describes the output variables contained within each snapshot.

4 Acknowledgements

I thank the University of Surrey for funding the project that led to the development of this code. In alphabetical order, I thank Fabio Antonini, Filippo Contenta, Maxime Delorme, Mark Gieles, Alessia Gualandris, Douglas Heggie, David Munroe, Matthew Orkney and Justin I. Read for useful contributions and discussions. I thank Mark Gieles especially for the basis of the "`out3_to_hdf5_nbody6df.py`" script. I am sincerely grateful to Sverre Aarseth and Keigo Nitadori for making the GPU-parallelised version of `NBODY6` public.

Bibliography

[1] S. J. Aarseth. From NBODY1 to NBODY6: The Growth of an Industry. *PASP*, 111:1333–1346, November 1999.

[2] J. Binney and S. Tremaine. *Galactic Dynamics*. Princeton Series in Astrophysics. Princeton University Press, second edition, 2008.

[3] S. Chandrasekhar. Dynamical Friction. I. General Considerations: the Coefficient of Dynamical Friction. *APJ*, 97:255, March 1943.

[4] W. Dehnen. A Family of Potential-Density Pairs for Spherical Galaxies and Bulges. *MNRAS*, 265:250, November 1993.

[5] M. Fujii, Y. Funato, and J. Makino. Dynamical Friction on Satellite Galaxies. *pasj*, 58:743–752, August 2006.

[6] J. R. Hurley, O. R. Pols, and C. A. Tout. Comprehensive analytic formulae for stellar evolution as a function of mass and metallicity. *mnras*, 315:543–569, July 2000.

[7] M. Miyamoto and R. Nagai. Three-dimensional models for the distribution of mass in galaxies. *pasj*, 27:533–543, 1975.

[8] K. Nitadori and S. J. Aarseth. Accelerating NBODY6 with graphics processing units. *MNRAS*, 424:545–552, July 2012.

[9] J. A. Petts, A. Gualandris, and J. I. Read. A semi-analytic dynamical friction model that reproduces core stalling. *mnras*, 454:3778–3791, December 2015.

[10] J. A. Petts, J. I. Read, and A. Gualandris. A semi-analytic dynamical friction model for cored galaxies. *mnras*, 463:858–869, November 2016.