Discussion on the perspectives of NBODY6/7 with Sverre Aarseth

chair: Rainer Spurzem, notes: Václav Pavlík

September 20, 2017

NBODY6 (Aarseth, 2003) is a direct N-body integrator written in `FORTRAN`. The newest version of the code has been upgraded to include the SSE[1] and GPU compatibility (Nitadori & Aarseth, 2012).

The user manual (current version from Nov 13, 2015, to be updated) may be found at `ftp://ftp.ast.cam.ac.uk/pub/sverre/nbody6/man6.pdf`.

1 Few-body subsystems

NBODY6 uses the Ahmad & Cohen (1973) integration scheme with a fourth-order Hermite integrator (Makino, 1991). In order to be effective, NBODY6 contains several procedures for special purposes, e.g. for the integration of binaries (Mikkola & Aarseth, 1998). In principle multiple methods may be combined, because the code is the same before and after these routines. However, the bottleneck of most calculations is the *decision making*, i.e. when to switch the equations that are integrated (methods to use).

For instance, we mention the chain regularization (Mikkola & Aarseth, 1993) that is intended for the integration of a binary and a strong perturber. However, after integrating three particles in a subsystem, more members may be needed, e.g. two colliding binaries. The code will allow a chain of arbitrary length but this is not practical. Every time we include another body in the chain, we must add more external perturbers which must be predicted frequently and this increases the cost. The best way is to use this procedure for three to five bodies, i.e. two binaries with a perturber, or a small hierarchical system.

2 Data structure

The code is intended to be as clean as possible in the output, but we must prepare it. Data are stored in several dedicated binary files. To keep track of the stars, each member has its own identifier. The main output contains arrays for mass, coordinates, velocities, etc. If we wish to follow the stellar evolution we may include the stellar type, e.g. add `(KSTAR(J),J=1,NTOT)` in the `WRITE (3)` statement in `output.f`.

The data structure is generalised. For binaries, we save their relative motion and the corresponding centre of mass. In the case of stable (hierarchical) multiples, ghost particles with zero mass may appear in the output if printed without the pre-processing. In the latter case, all the components should be included in one table.

A custom output can be done but we need to be consistent with the data structure. However, beware that almost all units of the `FORTRAN` output are already used for something. Before printing other results, the list of units used (`fort.x` in directory `Docs`) should be consulted to avoid overwriting. The precision of the existing outputs can be changed. Sometimes more decimal places in the output is desirable to catch what is going on (but you cannot please everybody).

3 NBODY6 & NBODY7

The main difference between the two versions of the code is in the post-Newtonian expansion present in NBODY7. It was included mainly for educational purposes but has been expanded to the treatment of black holes. We definitely need improvements! NBODY7 has different orders for the post-Newtonian. The cross terms for the third body are quite complicated – we need *a volunteer* to do it. To get the NBODY7 version, compile with the source code of NBODY6 after changing two lines in the `Makefile`. An attempt to make the parallel block-step scheme threadsafe is underway.

4 Discussion

4.1

If there is a binary BH and a possible third member, Sambaran found that before the BH coalescence, the chain breaks and the third member is lost without any output.

Answer

In the case of BH coalescence with only two members of the chain, it is necessary to include some ex-

[1]Streaming SIMD (Single Instruction Multiple Data) Extensions

tra routine to perform the desired diagnostic analysis for the most dominant perturber. An existing routine `bhplot.f` could be modified for this purpose, using negative value for option `KZ(45)` inside.

4.2

Wang Long recommended to update the `common6.h` with variable names, so that the code is more readable and comprehensive for the user. Also to keep consistent names to recognise global and local variables. **To do that, we need volunteers.**

4.3

How far has the inclusion of NBODY6 to AMUSE gone?

Answer

Interface is working for a version of NBODY6 from 2013. Maxwell has a plan to include the latest NBODY6 and NBODY6++. He has also a Python interface for running NBODY6. The current progress is on GitHub.

4.4

Would it be easy to switch the stellar evolution code?

Answer

In principle yes, but there is still work to do. Single stars are "easy", the binaries are complicated.

4.5

We need a comprehensive **documentation** on platforms and compilers. There is not yet any suite of test problems and examples. In NBODY6++ there are figures as a guide what to use for a specific kind of problem. However, in NBODY6, there are two README files in the directory `GPU2` for system-related matters.

4.6

Interaction between the GPUs and CPUs on the machine needs to be done by the user, there is no automated way built in. However, standard default parameters are set and there are input templates.

4.7

Is NBODY6 and NBODY7 compatible? Can we use dumps from one of them to run in the other one?

Answer

So far, we do not have a definite answer, however, Sverre was unsuccessful to restart NBODY7 dump from NBODY6.

From a recent differencing of `mydump.f` it can be seen that the limit for writing and reading `KBLIST(I)` in the directory `Ncode` should be taken as `NPAIRS` instead of `KMAX`, which fixes the problem. It is desirable to enable restart from either code but care must be taken when dealing with old data.

4.8

There is missing documentation in a number of routines. **A complete documentation is needed and would be useful.** Perhaps do it on Wiki or GitHub.

Answer

There is actually a GitHub site for NBODY6 `https://github.com/nbodyx/nbody6/wiki`, however, not maintained from 2016. We address the **community for help**, e.g. with test runs, FAQ etc. **This should be done.**

4.9

What is the best way to simulate a cluster model in a galactic field?

Answer

Routines exist in NBODY6 or there is NBODY6tt for cosmological purposes (a separate branch).

4.10

From the user point of view, is it possible to do gas expulsion/tidal field within NBODY6?

Answer

Already possible. This can be done with setting a tidal tail potential, using `KZ(14)` (e.g. `=3` for galactic disk, bulge, halo).

4.11

The stellar evolution should be done better. At least test against newest results.

4.12

Maxwell and Rainer pointed out that there is `HDF5` output in NBODY6++ which is better for frequent output, as it only records those particles that have been integrated in a particular block. Flag `KZ(46)`. It is readable by Python, R etc.

5 NBODY6++

NBODY6++ is a parallel multi-CPU multi-GPU code (using openMP, MPI and CUDA) which is based on

Spurzem (1999) and has been much improved and accelerated by Long Wang (Wang et al., 2015, see GitHub link for sources and more documentation therein). It is based on NBODY6, but not identical to it. For questions or email list membership, contact Rainer Spurzem `spurzem@ari.uni-heidelberg.de`.

It has a comprehensive manual, including flowcharts and many more, originally written by Emil Khalisi, but updated by others later, which can be found on `http://silkroad.bao.ac.cn/repos/nbody6++gpu/doc/`.

Most discussion points above also apply to NBODY6++.

5.1 Future work directions specific for NBODY6++

Improving KS binary parallelization, data structure (domain decomposition), project for research software sustainability including international collaboration is currently under review at DFG (German Science Foundation), see `http://wwwstaff.ari.uni-heidelberg.de/mitarbeiter/spurzem/software.html`

6 Conclusions

NBODY6 should be considered a project. It is not a dead code.

Acknowledgements

Thanks to Sverre and Rainer for their comments and edits.

References

Ahmad, A. & Cohen, L. (1973). A numerical integration scheme for the N-body gravitational problem. *Journal of Computational Physics*, 12:389–402.

Makino, J. (1991). A Modified Aarseth Code for GRAPE and Vector Processors. PASJ, 43:859–876.

Mikkola, S. & Aarseth, S. J. (1993). An implementation of N-body chain regularization. *Celestial Mechanics and Dynamical Astronomy*, 57:439–459.

Mikkola, S. & Aarseth, S. J. (1998). An efficient integration method for binaries in N-body simulations. New A, 3:309–320.

Spurzem, R. (1999). Direct N-body Simulations. *Journal of Computational and Applied Mathematics*, 109:407–432.

Aarseth, S. J. (2003). *Gravitational N-Body Simulations.* Cambridge University Press, Cambridge, UK.

Nitadori, K. & Aarseth, S. J. (2012). Accelerating NBODY6 with graphics processing units. MNRAS, 424:545–552.

Wang, L., Spurzem, R., Aarseth, S., Nitadori, K., Berczik, P., Kouwenhoven, M. B. N., & Naab, T. (2015). NBODY6++GPU: ready for the gravitational million-body problem. MNRAS, 450:4070–4080.